User's Manual

Bluetooth® Low Energy Protocol Stack

API Reference Manual: PASP

Renesas MCU

Target Device

RL78/G1D

Renesas Electronics
www.renesas.com

Rev.1.01 Oct 2015

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.

   “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.

   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(2012.4)

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins

   Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.

   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment when power is supplied.

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of an MPU or MCU in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

How to Use This Manual

1. Purpose and Target Readers

This manual describes the API (Application Program Interface) of the Phone Alert Status profile (PASP) of the Bluetooth Low Energy protocol stack (BLE software), which is used to develop Bluetooth applications that incorporate the Renesas Bluetooth low energy microcontroller RL78/G1D. It is intended for users designing application systems incorporating this software. A basic knowledge of microcontrollers and Bluetooth low energy is necessary in order to use this manual.

Related documents

The related documents indicated in this publication may include preliminary versions. However, preliminary versions are not marked as such.

Document Name	Document No.
Bluetooth Low Energy Protocol Stack	
User's Manual	R01UW0095E
API Reference Manual: Basics	R01UW0088E
API Reference Manual: FMP	R01UW0089E
API Reference Manual: PXP	R01UW0090E
API Reference Manual: HTP	R01UW0091E
API Reference Manual: BLP	R01UW0092E
API Reference Manual: HOGP	R01UW0093E
API Reference Manual: ScPP	R01UW0094E
API Reference Manual: HRP	R01UW0097E
API Reference Manual: CSCP	R01UW0098E
API Reference Manual: CPP	R01UW0099E
API Reference Manual: GLP	R01UW0103E
API Reference Manual: TIP	R01UW0106E
API Reference Manual: RSCP	R01UW0107E
API Reference Manual: ANP	R01UW0108E
API Reference Manual: PASP	This manual
API Reference Manual: LNP	R01UW0113E
Application Note: Sample Program	R01AN1375E
Application Note: rBLE Command Specification	R01AN1376E

List of Abbreviations and Acronyms

Abbreviation	Full Form	Remark
ANP	Alert Notification Profile	
ANS	Alert Notification Service	
API	Application Programming Interface	
ATT	Attribute Protocol	
BAS	Battery Service	
BB	Base Band	
BD_ADDR	Bluetooth Device Address	
BLE	Bluetooth low energy	
BLP	Blood Pressure Profile	
BLS	Blood Pressure Service	
CPP	Cycling Power Profile	
CPS	Cycling Power Service	
CSCP	Cycling Speed and Cadence Profile	
CSCS	Cycling Speed and Cadence Service	
CSRK	Connection Signature Resolving Key	
CTS	Current Time Service	
DIS	Device Information Service	
EDIV	Encrypted Diversifier	
FMP	Find Me Profile	
GAP	Generic Access Profile	
GATT	Generic Attribute Profile	
GLP	Glucose Profile	
GLS	Glucose Service	
HCI	Host Controller Interface	
HID	Human Interface Device	
HIDS	HID Service	
HOGP	HID over GATT Profile	
HRP	Heart Rate Profile	
HRS	Heart Rate Service	
HTP	Health Thermometer Profile	
HTS	Health Thermometer Service	
IAS	Immediate Alert Service	
IRK	Identity Resolving Key	
L2CAP	Logical Link Control and Adaptation Protocol	
LE	Low Energy	

Abbreviation	Full Form	Remark
LL	Link Layer	
LLS	Link Loss Service	
LNP	Location and Navigation Profile	
LNS	Location and Navigation Service	
LTK	Long Term Key	
MCU	Micro Controller Unit	
MITM	Man-in-the-middle	
MTU	Maximum Transmission Unit	
NDCS	Next DST Change Service	
OOB	Out of Band	
OS	Operating System	
PASP	Phone Alert Status Profile	
PASS	Phone Alert Status Service	
PXP	Proximity Profile	
RF	Radio Frequency	
RSCP	Running Speed and Cadence Profile	
RSCS	Running Speed and Cadence Service	
RSSI	Received Signal Strength Indication	
RTUS	Reference Time Update Service	
ScPP	Scan Parameters Profile	
ScPS	Scan Parameters Service	
SM	Security Manager	
SMP	Security Manager Protocol	
STK	Short Term Key	
TIP	Time Profile	
TK	Temporary Key	
TPS	Tx Power Service	
UART	Universal Asynchronous Receiver Transmitter	
UUID	Universal Unique Identifier	

Abbreviation	Full Form	Remark
APP	Application	
CSI	Clocked Serial Interface	
IIC	Inter-Integrated Circuit	
RSCIP	Renesas Serial Communication Interface Protocol	
VS	Vendor Specific	

Contents

1. Overview

This manual describes the API (Application Program Interface) of the Phone Alert Status profile (PASP) of the Bluetooth Low Energy protocol stack (BLE software), which is used to develop Bluetooth applications that incorporate Renesas Bluetooth low energy microcontroller RL78/G1D.

For details about the organization and features of BLE software, see the Bluetooth Low Energy Protocol Stack User's Manual.

2. Common Definitions

This section describes the definitions common to the API of each profile.

2.1 Service Definitions

This section describes the common definitions of services used by the API of multiple profiles.

- Declaration of enumerated type for alert level

```
enum RBLE_SVC_ALT_LVL_enum {
   RBLE_SVC_ALERT_NONE = 0x00,            No alert
   RBLE_SVC_ALERT_MILD,                   Mild alert
   RBLE_SVC_ALERT_HIGH                    High alert
};
```

- Declaration of enumerated type for PnP ID characteristic vendor ID field

```
enum RBLE_SVC_PNP_VENDOR_ID_enum {
   RBLE_SVC_SIG_ASSIGNED_ID = 0x01,       Vendor ID assigned by Bluetooth SIG
   RBLE_SVC_USB_ASSIGNED_ID               Vendor ID assigned by USB Implementer's
                                          Forum
};
```

- Declaration of enumerated type for Name Space field of Characteristic Presentation Format descriptor

```
enum RBLE_SVC_PRESEN_NAMESPASE_enum {
   RBLE_SVC_NAMESPACE_SIG = 0x01,         Defined by Bluetooth SIG
};
```

- Declaration of enumerated type for security level of Service

```
enum RBLE_SVC_SEC_LVL_enum {
   RBLE_SVC_SEC_NONE = 0x01,              No security
   RBLE_SVC_SEC_UNAUTH = 0x02,            Require unauthenticated pairing
   RBLE_SVC_SEC_AUTH = 0x04,              Require authenticated pairing
   RBLE_SVC_SEC_AUTZ = 0x08,              Require authorization
   RBLE_SVC_SEC_ENC = 0x10                Require encryption
};
```

- Declaration of enumerated type for connection types

```
enum RBLE_PRF_CON_enum {
    RBLE_PRF_CON_DISCOVERY = 0x00,        Configuration connection performed
                                          when connecting for the first time
    RBLE_PRF_CON_NORMAL                   Normal connection performed when
                                          connecting for the second and
                                          subsequent times
};
```

- Declaration of enumerated type for client configuration characteristic value

```
enum RBLE_PRF_CLIENT_CONFIG_enum {
    RBLE_PRF_STOP_NTFIND = 0x00,            Stop notification or indication of
                                            characteristic value.
    RBLE_PRF_START_NTF,                     Start notification of
                                            characteristic value.
    RBLE_PRF_START_IND                      Start indication of
                                            characteristic value.
};
```

- Declaration of enumerated type for server configuration characteristic value

```
enum RBLE_PRF_SERVER_CONFIG_enum {
    RBLE_PRF_STOP_BRD = 0x00,               Stop broadcast of characteristic value.
    RBLE_PRF_START_BRD                      Start broadcast of characteristic value.
};
```

2.2 Status Definitions

This section describes the status definitions used by the API of each profile.

