Phone Alert Status Service (PASS)

Application Programming Interface Reference Manual

Profile Version: 1.0

Release: 4.0.1
January 10, 2013

Louisville, KY www.stonestreetone.com

Table of Contents

1. Introduction

Bluetopia®+LE is Stonestreet One's Bluetooth protocol stack that supports the adopted Bluetooth low energy specification. Stonestreet One's upper level protocol stack that supports Single Mode devices is Bluetopia®+LE Single. More specifically, this stack is a software solution that resides above the Physical HCI (Host Controller Interface) Transport Layer and extends through the L2CAP (Logical Link Control and Adaptation Protocol), ATT (Attribute Protocol) Link Layers, the GAP (Generic Attribute Profile) Layer and the Genetic Attribute Protocol (GATT) Layer. In addition to basic functionality of these layers, the Bluetooth Protocol Stack by Stonestreet One provides implementations of the Device Information Service (DIS), PASS (Phone Alert Status Service), and several of the Bluetooth Profiles. Program access to these layers, services, and profiles is handled via Application Programming Interface (API) calls.

The remainder of this chapter has sections on the scope of this document, other documents applicable to this document, and a listing of acronyms and abbreviations. Chapter 2 is the API reference that contains a description of all programming interfaces for the Phone Alert Status Service Profile Stack provided by Bluetopia®+LE Single. And, Chapter 3 contains the header file name list for the Phone Alert Status Service library.

1.1 Scope

This reference manual provides information on the PASS API. This API is available on the full range of platforms supported by Stonestreet One:

- Windows
- Windows Mobile
- Windows CE
- Linux
- QNX
- Other Embedded OS

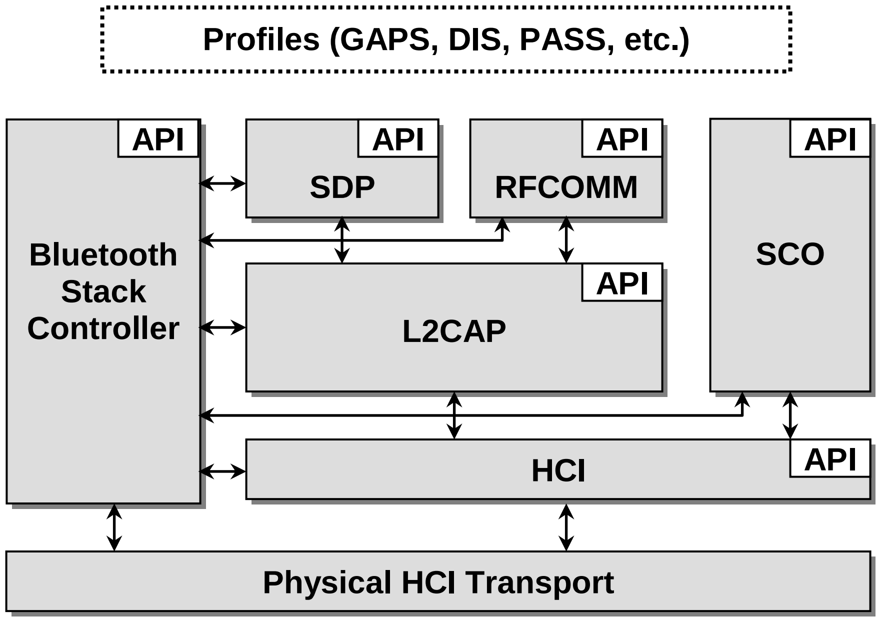

Figure 1-1 The Stonestreet One Bluetooth Protocol Stack

1.2 Applicable Documents

The following documents may be used for additional background and technical depth regarding the Bluetooth technology.

1. *Specification of the Bluetooth System, Volume 1, Architecture and Terminology Overview*, version 4.0, June 30, 2010.
2. *Specification of the Bluetooth System, Volume 6, Core System Package [Low Energy Controller Volume]*, version 4.0, June 30, 2010.
3. *Bluetopia® Protocol Stack, Application Programming Interface Reference Manual*, version 4.0.1, January 10, 2013.
4. *Bluetooth Phone Alert Status Service Specification*, version v10r00, April 3, 2012.

