



Headset Profile (HDSET)

Application Programming Interface Reference Manual

Profile Version: 1.2

Release: 4.0.1
January 10, 2014



Bluetooth and the Bluetooth logos are trademarks owned by Bluetooth SIG, Inc., USA and licensed to Stonestreet One, LLC. Bluetopia®, Stonestreet One™, and the Stonestreet One logo are registered trademarks of Stonestreet One, LLC, Louisville, Kentucky, USA. All other trademarks are property of their respective owners.
Copyright © 2000-2014 by Stonestreet One, LLC. All rights reserved.

Table of Contents

| | | |
|-----------|---|------------------|
| 1. | <u>INTRODUCTION.....</u> | <u>4</u> |
| 1.1 | Scope | 4 |
| 1.2 | Applicable Documents | 5 |
| 1.3 | Acronyms and Abbreviations | 6 |
| 2. | <u>HEADSET PROFILE PROGRAMMING INTERFACE</u> | <u>8</u> |
| 2.1 | Headset Profile Commands..... | 8 |
| | HDSET_Open_Headset_Server_Port | 9 |
| | HDSET_Open_Audio_Gateway_Server_Port..... | 10 |
| | HDSET_Close_Server_Port | 11 |
| | HDSET_Open_Port_Request_Response | 12 |
| | HDSET_Register_Headset_SDP_Record..... | 12 |
| | HDSET_Register_Audio_Gateway_SDP_Record | 14 |
| | HDSET_Open_Remote_Headset_Port | 15 |
| | HDSET_Open_Remote_Audio_Gateway_Port..... | 16 |
| | HDSET_Ring_Indication..... | 17 |
| | HDSET_Send_Button_Press | 17 |
| | HDSET_Accept_Incoming_Call | 18 |
| | HDSET_End_Call | 19 |
| | HDSET_Close_Port..... | 20 |
| | HDSET_Set_Speaker_Gain..... | 21 |
| | HDSET_Set_Microphone_Gain | 22 |
| | HDSET_Setup_Audio_Connection | 22 |
| | HDSET_Release_Audio_Connection..... | 23 |
| | HDSET_Send_Audio_Data | 24 |
| | HDSET_Get_Server_Mode ¹ | 25 |
| | HDSET_Set_Server_Mode ¹ | 26 |
| 2.2 | Headset Profile Event Callback Prototypes..... | 27 |
| | HDSET_Event_Callback_t | 27 |
| 2.3 | Headset Profile Events..... | 28 |
| | etHDSET_Open_Port_Request_Indication | 28 |
| | etHDSET_Open_Port_Indication | 29 |
| | etHDSET_Open_Port_Confirmation | 29 |
| | etHDSET_Close_Port_Indication..... | 30 |
| | etHDSET_Ring_Indication..... | 30 |
| | etHDSET_Button_Pressed_Indication | 30 |
| | etHDSET_Speaker_Gain_Indication | 30 |
| | etHDSET_Microphone_Gain_Indication | 31 |
| | etHDSET_Audio_Connection_Indication | 31 |
| | etHDSET_Audio_Disconnection_Indication..... | 31 |
| | etHDSET_Audio_Data_Indication | 32 |
| | etHDSET_Audio_Transmit_Buffer_Empty_Indication | 32 |
| 3. | <u>FILE DISTRIBUTIONS.....</u> | <u>34</u> |

1. Introduction

Bluetopia®, the Bluetooth Protocol Stack by Stonestreet One, provides a software architecture that encapsulates the upper functionality of the Bluetooth Protocol Stack. 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) and the SCO (Synchronous Connection-Oriented) Link layers. In addition to basic functionality at these layers, the Bluetooth Protocol Stack by Stonestreet One provides implementations of the Service Discovery Protocol (SDP), RFCOMM (the Radio Frequency serial COMMunications port emulator), and several of the Bluetooth Profiles. Program access to these layers, services, and profiles is handled via Application Programming Interface (API) calls.

This document focuses on the API reference that contains a description of all programming interfaces for the Bluetooth Headset Profile provided by Bluetopia. Chapter 2 contains a description of the programming interface for this profile. And, Chapter 3 contains the header file name list for the Bluetooth Headset Profile library.