- Declaration of enumerated type for rBLE status

```
enum RBLE_STATUS_enum {
    RBLE_OK = 0x00,                          Normal operation
    RBLE_PRF_ERR_INVALID_PARAM = 0x90,       Invalid parameter specified for
                                             setting or acquiring a characteristic
                                             value
    RBLE_PRF_ERR_INEXISTENT_HDL,             Invalid handle specified for setting
                                             or acquiring a characteristic value
    RBLE_PRF_ERR_STOP_DISC_CHAR_MISSING,     The characteristic value is missing.
    RBLE_PRF_ERR_MULTIPLE_IAS,               Multiple IASs exist.
    RBLE_PRF_ERR_INCORRECT_PROP,             Incorrect property
    RBLE_PRF_ERR_MULTIPLE_CHAR,              Multiple characteristic values exist.
    RBLE_PRF_ERR_NOT_WRITABLE,               Writing is not permitted.
    RBLE_PRF_ERR_NOT_READABLE,               Reading is not permitted.
    RBLE_PRF_ERR_REQ_DISALLOWED,             Requesting is not permitted.
    RBLE_PRF_ERR_NTF_DISABLED,               Notification is disabled.
    RBLE_PRF_ERR_IND_DISABLED,               Indication is disabled.
    RBLE_PRF_ERR_ATT_NOT_SUPPORTED,          The characteristic value is not
                                             supported.
};
```

Note: Statuses other than the above are described in *API Reference Manual: Basics*.

3. Phone Alert Status Profile

This section describes the API of the Phone Alert Status profile. The Phone Alert Status profile is used to enable a data collection device to obtain data from phone alert status server.

3.1 Definitions

This section describes the definitions used by the API of the Phone Alert Status profile.

- Declaration of enumerated type for PASP event types

```
enum RBLE_PASP_EVENT_TYPE_enum {
    RBLE_PASP_EVENT_SERVER_ENABLE_COMP = 0x01,   Server enable completion event
                                                 (Parameter: server_enable)
    RBLE_PASP_EVENT_SERVER_DISABLE_COMP,         Server disable completion event
                                                 (Parameter: server_disable)
    RBLE_PASP_EVENT_SERVER_ERROR_IND,            Server error indication event
                                                 (Parameter: error_ind)
    RBLE_PASP_EVENT_SERVER_SEND_ALERT_STATUS_COMP,   Alert Status send completion event
                                                 (Parameter: send_alert)
    RBLE_PASP_EVENT_SERVER_SEND_RINGER_SETTING_COMP, Ringer Setting send completion event
                                                 (Parameter: send_ringer)
    RBLE_PASP_EVENT_SERVER_CHG_RINGER_CP_IND,    Ringer control point change
                                                 indication event
                                                 (Parameter: chg_ringer_cp_ind)
    RBLE_PASP_EVENT_SERVER_CFG_NTF_IND,          Characteristic configuration change
                                                 indication event
                                                 (Parameter: cfg_ntf_ind)
    RBLE_PASP_EVENT_SERVER_COMMAND_DISALLOWED_IND, Command disallowed indication event
                                                 (Parameter: cmd_disallowed_ind)
    RBLE_PASP_EVENT_CLIENT_ENABLE_COMP = 0x81,   Client enable completion event
                                                 (Parameter: client_enable)
    RBLE_PASP_EVENT_CLIENT_DISABLE_COMP,         Client disable completion event
                                                 (Parameter: client_disable)
    RBLE_PASP_EVENT_CLIENT_ERROR_IND,            Client error indication event
                                                 (Parameter: error_ind)
    RBLE_PASP_EVENT_CLIENT_ALERT_STATUS_NTF,     Alert Status notification event
                                                 (Parameter: alert_ntf)
    RBLE_PASP_EVENT_CLIENT_RINGER_SETTING_NTF,   Ringer Setting notification event
                                                 (Parameter: ringer_ntf)
    RBLE_PASP_EVENT_CLIENT_READ_CHAR_RESPONSE,   Characteristic value read request
                                                 response event
                                                 (Parameter: rd_char_resp)
    RBLE_PASP_EVENT_CLIENT_WRITE_CHAR_RESPONSE,  Characteristic value write request
                                                 response event
                                                 (Parameter: wr_char_resp)
    RBLE_PASP_EVENT_CLIENT_COMMAND_DISALLOWED_IND Command disallowed indication event
                                                 (Parameter: cmd_disallowed_ind)
```

```
};
```

- Declaration of data type for PASP event types

```
typedef uint8_t RBLE_PASP_EVENT_TYPE;
```

- Declaration of data type for PASP Server event callback function

```
typedef void ( *RBLE_PASPS_EVENT_HANDLER )( RBLE_PASPS_EVENT *event );
```

- Declaration of data type for PASP Client event callback function

```
typedef void ( *RBLE_PASPC_EVENT_HANDLER )( RBLE_PASPC_EVENT *event );
```

- Declaration of enumerated type for reading phone alert status service characteristic codes

```
enum RBLE_PASPC_RD_CHAR_CODE_enum {
    RBLE_PASPC_RD_PASS_ALERT_STATUS = 0x00,      Alert Status
    RBLE_PASPC_RD_PASS_ALERT_STATUS_CFG,         Alert Status Notification
    RBLE_PASPC_RD_PASS_RINGER_SETTING,           Ringer Setting
    RBLE_PASPC_RD_PASS_RINGER_SETTING_CFG,       Ringer Setting Notification
};
```

- Declaration of enumerated type for setting phone alert status service characteristic codes

```
enum RBLE_PASP_WR_CHAR_CODE_enum {
    RBLE_PASP_ALERT_STATUS_CODE = 0x00,          Alert Status
    RBLE_PASP_RINGER_SETTING_CODE                Ringer Setting
};
```

- Declaration of enumerated type for ringer setting

```
enum RBLE_PASP_RINGER_SETTING_enum {
    RBLE_PASP_RINGER_SILENT = 0x00,              Ringer Silent
    RBLE_PASP_RINGER_NORMAL                      Ringer Normal
};
```