Possible error returns are listed for each API function call. These are the *most likely* errors, but in fact programmers should allow for the possibility of any error listed in the BTErrors.h header file to occur as the value of a function return.

1.3 Acronyms and Abbreviations

Acronyms and abbreviations used in this document and other Bluetooth specifications are listed in the table below.

Term	Meaning
API	Application Programming Interface
ATT	Attribute Protocol
BD_ADDR	Bluetooth Device Address
BT	Bluetooth
GAPS	Generic Access Profile Service
GATT	Generic Attribute Protocol
HCI	Host Controller Interface
HS	High Speed
L2CAP	Logical Link Control and Adaptation Protocol
LE	Low Energy
LSB	Least Significant Bit
MSB	Most Significant Bit
PASS	Phone Alert Status Service

2. PASS Programming Interface

The Phone Alert Status Service, PASS, programming interface defines the protocols and procedures to be used to implement PASS capabilities for both Server and Client services. The PASS commands are listed in section 2.1, the event callback prototypes are described in section 2.2, the PASS events are itemized in section 2.3. The actual prototypes and constants outlines in this section can be found in the **PASSAPI.h** header file in the Bluetopia distribution.

2.1 Phone Alert Status Service Commands

The available PASS command functions are listed in the table below and are described in the text that follows.

Server Commands	
Function	**Description**
PASS_Initialize_Service	Opens a PASS Server.
PASS _Cleanup_Service	Closes an opened PASS Server.
PASS_Set_Alert_Status	Sets the Alert Status characteristic on specified PASS instance
PASS_Query_Alert_Status	Gets the current Alert Status characteristic value on the specified PASS instance.
PASS_Set_Ringer_Setting	Sets the Ringer Setting characteristic on the specified PASS instance.
PASS_Query_Ringer_Settings	Gets the current Ringer Setting characteristic value on the specified PASS instance.
PASS_Read_Client_Configuration_Response	Responds to a Read Client Configuration Request.
PASS_Send_Notification	Sends a notification of a specified characteristic to a specified remote device.
PASS_Decode_Ringer_Setting	Parses a value received from a remote PASS Server, interpreting it as the Ringer Setting characteristic.
PASS_Decode_Alert_Status	Parses a value received from a remote PASS Server, interpreting it as the Alert Status characteristic.
PASS_Format_Ringer_Control_Command	Formats a Ringer Control Point command.

PASS_Initialize_Service

This function opens a PASS Server on a specified Bluetooth Stack.

Notes:

1. Only one PASS Server, per Bluetooth Stack ID, may be open at a time.
2. All Client Requests will be dispatched to the EventCallback function that is specified by the second parameter to this function.

Prototype:

int BTPSAPI **PASS_Initialize_Service**(unsigned int BluetoothStackID,
PASS_Event_Callback_t EventCallback,
unsigned long CallbackParameter, unsigned int*ServiceID);

Parameters:

BluetoothStackID[1]	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize.
EventCallback	Callback function that is registered to receive events that are associated with the specified service.
CallbackParameter	A user-defined parameter that will be passed back to the user in the callback function.
ServiceID	Unique GATT Service ID of the registered PASS service returned from GATT_Register_Service API.

Return:

Positive non-zero if successful. The return value will be the Service ID of PASS Server that was successfully opened on the specified Bluetooth Stack ID. This is the value that should be used in all subsequent function calls that require Instance ID.

Negative if an error occurred. Possible values are:

PASS_ERROR_INSUFFICIENT_RESOURCES
PASS_ERROR_SERVICE_ALREADY_REGISTERED
PASS_ERROR_INVALID_PARAMETER
BTGATT_ERROR_INVALID_SERVICE_TABLE_FORMAT
BTGATT_ERROR_INSUFFICIENT_RESOURCES
BTGATT_ERROR_INVALID_PARAMETER
BTGATT_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGATT_ERROR_NOT_INITIALIZED

Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

PASS_Cleanup_Service

This function is responsible for cleaning up and freeing all resources associated with a PASS Service Instance. After this function is called, no other PASS Service function can be called until after a successful call to the PASS_Initialize_Service() function is performed.