1.1 Scope

This reference manual provides information on the Headset Profile API identified in Figure 1-1 below. These APIs are available on the full range of platforms supported by Stonestreet One:

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

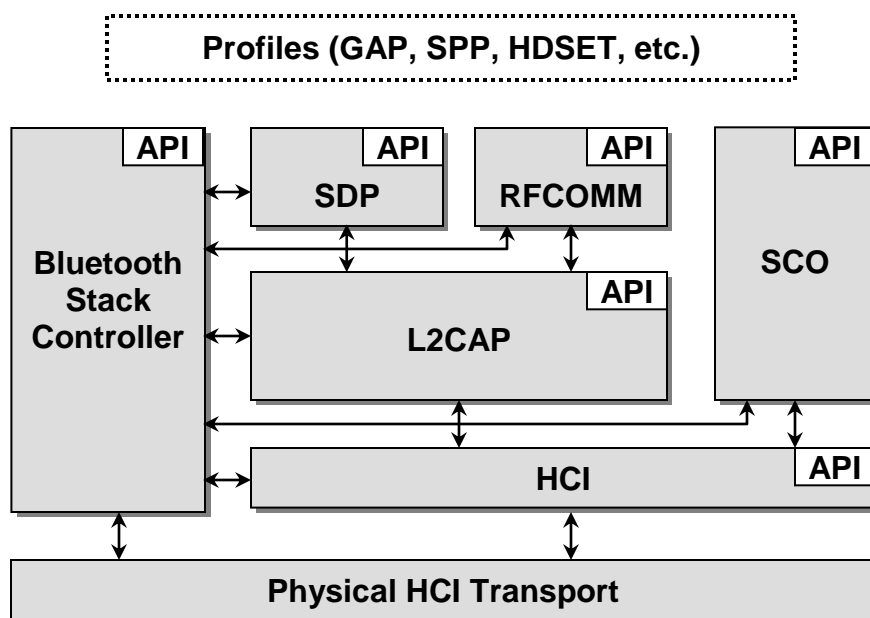


Figure 1-1 The Stonestreet One Bluetooth Protocol Stack

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, Core*, version 1.1, February 22, 2001.
2. *Specification of the Bluetooth System, Volume 2, Profiles*, version 1.1, February 22, 2001.
3. *Specification of the Bluetooth System, Volume 1, Architecture and Terminology Overview*, version 2.0 + EDR, November 4, 2004.
4. *Specification of the Bluetooth System, Volume 2, Core System Package*, version 2.0 + EDR, November 4, 2004.
5. *Specification of the Bluetooth System, Volume 3, Core System Package*, version 2.0 + EDR, November 4, 2004.
6. *Specification of the Bluetooth System, Volume 0, Master Table of Contents & Compliance Requirements*, version 2.1+EDR, July 26, 2007.
7. *Specification of the Bluetooth System, Volume 1, Architecture and Terminology Overview*, version 2.1+EDR, July 26, 2007.
8. *Specification of the Bluetooth System, Volume 2, Core System Package [Controller Volume]*, version 2.1+EDR, July 26, 2007.
9. *Specification of the Bluetooth System, Volume 3, Core System Package [Host Volume]*, version 2.1+EDR, July 26, 2007.
10. *Specification of the Bluetooth System, Volume 4, Host Controller Interface [Transport Layer]*, version 2.1+EDR, July 26, 2007.
11. *Specification of the Bluetooth System, Bluetooth Core Specification Addendum 1*, June 26, 2008.
12. *Specification of the Bluetooth System, Volume 0, Master Table of Contents & Compliance Requirements*, version 3.0+HS, April 21, 2009.
13. *Specification of the Bluetooth System, Volume 1, Architecture and Terminology Overview*, version 3.0+HS, April 21, 2009.
14. *Specification of the Bluetooth System, Volume 2, Core System Package [Controller Volume]*, version 3.0+HS, April 21, 2009.
15. *Specification of the Bluetooth System, Volume 3, Core System Package [Host Volume]*, version 3.0+HS, April 21, 2009.
16. *Specification of the Bluetooth System, Volume 4, Host Controller Interface [Transport Layer]*, version 3.0+HS, April 21, 2009.
17. *Specification of the Bluetooth System, Volume 5, Core System Package [AMP Controller Volume]*, version 3.0+HS, April 21, 2009.