- Declaration of enumerated type for ringer mode

```
enum RBLE_PASP_RINGER_MODE_enum {
    RBLE_PASP_SILENT_MODE = 0x01,                Silent Mode
    RBLE_PASP_MUTE_ONCE,                         Mute Once
    RBLE_PASP_CANCEL_SILENT_MODE                 Cancel Silent Mode
};
```

- Declaration of enumerated type for alert status

```
enum RBLE_PASP_ALERT_STATE_BIT_enum {
    RBLE_PASP_RINGER_STATE_BIT = 0x00,           Ringer State
    RBLE_PASP_VIBRATOR_STATE_BIT,                Vibrator State
    RBLE_PASP_DISPLAY_ALERT_STATE_BIT            Display Alert Status
};
```

- Phone Alert Status service characteristic information structures

```
typedef struct RBLE_PASP_SERVER_PARAM_t{
    uint16_t      alert_status_ntf_en;    Alert Status notification configuration value
    uint16_t      ringer_setting_ntf_en;  Ringer Setting notification configuration value
}RBLE_PASP_SERVER_PARAM;
```

- Phone Alert Status service content structures

```
typedef struct RBLE_PASS_CONTENT_t{
    uint16_t        shdl;                   Phone Alert Status service start handle
    uint16_t        ehdl;                   Phone Alert Status service end handle
    uint16_t        alert_status_char_hdl;      Alert Status characteristic handle
    uint16_t        alert_status_val_hdl;       Alert Status characteristic
                                                value handle
    uint16_t        alert_status_cfg_hdl;       Alert Status characteristic
                                                configuration descriptor handle
    uint8_t         alert_status_prop;          Alert Status characteristic property
    uint8_t         reserved1;                  Reserved
    uint16_t        ringer_setting_char_hdl;    Ringer Setting characteristic handle
    uint16_t        ringer_setting_val_hdl;     Ringer Setting characteristic value
                                                handle
    uint16_t        ringer_setting_cfg_hdl;     Ringer Setting characteristic
                                                configuration descriptor handle
    uint8_t         ringer_setting_prop;        Ringer Setting characteristic property
    uint8_t         reserved2;                  Reserved
    uint16_t        ringer_cp_char_hdl;         Ringer Control Point
                                                characteristic handle
    uint16_t        ringer_cp_val_hdl;          Ringer Control Point
                                                characteristic value handle
    uint8_t         ringer_cp_prop;             Ringer Control Point
                                                characteristic property
    uint8_t         reserved2;                  Reserved
}RBLE_PASS_CONTENT;
```

- PASP Server event parameter structures

```
typedef struct RBLE_PASPS_EVENT_t {
    RBLE_PASP_EVENT_TYPE            type;           PASP event type
    uint8_t                         reserved;       Reserved
    union Event_Pass_Parameter_u {
        Generic event
        RBLE_STATUS                 status;         Status

        Server enable completion event
        struct RBLE_PASP_Server_Enable_t{
            RBLE_STATUS             status;         Status
            uint8_t                 reserved;       Reserved
            uint16_t                conhdl;         Connection handle
        }server_enable;
```

Server disable completion event

```
struct RBLE_PASP_Server_Disable_t{
    uint16_t                  conhdl;         Connection handle
    RBLE_PASP_SERVER_PARAM  server_info;     Phone Alert Status Service
                                             information
}server_disable;
```

Server error indication event

```
struct RBLE_PASP_Server_Error_Ind_t{
    RBLE_STATUS             status;          Status
    uint8_t                 reserved;        Reserved
    uint16_t                conhdl;          Connection handle
}error_ind;
```

Server alert status value send completion event

```
struct RBLE_PASP_Server_Send_Alert_Status_t{
    RBLE_STATUS             status;          Status
    uint8_t                 reserved;        Reserved
    uint16_t                conhdl;          Connection handle
}send_alert;
```

Server ringer setting value send completion event

```
struct RBLE_PASP_Server_Send_Ringer_Setting_t{
    RBLE_STATUS             status;          Status
    uint8_t                 reserved;        Reserved
    uint16_t                conhdl;          Connection handle
}send_ringer;
```

Server characteristic value set completion event

```
struct RBLE_PASP_Server_Set_Data_Ind_t{
    RBLE_STATUS           status;           Status
}set_data;
```

Server ringer control point change indication event

```
struct RBLE_PASP_Server_Chg_Ringer_Cp_Ind_t{
    uint16_t                conhdl;          Connection handle
    uint8_t                 cp_val;          Control point value
}chg_ringer_cp_ind;
```

Server configuration characteristic value indication event

```
struct RBLE_PASP_Server_Cfg_Ntf_Ind_t{
    uint16_t                conhdl;          Connection handle
    uint8_t                 char_code;       Characteristic value code
    uint8_t                 reserved;        Reserved
    uint16_t                cfg_val;         Configuration characteristic
```

```
                                                    value
        }cfg_ntf_ind;

        Server command disallowed indication event
        struct RBLE_PASP_Server_Command_Disallowed_Ind_t{
            RBLE_STATUS             status;             Status
            uint8_t                 reserved;           Reserved
            uint16_t                opcode;             Opcode
        }cmd_disallowed_ind;
    } param;
} RBLE_PASPS_EVENT;
```

- PASP Client event parameter structures

```
typedef struct RBLE_PASPC_EVENT_t {
    RBLE_PASP_EVENT_TYPE          type;              PASP event type
    uint8_t                       reserved;          Reserved
    union Event_Paspc_Parameter_u {
        Generic event
        RBLE_STATUS               status;            Status

        Client enable completion event
        struct RBLE_PASP_Client_Enable_t{
            RBLE_STATUS           status;            Status
            uint8_t               reserved;          Reserved
            uint16_t              conhdl;            Connection handle
            RBLE_PASS_CONTENT     pass;              Phone Alert Status service
                                                     content
        }client_enable;

        Client disable completion event
        struct RBLE_PASP_Client_Disable_t{
            RBLE_STATUS           status;            Status
            uint8_t               reserved;          Reserved
            uint16_t              conhdl;            Connection handle
        }client_disable;

        Client error indication event
        struct RBLE_PASP_Client_Error_Ind_t{
            RBLE_STATUS           status;            Status
            uint8_t               reserved;          Reserved
            uint16_t              conhdl;            Connection handle
        }error_ind;

        Client alert status notification event
        struct RBLE_PASP_Client_Alert_Status_Ntf_t{
            uint16_t              conhdl;             Connection handle
            uint8_t               alert_status;       Alert status
        }alert_ntf;

        Client ringer_setting event
        struct RBLE_PASP_Client_Ringer_Setting_Ntf_t{
            uint16_t              conhdl;            Connection handle
            uint8_t               ringer_setting;    Ringer Setting
        }ringer_ntf;

        Client characteristic value read request response event
        struct RBLE_PASP_Client_Read_Char_Response_t{
```

```
            uint16_t                conhdl;             Connection handle
            uint8_t                 att_code;           Status
            uint8_t                 reserved;           Reserved
            RBLE_ATT_INFO_DATA      data;               Acquired characteristic data
        }rd_char_resp;

        Client characteristic value write request response event
        struct RBLE_PASP_Client_Write_Char_Response_t{
            uint16_t                conhdl;             Connection handle
            uint8_t                 att_code;           Status
        }wr_char_resp;

        Client command disallowed indication event
        struct RBLE_PASP_Client_Command_Disallowed_Ind_t{
            RBLE_STATUS             status;             Status
            uint8_t                 reserved;           Reserved
            uint16_t                opcode;             Opcode
        }cmd_disallowed_ind;
    } param;
} RBLE_PASPC_EVENT;
```

3.2 Functions

The following table shows the API functions defined for the PASP of rBLE and the following sections describe the API functions in detail.