Prototype:

int BTPSAPI **PASS_Cleanup_Service**(unsigned int BluetoothStackID,
unsigned int InstanceID);

Parameters:

BluetoothStackID[1]	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize.
InstanceID	The Service Instance ID to close. This InstanceID was returned from the PASS_Initialize_Service().

Return:

Zero if successful.

Negative if an error occurred. Possible values are:

PASS_ERROR_INVALID_PARAMETER
PASS_ERROR_INVALID_INSTANCE_ID

Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

PASS_Set_Alert_Status

This function is responsible for setting the Alert Status characteristic on the specified PASS instance.

Prototype:

int BTPSAPI **PASS_Set_Alert_Status**(unsigned int BluetoothStackID, unsigned int
InstanceID, PASS_Alert_Status_t AlertStatus);

Parameters:

BluetoothStackID[1]	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize.
InstanceID	The Service Instance ID to close. This InstanceID was returned from the PASS_Initialize_Service().
AlertStatus	The Alert Status to set as the current Alert Status for the specified PASS Instance. The Alert Status structure is defined as follows:

```
typedef struct _tagPASS_Alert_Status_t
{
    Boolean_t       RingerStateActive;
    Boolean_t       VibrateStateActive;
    Boolean_t       DisplayStateActive;
} PASS_Alert_Status_t;
```

Return:

Zero if successful.

Negative if an error occurred. Possible values are:

PASS_ERROR_INVALID_INSTANCE_ID
PASS_ERROR_INVALID_PARAMETER
BTGATT_ERROR_NOT_INITIALIZED
BTGATT_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGATT_ERROR_INVALID_PARAMETER

Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

PASS_Query_Alert_Status

This function is responsible for querying the current Alert Status characteristic value on the specified PASS instance.

Prototype:

int BTPSAPI **PASS_Query_Alert_Status**(unsigned int BluetoothStackID, unsigned int InstanceID, PASS_Alert_Status_t *AlertStatus);

Parameters:

BluetoothStackID[1]	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize.
InstanceID	The Service Instance ID to close. This InstanceID was returned from the PASS_Initialize_Service().
AlertStatus	A pointer to an Alert Status structure to return the current Alert Status for the specified PASS Instance. The Alert Status structure is defined as follows:

```
typedef struct _tagPASS_Alert_Status_t
{
    Boolean_t       RingerStateActive;
    Boolean_t       VibrateStateActive;
    Boolean_t       DisplayStateActive;
} PASS_Alert_Status_t;
```

Return:

Zero if successful.

An error code if negative; one of the following values:

```
PASS_ERROR_INVALID_INSTANCE_ID
PASS_ERROR_INVALID_PARAMETER
BTGATT_ERROR_NOT_INITIALIZED
BTGATT_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGATT_ERROR_INVALID_PARAMETER
```

Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

PASS_Set_Ringer_Setting

The following function is responsible for setting the Ringer Setting characteristic on the specified PASS instance.

Prototype:

int BTPSAPI **PASS_Set_Ringer_Setting**(unsigned int BluetoothStackID, unsigned int InstanceID, PASS_Ringer_Setting_t RingerSetting);

Parameters:

BluetoothStackID[1]	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize.
InstanceID	The Service Instance ID to close. This InstanceID was returned from the PASS_Initialize_Service().
RingerSetting	Ringer Setting to set as the current Ringer Setting for the specified PASS instance. The Ringer Setting enum is defined as follows:

```
typedef enum
{
   rsSilent =
      PASS_RINGER_SETTING_RINGER_SILENT,
   rsNormal =
      PASS_RINGER_SETTING_RINGER_NORMAL
} PASS_Ringer_Setting_t;
```

Return:

Zero if successful.

Negative if an error occurred. Possible values are:

```
PASS_ERROR_INVALID_INSTANCE_ID
PASS_ERROR_INVALID_PARAMETER
```

BTGATT_ERROR_NOT_INITIALIZED
BTGATT_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGATT_ERROR_INVALID_PARAMETER

Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

PASS_Query_Ringer_Setting

The following function is responsible for querying the current Ringer Setting characteristic value on the specified PASS instance.