18. *Specification of the Bluetooth System, Volume 0, Master Table of Contents & Compliance Requirements*, version 4.0, June 30, 2010.
19. *Specification of the Bluetooth System, Volume 1, Architecture and Terminology Overview*, version 4.0, June 30, 2010.
20. *Specification of the Bluetooth System, Volume 2, Core System Package [BR/EDR Controller Volume]*, version 4.0, June 30, 2010.
21. *Specification of the Bluetooth System, Volume 3, Core System Package [Host Volume]*, version 4.0, June 30, 2010.
22. *Specification of the Bluetooth System, Volume 4, Host Controller Interface [Transport Layer]*, version 4.0, June 30, 2010.
23. *Specification of the Bluetooth System, Volume 5, Core System Package [AMP Controller Volume]*, version 4.0, June 30, 2010.
24. *Specification of the Bluetooth System, Volume 6, Core System Package [Low Energy Controller Volume]*, version 4.0, June 30, 2010.
25. *Bluetooth Assigned Numbers*, version 1.1, February 22, 2001.
26. *Digital cellular telecommunications system (Phase 2+); Terminal Equipment to Mobile Station (TE-MS) multiplexer protocol (GSM 07.10)*, version 7.1.0, Release 1998; commonly referred to as: ETSI TS 07.10.
27. *Bluetopia® Protocol Stack, Application Programming Interface Reference Manual*, version 4.0.1, January 10, 2013.

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 |
| BD_ADDR | Bluetooth Device Address |
| BR | Basic Rate |
| BT | Bluetooth |
| EDR | Enhanced Data Rate |
| HS | High Speed |
| LE | Low Energy |
| LSB | Least Significant Bit |

| Term | Meaning |
|------|---|
| MSB | Most Significant Bit |
| SDP | Service Discovery Protocol |
| SPP | Serial Port Protocol |
| UART | Universal Asynchronous Receiver/Transmitter |
| USB | Universal Serial Bus |

2. Headset Profile Programming Interface

The Headset Profile programming interface defines the protocols and procedures to be used to implement headset capabilities. The Headset Profile commands are listed in section 2.1, the event callback prototype is described in section 2.2, and the Headset Profile events are itemized in section 2.3. The actual prototypes and constants outlined in this section can be found in the **HDSETAPI.H** header file in the Bluetopia distribution.

2.1 Headset Profile Commands

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

| Function | Description |
|---|---|
| HDSET_Open_Headset_Server_Port | Opens a Headset server on the specified Bluetooth SPP serial port. |
| HDSET_Open_Audio_Gateway_Server_Port | Opens an Audio Gateway server on the specified Bluetooth SPP serial port. |
| HDSET_Close_Server_Port | Un-Registers a HDSET Port server (which was registered by a successful call to either the HDSET_Open_Headset_Server_Port() or the HDSET_Open_Audio_Gateway_Server_Port() function). |
| HDSET_Open_Port_Request_Response | Responds to requests to connect to a Server (Head |
| HDSET_Register_Headset_SDP_Record | Adds a generic Headset Service Record to the SDP database. |
| HDSET_Register_Audio_Gateway_SDP_Record | Adds a generic Audio Gateway Service Record to the SDP database. |
| HDSET_Un_Register_SDP_Record | Removes a generic Headset Service Record or Audio Gateway Service Record from the SDP database. |
| HDSET_Open_Remote_Headset_Port | Opens a Remote Headset port on the specified remote device. |
| HDSET_Open_Remote_Audio_Gateway_Port | Opens a Remote Audio Gateway port on the specified remote device. |
| HDSET_Ring_Indication | Sends a Ring Indication to the remote side. |
| HDSET_Send_Button_Press | Sends a button press event from the Headset to the remote Audio Gateway |
| HDSET_Accept_Incoming_Call | Accepts or Rejects a incoming call request. This function has been superceded by the |