Table 3-1 API Functions Used by the PASP

RBLE_PASP_Server_Enable	Enables the Server role.
RBLE_PASP_Server_Disable	Disables the Server role.
RBLE_PASP_Server_Send_Alert_Status	Sends the alert status information
RBLE_PASP_Server_Send_Ringer_Setting	Sends the ringer setting information.
RBLE_PASP_Client_Enable	Enables the Client role.
RBLE_PASP_Client_Disable	Disables the Client role.
RBLE_PASP_Client_Read_Char	Reads the characteristic value.
RBLE_PASP_Client_Write_Ringer_ControlPoint	Sets the ringer control point.
RBLE_PASP_Client_Write_Char	Writes the characteristic value.

3.2.1 RBLE_PASP_Server_Enable

RBLE_STATUS RBLE_PASP_Server_Enable(uint16_t conhdl, uint8_t sec_lvl, uint8_t con_type, RBLE_PASP_SERVER_PARAM *param, RBLE_PASPS_EVENT_HANDLER call_back)

This function enables the PASP Server role.
If the notification settings of the transmission data are configured from the Client, set the notification setting parameter to 0 to configure the connection. If this setting or information has been specified from the Server, perform a normal connection in accordance with the notification setting parameter.
The result is reported by using the Server role enable completion event RBLE_PASP_EVENT_SERVER_ENABLE_COMP.

Parameters:

conhdl	Connection handle		
sec_lvl	Security level		
con_type	RBLE_PRF_CON_DISCOVERY		Configuration connection
	RBLE_PRF_CON_NORMAL		Normal connection
*param	*alert_status_ntf_en*	RBLE_PRF_STOP_NTFIND	Stop notification of alert status information.
		RBLE_PRF_START_NTF	Start notification of alert status information.
	ringer_setting_ntf_en	RBLE_PRF_STOP_NTFIND	Stop notification of ringer setting information.
		RBLE_PRF_START_NTF	Start notification of ringer setting information.
call_back	Specify the callback function that reports the PASP event.		

Return:

RBLE_OK	Success
RBLE_ERR	Error occurred in Server role enable processing
RBLE_PARAM_ERR	Invalid parameter
RBLE_STATUS_ERROR	Not executable because the rBLE mode is other than RBLE_MODE_ACTIVE.

3.2.2 RBLE_PASP_Server_Disable

RBLE_STATUS RBLE_PASP_Server_Disable(uint16_t conhdl)		
This function disables the PASP Server role. The result is reported by using the Server role disable completion event RBLE_PASP_EVENT_SERVER_DISABLE_COMP.		
Parameters:		
conhdl	Connection handle	
Return:		
RBLE_OK	Success	
RBLE_STATUS_ERROR	Not executable because the rBLE mode is other than RBLE_MODE_ACTIVE.	

3.2.3 RBLE_PASP_Server_Send_Alert_Status

RBLE_STATUS RBLE_PASP_Server_Send_Alert_Status(uint16_t conhdl, uint8_t alert_status)	
This function sends the alert status value data from the server. The result is reported by using the Server role alert status value send completion event RBLE_PASP_EVENT_SERVER_SEND_ALERT_STATUS_COMP.	
Parameters:	
conhdl	Connection handle
alert_status	Alert Status ・RBLE_PASP_RINGER_STATE_BIT (bit[0]) →Ringer State (0：not active 1：active) ・RBLE_PASP_VIBRATOR_STATE_BIT (bit[1]) →Vibrator State (0：not active 1：active) ・RBLE_PASP_DISPLAY_ALERT_STATE_BIT (bit[2]) →Display Alert Status (0：not active 1：active)
Return:	
RBLE_OK	Success
RBLE_STATUS_ERROR	Not executable because the rBLE mode is other than RBLE_MODE_ACTIVE.

3.2.4 RBLE_PASP_Server_Send_Ringer_Setting

RBLE_STATUS RBLE_PASP_Server_Send_Ringer_Setting(uint16_t conhdl, ringer_setting)		
This function sends the ringer setting value data from the server. The result is reported by using the Server role ringer setting value send completion event RBLE_PASP_EVENT_SERVER_SEND_RINGER_SETTING_COMP.		
Parameters:		
conhdl	Connection handle	
ringer_setting	RBLE_PASP_RINGER_SILENT	Ringer Silent
	RBLE_PASP_RINGER_NORMAL	Ringer Normal
Return:		
RBLE_OK	Success	
RBLE_STATUS_ERROR	Not executable because the rBLE mode is other than RBLE_MODE_ACTIVE.	

3.2.5 RBLE_PASP_Client_Enable

RBLE_STATUS RBLE_PASP_Client_Enable(uint16_t conhdl, uint8_t con_type,
RBLE_PASS_CONTENT *pass, RBLE_PASPC_EVENT_HANDLER call_back)

This function enables the PASP Client role and starts access to the service exposed by the PASP Server. The result is reported by using the Client role enable completion event RBLE_PASP_EVENT_CLIENT_ENABLE_COMP.
When starting access to the service exposed by a Server to be connected for the first time, set 0 to the parameters of the service to configure the connection and to discover the service for the Server. If the handle information about the discovered service is saved and is used when the Server is connected normally for a second or subsequent time, detecting the service is skipped, which enables a high-speed access to the service.
While the Client role is enabled, the service exposed by only one Server is accessible. To connect to more than one Server at the same time and access the services exposed by each Server, repeat enable(RBLE_PASP_Client_Enable) / disable(RBLE_PASP_Client_Disable) of the Client role in order to switch access to them. At that time, perform normal connection by using the connection handle (which was obtained when connecting to each Server) and the handle information (which was saved when starting access to the service for the first time) as parameters.