Prototype:

int BTPSAPI **PASS_Query_Ringer_Setting**(unsigned int BluetoothStackID, unsigned int InstanceID, PASS_Ringer_Setting_t *RingerSetting);

Parameters:

BluetoothStackID[1]	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize.
InstanceID	The Service Instance ID to close. This InstanceID was returned from the PASS_Initialize_Service().
RingerSetting	A pointer to store the current Ringer Setting for the specified PASS instance. The Ringer Setting enum is defined as follows:

```
typedef enum
{
    rsSilent =
        PASS_RINGER_SETTING_RINGER_SILENT,
    rsNormal =
        PASS_RINGER_SETTING_RINGER_NORMAL
} PASS_Ringer_Setting_t;
```

Return:

Zero if successful.

Negative if an error occurred. Possible values are:

PASS_ERROR_INVALID_INSTANCE_ID
PASS_ERROR_INVALID_PARAMETER
BTGATT_ERROR_NOT_INITIALIZED
BTGATT_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGATT_ERROR_INVALID_PARAMETER

Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

PASS_Read_Client_Configuration_Response

The following function is responsible for responding to a Read Client Configuration Request.

Prototype:

int BTPSAPI **PASS_Read_Client_Configuration_Response**(unsigned int BluetoothStackID, unsigned int InstanceID, unsigned int TransactionID, Boolean_t NotificationsEnabled);

Parameters:

BluetoothStackID[1]	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize.
InstanceID	The Service Instance ID to close. This InstanceID was returned from the PASS_Initialize_Service().
TransactionID	The TransactionID of the request.
NotificationsEnabled	Contains the client configuration to send to the remote device.

Return:

Zero if successful.

Negative if an error occurred. Possible values are:

PASS_ERROR_INVALID_INSTANCE_ID
PASS_ERROR_INVALID_PARAMETER
BTGATT_ERROR_NOT_INITIALIZED
BTGATT_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGATT_ERROR_INVALID_PARAMETER

Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

PASS_Send_Notification

The following function is responsible for sending a notification of a specified characteristic to a specified remote device.

Prototype:

int BTPSAPI **PASS_Send_Notification**(unsigned int BluetoothStackID, unsigned int InstanceID, unsigned int ConnectionID, PASS_Characteristic_Type_t CharacteristicType);

Parameters:

BluetoothStackID[1]	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize.
InstanceID	The Service Instance ID to close. This InstanceID was returned from the PASS _Initialize_Service().
ConnectionID	Connection ID of the currently connected remote client device to send the handle/value notification.
CharacteristicType	The characteristic to notify.The Characteristic Type enum is defined as follows:

```
typedef enum
{
    rrAlertStatus,
    rrRingerSetting
} PASS_Characteristic_Type_t;
```

Return:

Zero if successful.

Negative if an error occurred. Possible values are:

```
PASS_ERROR_INVALID_INSTANCE_ID
PASS_ERROR_INVALID_PARAMETER
PASS_ERROR_UNKNOWN_ERROR
BTGATT_ERROR_NOT_INITIALIZED
BTGATT_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGATT_ERROR_INVALID_PARAMETER
```

Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

PASS_Decode_Alert_Status

The following function is responsible for parsing a value received from a remote PASS Server, and interpreting it as the Alert Status characteristic.

Prototype:

int BTPSAPI **PASS_Decode_Alert_Status**(unsigned int ValueLength, Byte_t *Value, PASS_Alert_Status_t *AlertStatusResult);

Parameters:

ValueLength	The length of the value returned by the remote PASS Server
Value	Pointer to the data returned by the remote PASS Server.
AlertStatusResult	Pointer to store the parsed Alert Status value. The Alert Status structure is defined as follows:

```
typedef struct _tagPASS_Alert_Status_t
{
   Boolean_t     RingerStateActive;
   Boolean_t     VibrateStateActive;
   Boolean_t     DisplayStateActive;
} PASS_Alert_Status_t;
```

Return:

Zero if successful.

Negative if an error occurred.

PASS_Decode_Ringer_Setting

The following function is responsible for formatting a Ringer Control Point command.