| | |
|--------------------------------|--|
| | HDSET_Send_Button_Press() function and should no longer be used. |
| HDSET_End_Call | Requests a Remote Audio Gateway to end the current connection. This function has been superceded by the HDSET_Send_Button_Press() function and should no longer be used. |
| HDSET_Close_Port | Closes a HDSET Port that was previously opened. |
| HDSET_Set_Speaker_Gain | Allows the local entity a mechanism of notifying the Remote entity (either Headset OR Audio Gateway) that the speaker gain has changed. |
| HDSET_Set_Microphone_Gain | Allows the local entity a mechanism of notifying the Remote entity (either Headset OR Audio Gateway) that the microphone gain has changed. |
| HDSET_Setup_Audio_Connection | Allows the local Audio Gateway the ability to establish the audio channel to the remote Headset (manually). |
| HDSET_Release_Audio_Connection | Allows the local Audio Gateway the ability to disconnect the audio channel from the remote Headset (manually). |
| HDSET_Send_Audio_Data | Allows a local entity to send SCO Audio Data to the Remote entity (either Headset OR Audio Gateway). |
| HDSET_Get_Server_Mode | Retrieves the current Headset/Audio Gateway Server Mode for a specified Headset/Audio Gateway server. |
| HDSET_Set_Server_Mode | Changes the Headset/Audio Gateway Server Mode for the specified Headset/Audio Gateway server. |

HDSET_Open_Headset_Server_Port

The following function is responsible for Opening a Headset Server on the specified Bluetooth SPP serial port.

Prototype:

```
int BTPSAPI HDSET_Open_Headset_Server_Port(unsigned int BluetoothStackID,
    unsigned int ServerPort, Boolean_t RemoteVolumeControlSupported,
    HDSET_Event_Callback_t EventCallback, unsigned long CallbackParameter)
```

Parameters:

BluetoothStackID¹ Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().

| | |
|-------------------------------|---|
| ServerPort | Local serial port Server Number to use. This must fall in the range defined by the following constants: SPP_PORT_NUMBER_MINIMUM SPP_PORT_NUMBER_MAXIMUM |
| RemoteVolumeControlsSupported | Boolean Flag that specifies whether or not the Headset supports Remote Audio Volume Controls (TRUE if supported). |
| EventCallback | Function to call when events occur on this port. |
| CallbackParameter | A user-defined parameter (e.g., a tag value) that will be passed back to the user in the callback function with each packet. |

Return:

A positive, non-zero, value if successful. A successful return code will be a HDSET Port ID that can be used to reference the Opened HDSET Port in ALL other functions in this module except for the HDSET_Register_Audio_Gateway_SDP_Record() function which is specific to an Audio Gateway Server NOT a Headset Server.

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

BTHDSET_ERROR_INSUFFICIENT_RESOURCES
BTHDSET_ERROR_NOT_INITIALIZED
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID
BTHDSET_ERROR_INVALID_PARAMETER

Possible Events:

etHDSET_Open_Port_Indication

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.

HDSET_Open_Audio_Gateway_Server_Port

The following function is responsible for Opening an Audio Gateway Server on the specified Bluetooth SPP serial port.

Prototype:

```
int BTPSAPI HDSET_Open_Audio_Gateway_Server_Port(
    unsigned int BluetoothStackID, unsigned int ServerPort,
    HDSET_Event_Callback_t EventCallback, unsigned long CallbackParameter)
```

Parameters:

| | |
|-------------------------------|---|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| ServerPort | Local serial port Server Number to use. This must fall in the range defined by the following constants: |

SPP_PORT_NUMBER_MINIMUM
SPP_PORT_NUMBER_MAXIMUM

| | |
|-------------------|--|
| EventCallback | Function to call when events occur on this port. |
| CallbackParameter | A user-defined parameter (e.g., a tag value) that will be passed back to the user in the callback function with each packet. |

Return:

This function returns a positive, non-zero, value if successful. A successful return code will be a HDSET Port ID that can be used to reference the Opened HDSET Port in ALL other functions in this module except for the HDSET_Register_Headset_SDP_Record() function which is specific to a Headset Server NOT an Audio Gateway.

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

BTHDSET_ERROR_INSUFFICIENT_RESOURCES
BTHDSET_ERROR_NOT_INITIALIZED
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID
BTHDSET_ERROR_INVALID_PARAMETER

Possible Events:

etHDSET_Open_Port_Indication

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.