Parameters:

conhdl	Connection handle	
con_type	RBLE_PRF_CON_DISCOVERY	Configuration connection performed when connecting for the first time
	RBLE_PRF_CON_NORMAL	Normal connection performed when connecting for the second and subsequent times
*pass	*shdl*	Phone Alert Status service start handle
	ehdl	Phone Alert Status service end handle
	alert_status_char_hdl	Alert Status characteristic handle
	alert_status_val_hdl	Alert Status characteristic value handle
	alert_status_cfg_hdl	Alert Status characteristic configuration descriptor handle
	alert_status_prop	Alert Status characteristic property
	ringer_setting_char_hdl	Ringer Setting characteristic handle
	ringer_setting_val_hdl	Ringer Setting characteristic value handle
	ringer_setting_cfg_hdl	Ringer Setting characteristic configuration descriptor handle
	ringer_setting_prop	Ringer Setting characteristic property
	ringer_cp_char_hdl	Ringer Control Point characteristic handle
	ringer_cp_val_hdl	Ringer Control Point characteristic value handle
	ringer_cp_prop	Ringer Control Point characteristic property
call_back	Callback	

Return:

RBLE_OK	Success
RBLE_ERR	Error occurred in initialization processing
RBLE_PARAM_ERR	Invalid parameter
RBLE_STATUS_ERROR	Not executable because the rBLE mode is other than RBLE_MODE_ACTIVE.

3.2.6 RBLE_PASP_Client_Disable

RBLE_STATUS RBLE_PASP_Client_Disable(uint16_t conhdl)		
This function disables the PASP Client role and terminates the access to the service exposed by PASP Server. The result is reported by using the Client role disable completion event RBLE_PASP_EVENT_CLIENT_DISABLE_COMP.		
Parameters:		
conhdl	Connection handle	
Return:		
RBLE_OK	Success	
RBLE_STATUS_ERROR	Not executable because the rBLE mode is other than RBLE_MODE_ACTIVE.	

3.2.7 RBLE_PASP_Client_Read_Char

RBLE_STATUS RBLE_PASP_Client_Read_Char (uint16_t conhdl, uint8_t char_code)		
This function reads the characteristic value of the phone alert status service. The result is reported by using the characteristic value read request response event RBLE_PASP_EVENT_CLIENT_READ_CHAR_RESPONSE.		
Parameters:		
conhdl	Connection handle	
char_code	RBLE_PASPC_RD_PASS_ALERT_STATUS	Alert Status
	RBLE_PASPC_RD_PASS_ALERT_STATUS_CFG	Alert Status Notification
	RBLE_PASPC_RD_PASS_RINGER_SETTING	Ringer Setting
	RBLE_PASPC_RD_PASS_RINGER_SETTING_CFG	Ringer Setting Notification
Return:		
RBLE_OK	Success	
RBLE_STATUS_ERROR	Not executable because the rBLE mode is other than RBLE_MODE_ACTIVE.	

3.2.8 RBLE_PASP_Client_Write_Ringer_Control_Point

RBLE_STATUS RBLE_PASP_Client_Write_Ringer_Control_Point (uint16_t conhdl, uint8_t cp_val)		
This function sets the Ringer control point characteristic information of the phone alert status service. The result is reported by using the characteristic value write request response event RBLE_PASP_EVENT_CLIENT_WRITE_CHAR_RESPONSE.		
Parameters:		
conhdl	Connection handle	
cp_val	RBLE_PASP_SILENT_MODE	Silent Mode
	RBLE_PASP_MUTE_ONCE	Mute Once
	RBLE_PASP_CANCEL_SILENT_MODE	Cancel Silent Mode
Return:		
RBLE_OK	Success	
RBLE_STATUS_ERROR	Not executable because the rBLE mode is other than RBLE_MODE_ACTIVE.	

3.2.9 RBLE_PASP_Client_Write_Char

RBLE_STATUS RBLE_PASP_Client_Write_Char(uint16_t conhdl, uint8_t char_code, uint16_t cfg_val)			
This function writes each client characteristic configuration descriptor of the phone alert service. The result is reported by using the characteristic value write request response event RBLE_PASP_EVENT_CLIENT_WRITE_CHAR_RESPONSE.			
Parameters:			
	conhdl	Connection handle	
	char_code	RBLE_PASP_ALERT_STATUS_CODE	alert status client characteristic configuration descriptor
		RBLE_PASP_RINGER_SETTING_CODE	ringer setting client characteristic configuration descriptor
	cfg_val	RBLE_PRF_STOP_NTFIND	Stop notification
		RBLE_PRF_START_NTF	Start notification
Return:			
	RBLE_OK	Success	
	RBLE_STATUS_ERROR	Not executable because the rBLE mode is other than RBLE_MODE_ACTIVE.	

3.3 Events

The following table shows the events defined for the PASP of rBLE and the following sections describe the events in detail.

Table 3-2 Events Defined for the PASP

RBLE_PASP_EVENT_SERVER_ENABLE_COMP	Server role enable completion event
RBLE_PASP_EVENT_SERVER_DISABLE_COMP	Server role disable completion event
RBLE_PASP_EVENT_SERVER_ERROR_IND	Server role error indication event
RBLE_PASP_EVENT_SERVER_SEND_ALERT_STATUS_COMP	Alert Status information send completion event
RBLE_PASP_EVENT_SERVER_SEND_RINGER_SETTING_COMP	Ringer Setting information send completion event
RBLE_PASP_EVENT_SERVER_CHG_RINGER_CP_IND	Ringer control point change indication event
RBLE_PASP_EVENT_SERVER_CFG_NTF_IND	Characteristic value indication event
RBLE_PASP_EVENT_SERVER_COMMAND_DISALLOWED_IND	Server role command disallowed indication event
RBLE_PASP_EVENT_CLIENT_ENABLE_COMP	Client role enable completion event
RBLE_PASP_EVENT_CLIENT_DISABLE_COMP	Client role disable completion event
RBLE_PASP_EVENT_CLIENT_ERROR_IND	Client role error indication event
RBLE_PASP_EVENT_CLIENT_ALERT_STATUS_NTF	Alert Status notification event
RBLE_PASP_EVENT_CLIENT_RINGER_SETTING_NTF	Ringer Setting notification event
RBLE_PASP_EVENT_CLIENT_READ_CHAR_RESPONSE	Characteristic value read request response event
RBLE_PASP_EVENT_CLIENT_WRITE_CHAR_RESPONSE	Characteristic value write request response event
RBLE_PASP_EVENT_CLIENT_COMMAND_DISALLOWED_IND	Client role command disallowed indication event

3.3.1 RBLE_PASP_EVENT_SERVER_ENABLE_COMP

RBLE_PASP_EVENT_SERVER_ENABLE_COMP		
This event reports the result of enabling the Server role (RBLE_PASP_Server_Enable).		
Parameters:		
	status	Result of enabling the Server role (See *2.2* and *Bluetooth Low Energy Protocol Stack API Reference Manual: Basics, 3.2, Declaration of enumerated type for rBLE status*.)
	conhdl	Connection handle

3.3.2 RBLE_PASP_EVENT_SERVER_DISABLE_COMP

RBLE_PASP_EVENT_SERVER_DISABLE_COMP				
This event reports the result of disabling the Server role (RBLE_PASP_Server_Disable).				
Parameters:				
	conhdl		Connection handle	
	server_info	*alert_status_ntf_en*	RBLE_PRF_STOP_NTFIND	Stop notification of alert status information
			RBLE_PRF_START_NTF	Start notification of alert status information
		ringer_setting_ntf_en	RBLE_PRF_STOP_NTFIND	Stop notification of ringer setting information
			RBLE_PRF_START_NTF	Start notification of ringer setting information

3.3.3 RBLE_PASP_EVENT_SERVER_ERROR_IND

RBLE_PASP_EVENT_SERVER_ERROR_IND		
This event indicates an error code unique to the Server role.		
Parameters:		
	conhdl	Connection handle
	status	Error code. (See *2.2* and *Bluetooth Low Energy Protocol Stack API Reference Manual: Basics, 3.2, Declaration of enumerated type for rBLE status*.)