Prototype:

int BTPSAPI **PASS_Decode_Ringer_Setting**(unsigned int ValueLength, Byte_t *Value,
   PASS_Ringer_Setting_t *RingerSetting);

Parameters:

ValueLength	The length of the value returned by the remote PASS Server
Value	Pointer to the data returned by the remote PASS Server.
RingerSetting	Pointer to store the parsed Ringer Setting value. The Ringer Setting enum is defined as follows:

```
typedef enum
{
   rsSilent =
      PASS_RINGER_SETTING_RINGER_SILENT,
   rsNormal =
      PASS_RINGER_SETTING_RINGER_NORMAL
} PASS_Ringer_Setting_t;
```

Return:

Zero if successful.

Negative if an error occurred.

PASS_Format_Ringer_Control_Command

The following function is responsible for formatting a Ringer Control Point command.

Prototype:

int BTPSAPI **PASS_Format_Ringer_Control_Command**(PASS_Ringer_Control_Command_t RingerControlCommand, unsigned int BufferLength, Byte_t *Buffer);

Parameters:

RingerControlCommand	The command to format. The Ringer Control Command enum is defined as follows:
	typedef enum
	{
	rcSilent = PASS_RINGER_CONTROL_COMMAND_SILENT_MODE,
	rcMuteOnce = ` PASS_RINGER_CONTROL_COMMAND_MUTE_ONCE,
	rcCancelSilent = PASS_RINGER_CONTROL_COMMAND_CANCEL_SILENT_MODE
	} **PASS_Ringer_Control_Command_t;**

BufferLength	The length of the buffer that will be user to hold the command
Buffer	Pointer to the buffer that will hold the formatted command.

Return:

Zero if successful.

Negative if an error occurred.

2.2 Phone Alert State Service Event Callback Prototypes

2.2.1 Server Event Callback

The event callback function mentioned in the PASS_Initialize_Service command accepts the callback function described by the following prototype.

PASS_Event_Callback_t

This The event callback function mentioned in the PASS_Initialize_Service command accepts the callback function described by the following prototype.

Prototype:

typedef void (BTPSAPI ***PASS_Event_Callback_t**)(unsigned int BluetoothStackID, PASS_Event_Data_t *PASS_Event_Data, unsigned long CallbackParameter);

Parameters:

BluetoothStackID[1]	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize.

PASS_Event_Data_t	Data describing the event for which the callback function is called. This is defined by the following structure:

```
typedef struct
{
    PASS_Event_Type_t    Event_Data_Type;
    Word_t               Event_Data_Size;
    union
    {
        PASS_Read_Client_Configuration_Data_t
            *PASS_Read_Client_Configuration_Data;
        PASS_Client_Configuration_Update_Data_t
            *PASS_Client_Configuration_Update_Data;
        PASS_Ringer_Control_Point_Command_Data_t
            *PASS_Ringer_Control_Point_Command_Data;
    } Event_Data;
} PASS_Event_Data_t;
```

Where, Event_Data_Type is one of the enumerations of the event types listed in the table in section 2.3, and each data structure in the union is described with its event in that section as well.

CallbackParameter	User-defined parameter that was defined in the callback registration.

Return:

XXX/None

Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

2.3 Phone Alert State Service Events

The Phone Alert State Service contains events that are received by the Server. The following sections detail those events.

2.3.1 Phone Alert State Service Server Events

The possible Phone Alert State Service Server Events from the Bluetooth stack are listed in the table below and are described in the text which follows:

Server Commands	
Function	**Description**
etPASS_Server_Read_Client_Configuration_Request	Dispatched to a PASS Server when a PASS Client is attempting to read a

	descriptor.
etPASS_Server_Client_Configuration_Update	Dispatched to a PASS Server when a PASS Client has written a Client Configuration descriptor.
etPASS_Server_Ringer_Control_Command_ Indication	Dispatched to a PASS Server when a PASS Client is attempting to write a command to the Ringer Control point characteristic.

etPASS_Server_Read_Client_Configuration_Request

Dispatched to a PASS Server when a PASS Client is attempting to read a descriptor.