3.3.4 RBLE_PASP_EVENT_SERVER_SEND_ALERT_STATUS_COMP

RBLE_PASP_EVENT_SERVER_SEND_ALERT_STATUS_COMP		
This event reports completion of sending the alert status value (RBLE_PASP_Server_Send_Alert_Status).		
Parameters:		
	conhdl	Connection handle
	status	Alert Status value send completion result. (See *2.2* and *Bluetooth Low Energy Protocol Stack API Reference Manual: Basics, 3.2, Declaration of enumerated type for rBLE status*.)

3.3.5 RBLE_PASP_EVENT_SERVER_SEND_RINGER_SETTING_COMP

RBLE_PASP_EVENT_SERVER_SEND_RINGER_SETTING_COMP		
This event reports completion of sending the ringer setting status value (RBLE_PASP_Server_Send_Ringer_Setting).		
Parameters:		
conhdl	Connection handle	
status	Alert Status value send completion result. (See *2.2* and *Bluetooth Low Energy Protocol Stack API Reference Manual: Basics, 3.2, Declaration of enumerated type for rBLE status*.)	

3.3.6 RBLE_PASP_EVENT_SERVER_CHG_RINGER_NTF_CP_IND

RBLE_PASP_EVENT_SERVER_CHG_RINGER_NTF_CP_IND		
This event indicates that the value of the ringer control point characteristic of the phone alert service has been changed by the Client.		
Parameters:		
conhdl	Connection handle	
cp_val	RBLE_PASP_SILENT_MODE	Silent Mode
	RBLE_PASP_MUTE_ONCE	Mute Once
	RBLE_PASP_CANCEL_SLENT_MODE	Cancel Silent Mode

3.3.7 RBLE_PASP_EVENT_SERVER_CFG_NTF_IND

RBLE_PASP_EVENT_SERVER_CFG_NTF_IND		
This event indicates that the value of the client characteristic configuration descriptor of the alert status characteristic or ringer setting characteristic has been set by the Client.		
Parameters:		
conhdl	Connection handle	
char_code	RBLE_PASP_ALERT_STATUS_CODE	alert status client characteristic configuration descriptor
	RBLE_PASP_RINGER_SETTING_CODE	ringer setting client characteristic configuration descriptor
cfg_val	RBLE_PRF_STOP_NTFIND	Stop notification.
	RBLE_PRF_START_NTF	Start notification.

3.3.8 RBLE_PASP_EVENT_SERVER_COMMAND_DISALLOWED_IND

RBLE_PASP_EVENT_SERVER_COMMAND_DISALLOWED_IND			
This event indicates the error that occurs when a command executed by the Server role cannot be accepted.			
Parameters:			
	status	Result of command execution. (See *2.2* and *Bluetooth Low Energy Protocol Stack API Reference Manual: Basics, 3.2, Declaration of enumerated type for rBLE status*.)	
	opcode	RBLE_CMD_PASP_SERVER_ENABLE	Server role enable command
		RBLE_CMD_PASP_SERVER_DISABLE	Server role disable command
		RBLE_CMD_PASP_SERVER_SEND_ALERT_STATUS	alert status information send command
		RBLE_CMD_PASP_SERVER_SEND_RINGER_SETTING	ringer setting information send command

3.3.9 RBLE_PASP_EVENT_CLIENT_ENABLE_COMP

RBLE_PASP_EVENT_CLIENT_ENABLE_COMP			
This event reports the result of enabling the Client role (RBLE_PASP_Client_Enable). Save the obtained handle information about the discovered service, to enable a high-speed access to the service without service detection when restarting access to the service.			
Parameters:			
	status	Result of command execution. (See *2.2* and *Bluetooth Low Energy Protocol Stack API Reference Manual: Basics, 3.2, Declaration of enumerated type for rBLE status*.)	
	conhdl	Connection handle	
	pass	*shdl*	Phone Alert Status service start handle
		ehdl	Phone Alert Status service end handle
		alert_status_char_hdl	Alert Status characteristic handle
		alert_status_val_hdl	Alert Status characteristic value handle
		alert_status_cfg_hdl	Alert Status characteristic configuration descriptor handle
		alert_status_prop	Alert Status characteristic property
		ringer_setting_char_hdl	Ringer Setting characteristic handle
		ringer_setting_val_hdl	Ringer Setting characteristic value handle
		ringer_setting_cfg_hdl	Ringer Setting characteristic configuration descriptor handle
		ringer_setting_prop	Ringer Setting characteristic property
		ringer_cp_char_hdl	Ringer Control Point characteristic handle
		ringer_cp_val_hdl	Ringer Control Point characteristic value handle
		ringer_cp_prop	Ringer Control Point characteristic property

3.3.10 RBLE_PASP_EVENT_CLIENT_DISABLE_COMP

RBLE_PASP_EVENT_CLIENT_DISABLE_COMP		
This event reports the result of disabling the Client role (RBLE_PASP_Client_Disable).		
Parameters:		
	status	Result of disabling the Client role. (See *2.2* and *Bluetooth Low Energy Protocol Stack API Reference Manual: Basics, 3.2, Declaration of enumerated type for rBLE status*.)
	conhdl	Connection handle

3.3.11 RBLE_PASP_EVENT_CLIENT_ERROR_IND

RBLE_PASP_EVENT_CLIENT_ERROR_IND		
This event indicates an error code unique to the PASP Client role.		
Parameters:		
	status	Error code. (See *2.2* and *Bluetooth Low Energy Protocol Stack API Reference Manual: Basics, 3.2, Declaration of enumerated type for rBLE status*.)
	conhdl	Connection handle

3.3.12 RBLE_PASP_EVENT_CLIENT_ALERT_STATUS_NTF

RBLE_PASP_EVENT_CLIENT_ALERT_STATUS_NTF		
This event indicates the alert status value sent from the Server.		
Parameters:		
	conhdl	Connection handle
	alert_status	Alert Status ・RBLE_PASP_RINGER_STATE_BIT (bit[0]) →Ringer State (0：not active 1：active) ・RBLE_PASP_VIBRATOR_STATE_BIT (bit[1]) →Vibrator State (0：not active 1：active) ・RBLE_PASP_DISPLAY_ALERT_STATE_BIT (bit[2]) →Display Alert Status (0：not active 1：active)

3.3.13 RBLE_PASP_EVENT_CLIENT_RINGER_SETTING_NTF

RBLE_PASP_EVENT_CLIENT_RINGER_SETTING_NTF			
This event indicates the ringer setting value sent from the Server.			
Parameters:			
	conhdl	Connection handle	
	ringer_setting	RBLE_PASP_RINGER_SILENT	Ringer Silent
		RBLE_PASP_RINGER_NORMAL	Ringer Normal

3.3.14 RBLE_PASP_EVENT_CLIENT_READ_CHAR_RESPONSE

RBLE_PASP_EVENT_CLIENT_READ_CHAR_RESPONSE
This event reports the response to the characteristic value read request (RBLE_PASP_Client_Read_Char). Read out the read data in accordance with the contents of the request.

・RBLE_PASPC_RD_PASS_ALERT_STATUS

LSB

Octet0	Octet1	Octet2	Octet3	Octet4	Octet5
Alert Status	-	-	-	-	-

MSB

・RBLE_PASPC_RD_PASS_RINGER_SETTING

LSB

Octet0	Octet1	Octet2	Octet3	Octet4	Octet5
Ringer Setting	-	-	-	-	-

MSB

・RBLE_PASPC_RD_PASS_ALERT_STATUS_CFG

・RBLE_PASPC_RD_PASS_RINGER_SETTING_CFG

LSB

Octet0	Octet1	Octet2	Octet3	Octet4	Octet5
client configuration (lower)	*client configuration (upper)*	-	-	-	-

MSB

Parameters:

conhdl	Connection handle	
att_code	0x00	Characteristic value successfully acquired
	Other than 0x00	Error occurred when acquiring characteristic value
data	*each_len*	Length of each result
	len	Data length
	data[RBLE_ATTM_MAX_VALUE]	Read characteristic data

3.3.15 RBLE_PASP_EVENT_CLIENT_WRITE_CHAR_RESPONSE

RBLE_PASP_EVENT_CLIENT_WRITE_CHAR_RESPONSE		
This event reports the response to the characteristic value write request (RBLE_PASP_Client_Write_Char).		
Parameters:		
conhdl	Connection handle	
att_code	0x00	Characteristic value successfully written
	Other than 0x00	Error occurred when writing characteristic value

3.3.16 RBLE_PASP_EVENT_CLIENT_COMMAND_DISALLOWED_IND

RBLE_PASP_EVENT_CLIENT_COMMAND_DISALLOWED_IND			
This event indicates the error that occurs when a command executed by the Client role cannot be accepted.			
Parameters:			
	status	Result of command execution (See *2.2* and *Bluetooth Low Energy Protocol Stack API Reference Manual: Basics, 3.2, Declaration of enumerated type for rBLE status*.)	
	opcode	RBLE_CMD_PASP_CLIENT_ENABLE	Client role enable command
		RBLE_CMD_PASP_CLIENT_DISABLE	Client role disable command
		RBLE_CMD_PASP_CLIENT_READ_CHAR	Characteristic read command
		RBLE_CMD_PASP_CLIENT_WRITE_RINGER_CP	Ringer control point write command
		RBLE_CMD_PASP_CLIENT_WRITE_CHAR	Characteristic write command

3.4 Message Sequence Chart

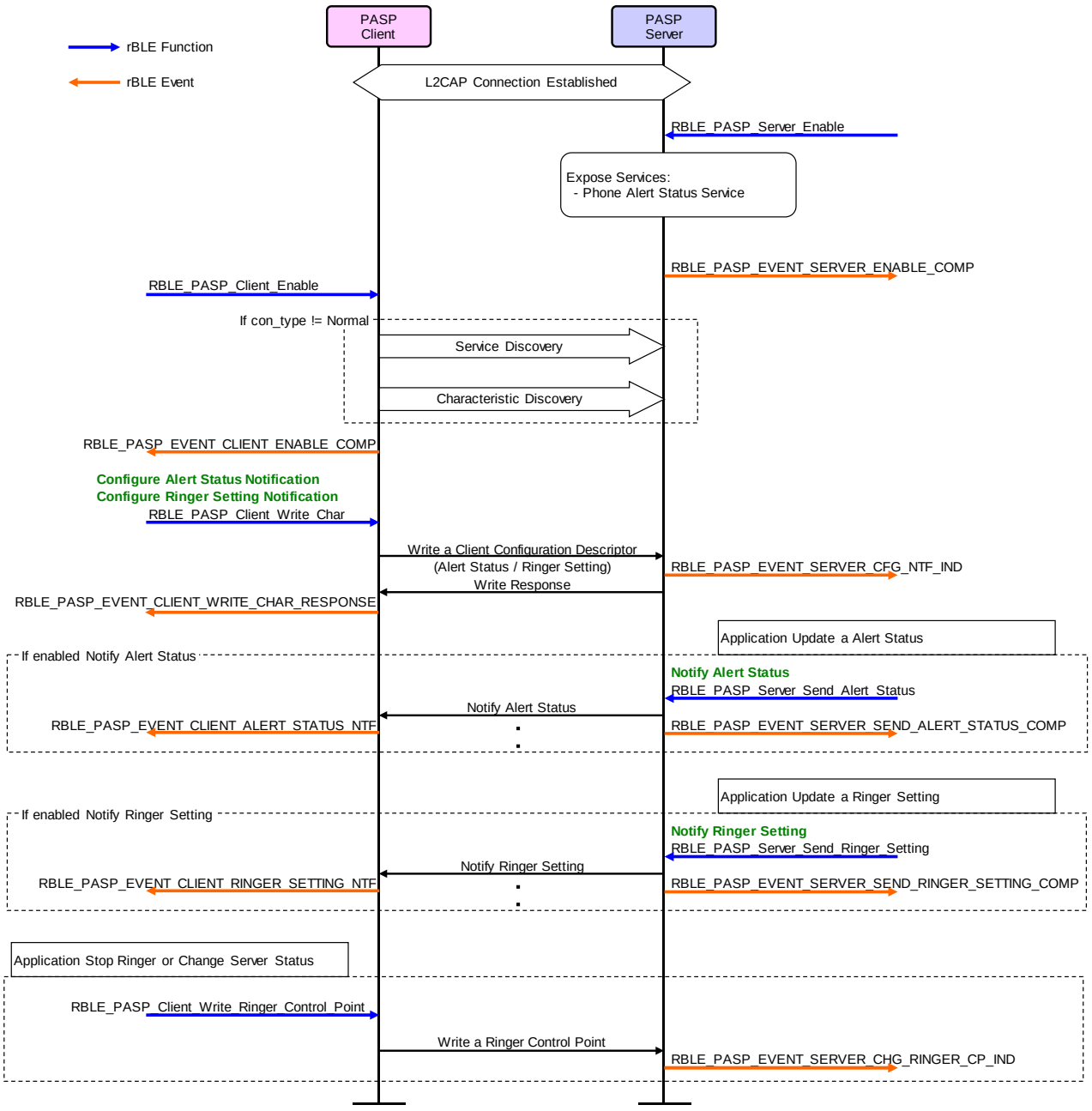

Figure 3-1 example of use case realization of PASP by using rBLE API

4. Notes

Appendix A How to Read Definition Tables

This section shows how to read the tables that describes the rBLE API functions and events shown in this document.