Return Structure:

```
typedef struct _tagPASS_Read_Client_Configuration_Data_t
{
    unsigned int                  InstanceID;
    unsigned int                  ConnectionID;
    unsigned int                  TransactionID;
    GATT_Connection_Type_t        ConnectionType;
    BD_ADDR_t                     RemoteDevice;
    PASS_Characteristic_Type_t    ClientConfigurationType;
} PASS_Read_Client_Configuration_Data_t;
```

Event Parameters:

InstanceID	Identifies the Local Server Instance to which the Remote Client has connected.
ConnectionID	Connection ID of the currently connected remote PASS server device.
TransactionID	The TransactionID identifies the transaction between a client and server. This identifier should be used to respond to the current request.
ConnectionType	Identifies the type of remote Bluetooth device that is connected. Currently this value will be gctLE only.
RemoteDevice	Specifies the address of the Client Bluetooth device that has connected to the specified Server.
ClientConfigurationType	Holds the configuration type. Can be either rrAlertStatus, or rrRingerSetting.

etPASS_Server_Client_Configuration_Update

Dispatched to a PASS Server when a PASS Client has written a Client Configuration descriptor.

Return Structure:

```
typedef struct _tagPASS_Client_Configuration_Update_Data_t
{
    unsigned int                 InstanceID;
    unsigned int                 ConnectionID;
    GATT_Connection_Type_t       ConnectionType;
    BD_ADDR_t                    RemoteDevice;
    PASS_Characteristic_Type_t   ClientConfigurationType;
    Boolean_t                    NotificationEnabled;
} PASS_Client_Configuration_Update_Data_t;
```

Event Parameters:

InstanceID	Identifies the Local Server Instance to which the Remote Client has connected.
ConnectionID	Connection ID of the currently connected remote PASS server device.
ConnectionType	Identifies the type of remote Bluetooth device that is connected. Currently this value will be gctLE only.
RemoteDevice	Specifies the address of the Client Bluetooth device that has connected to the specified Server.
ClientConfigurationType	Specifies the clients configuration type. Can be either rrAlertStatus, or rrRingerSetting.
NotificationEnabled	A Boolean variable to indicate wether or not notifications are enabled.

etPASS_Server_Ringer_Control_Point_Command

Dispatched to a PASS server when a PASS Client is attempting to write a command to the Ringer Control point characteristic.

Return Structure:

```
typedef struct
{
    unsigned int                        InstanceID;
    unsigned int                        ConnectionID;
    unsigned int                        TransactionID;
    GATT_Connection_Type_t              ConnectionType;
    BD_ADDR_t                           RemoteDevice;
    PASS_Ringer_Control _Command_t      Command;
} PASS_Ringer_Control_Command_Data_t;
```

Event Parameters:

InstanceID	Identifies the Local Server Instance to which the Remote Client has connected.
ConnectionID	Connection ID of the currently connected remote PASS server device.

TransactionID	The TransactionID identifies the transaction between a client and server. This identifier should be used to respond to the current request.
ConnectionType	Identifies the type of remote Bluetooth device that is connected. Currently this value will be gctLE only.
RemoteDevice	Specifies the address of the Client Bluetooth device that has connected to the specified Server.
Command	Specifies the command that the client is attempting to write. The Ringer Control Command enum is defined as follows:

```
typedef enum
{
  rcSilent      =
    PASS_RINGER_CONTROL_COMMAND_SILENT_
    MODE,
  rcMuteOnce    =
    PASS_RINGER_CONTROL_COMMAND_MUTE_ONCE,
  rcCancelSilent =
    PASS_RINGER_CONTROL_COMMAND_CANCEL_
    SILENT_MODE
} PASS_Ringer_Control_Command_t;
```

3. File Distributions

The header files that are distributed with the Bluetooth Phone Alert State Service Library are listed in the table below

File	Contents/Description
PASSAPI.h	Bluetooth Phone Alert State Service (GATT based) API Type Definitions, Constants, and Prototypes.
PASSTYPES.h	Bluetooth Phone Alert State Service Types.
SS1BTPASS.h	Bluetooth Phone Alert State Service Include file