A.1 How to Read Function Definition Tables

The following contents are included in the function definition tables:

The Parameters area describes the parameters specified for the function.
The italicized character strings on the left are the parameters of the function.
The meaning of each parameter is described on the far right following the variables.

The italicized character string(s) next to each parameter indicate the member(s) of the parameter (structure).

The values that can be specified for the parameter might be described between the parameter name and its description.

The function definition is shown at the top of the table in the row with the light green background. This area shows the function prototype.				
The operation of the function and the event reported after executing the function are described in this area.				
Parameters:				
	Parameter 1	Description of parameter 1		
	Parameter 2	*Member 1*	Value 1 that can be specified for member 1	Description of value 1 that can be specified for member 1
			Value 1 that can be specified for member 2	Description of value 1 that can be specified for member 2
		Member 2	Description of member 2	
Return:				
	Value 1 that might be returned	Description of value 1 that might be returned		
	Value 2 that might be returned	Description of value 2 that might be returned		

The Return area describes the values returned for the function.
The leftmost row shows the value that might be returned, and the next row describes the return value.

A.2 How to Read Event Definition Tables

The following contents are included in the event definition tables:

The Parameters area describes the parameters specified for the event.
The italicized character strings on the left show the parameters of the event parameter structure. The meaning of each parameter is described on the far right.

The italicized character string(s) next to each parameter indicate the member(s) of the parameter (structure).

The event definition is shown at the top of the table in the row with the orange background. This area shows the event type.			
The information reported by the event is described in this area.			
Parameters:			
	Parameter 1	Description of parameter 1	
	Parameter 2	*Member 1*	Description of member 1
		Member 2	Description of member 2
		Member 3	Description of member 3
	Parameter 3	Value 1 that can be specified for parameter 3	Description of value 1 that can be specified for parameter 3
		Value 2 that can be specified for parameter 3	Description of value 2 that can be specified for parameter 3

The values that can be specified for the parameter might be shown between the parameter name and its description.

Appendix B Referenced Documents

1. Bluetooth Core Specification v4.0, Bluetooth SIG
2. Find Me Profile Specification v1.0, Bluetooth SIG
3. Immediate Alert Service Specification v1.0, Bluetooth SIG
4. Proximity Profile Specification v1.0, Bluetooth SIG
5. Link Loss Service Specification v1.0, Bluetooth SIG
6. Tx Power Service Specification v1.0, Bluetooth SIG
7. Health Thermometer Profile Specification v1.0, Bluetooth SIG
8. Health Thermometer Service Specification v1.0, Bluetooth SIG
9. Device Information Service Specification v1.1, Bluetooth SIG
10. Blood Pressure Profile Specification v1.0, Bluetooth SIG
11. Blood Pressure Service Specification v1.0, Bluetooth SIG
12. HID over GATT Profile Specification v1.0, Bluetooth SIG
13. HID Service Specification v1.0, Bluetooth SIG
14. Battery Service Specification v1.0, Bluetooth SIG
15. Scan Parameters Profile Specification v1.0, Bluetooth SIG
16. Scan Parameters Service Specification v1.0, Bluetooth SIG
17. Heart Rate Profile Specification v1.0, Bluetooth SIG
18. Heart Rate Service Specification v1.0, Bluetooth SIG
19. Cycling Speed and Cadence Profile Specification v1.0, Bluetooth SIG
20. Cycling Speed and Cadence Service Specification v1.0, Bluetooth SIG
21. Cycling Power Profile Specification v0.9, Bluetooth SIG
22. Cycling Power Service Specification v0.9, Bluetooth SIG
23. Glucose Profile Specification v1.0, Bluetooth SIG
24. Glucose Service Specification v1.0, Bluetooth SIG
25. Time Profile Specification v1.0, Bluetooth SIG
26. Current Time Service Specification v1.0, Bluetooth SIG
27. Next DST Change Service Specification v1.0, Bluetooth SIG
28. Reference Time Update Service Specification v1.0, Bluetooth SIG
29. Alert Notification Service Specification v1.0, Bluetooth SIG
30. Alert Notification Profile Specification v1.0, Bluetooth SIG
31. Location and Navigation Service Specification v1.0, Bluetooth SIG
32. Location and Navigation Profile Specification v1.0, Bluetooth SIG
33. Phone Alert Status Service Specification v1.0, Bluetooth SIG
34. Phone Alert Status Profile Specification v1.0, Bluetooth SIG
35. Bluetooth SIG Assigned Numbers https://www.bluetooth.org/Technical/AssignedNumbers/home.htm
36. Services & Characteristics UUID http://developer.bluetooth.org/gatt/Pages/default.aspx
37. Personal Health Devices Transcoding White Paper v1.2, Bluetooth SIG

Appendix C Terminology

Term	Description
Service	A service is provided from a GATT server to a GATT client. The GATT server exposes some characteristics as the interface. The service prescribes how to access the exposed characteristics.
Profile	A profile enables implementation of a use case by using one or more services. The services used are defined in the specifications of each profile.
Characteristic	A characteristic is a value used to identify services. The characteristics to be exposed and their formats are defined by each service.
Role	Each device takes the role prescribed by the profile or service in order to implement the specified use case.
Client Characteristic Configuration Descriptor	A descriptor is used to control notifications or indications of characteristic values that include the client characteristic configuration descriptor sent from the GATT server.
Server Characteristic Configuration Descriptor	A descriptor is used to control broadcast of characteristic values that include the server characteristic configuration descriptor sent from the GATT server.
Connection Handle	The handle determined by the controller stack and is used to identify connection with a remote device. The valid handle range is between 0x0000 and 0x0EFF.

REVISION HISTORY	Bluetooth Low Energy Protocol Stack API Reference Manual: PASP

Rev.	Date	Description	
		Page	Summary
0.11	Jan 30, 2015	---	Provisional Edition issued
1.00	Apr 17, 2015	2	The service definitions are updated.
1.01	Oct 30, 2015	5	Changed the enumerated value for ringer mode.

Bluetooth Low Energy Protocol Stack
API Reference Manual: PASP

Publication Date: Rev.1.01 Oct 30, 2015

Published by: Renesas Electronics Corporation

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HALII Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 4.0

Bluetooth Low Energy Protocol Stack

Renesas Electronics Corporation

R01UW0109EJ0101