HDSET_Close_Server_Port

This function is responsible for un-Registering a HDSET Port server (which was registered by a successful call to either the HDSET_Open_Headset_Server_Port() or the HDSET_Open_Audio_Gateway_Server_Port() function).

Prototype:

```
int BTPSAPI HDSET_Close_Server_Port(unsigned int BluetoothStackID,
                                     unsigned int HDSETPortID)
```

Parameters:

| | |
|-------------------------------|---|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| HDSETPortID | The Headset Port ID to close. This is the value that was returned from either HDSET_Open_Headset_Server_Port() or HDSET_Open_Audio_Gateway_Server_Port(). |

Return:

Zero if successful.

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

BTHDSET_ERROR_NOT_INITIALIZED
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID
BTHDSET_ERROR_INVALID_PARAMETER

Possible Events:

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.

HDSET_Open_Port_Request_Response

This function is responsible for responding to requests to connect to a Server, which can be either a Headset or Audio Gateway server.

Prototype:

```
int BTPSAPI HDSET_Open_Port_Request_Response(unsigned int BluetoothStackID,  
      unsigned int HDSETPortID, Boolean_t AcceptConnection)
```

Parameters:

| | |
|-------------------------------|--|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| HDSETPortID | Headset Port ID which must have been obtained by call either HDSET_Open_Headset_Server_Port() or the HDSET_Open_Audio_Gateway_Server_Port() functions. |
| AcceptConnection | Boolean specifying whether to accept the connection or reject it (TRUE to accept). |

Return:

Zero if successful.

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

BTHDSET_ERROR_NOT_INITIALIZED
BTHDSET_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.

HDSET_Register_Headset_SDP_Record

This function adds a generic Headset Service Record to the SDP database.

Notes:

1. This function should only be called with the HDSET Port ID that was returned from the HDSET_Open_Headset_Server_Port() function. This function should **never** be used with the HDSET Port ID returned from the HDSET_Open_Audio_Gateway_Server_Port() function.
2. The Service Record Handle that is returned from this function will remain in the SDP Record Database until it is deleted by calling the SDP_Delete_Service_Record() function. A Macro is provided to delete the Service Record from the SDP Database. This Macro maps HDSET_Un_Register_SDP_Record() to SDP_Delete_Service_Record(), and is defined as follows:

HDSET_Un_Register_SDP_Record(__BluetoothStackID, __HDSETPortID, __SDPRecordHandle) (SDP_Delete_Service_Record(__BluetoothStackID, __SDPRecordHandle))

3. The Service Name is always added at Attribute ID 0x0100. A Language Base Attribute ID List is created that specifies that 0x0100 is UTF-8 Encoded, English Language.

Prototype:

```
int BTPSAPI HDSET_Register_Headset_SDP_Record(unsigned int BluetoothStackID,
        unsigned int HDSETPortID, char *ServiceName, DWord_t *SDPServiceRecordHandle)
```

Parameters:

| | |
|-------------------------------|---|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| HDSETPortID | The HDSET Port ID this command applies to. This value MUST have been obtained by calling the HDSET_Open_Headset_Server_Port() function. |
| ServiceName | Name to appear in the SDP Database for this service. |
| SDPServiceRecordHandle | Returned handle to the SDP Database entry that may be used to remove the entry at a later time. |

Return:

Zero if successful.

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

BTHDSET_ERROR_NOT_INITIALIZED
 BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID
 BTHDSET_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.

HDSET_Register_Audio_Gateway_SDP_Record

This function adds a generic Audio Gateway Service Record to the SDP database.

Notes:

1. This function should only be called with the HDSET Port ID that was returned from the HDSET_Open_Audio_Gateway_Server_Port() function. This function should **never** be used with the HDSET Port ID returned from the HDSET_Open_Headset_Server_Port() function.
2. The Service Record Handle that is returned from this function will remain in the SDP Record Database until it is deleted by calling the SDP_Delete_Service_Record() function. A Macro is provided to delete the Service Record from the SDP Database. This Macro maps HDSET_Un_Register_SDP_Record() to SDP_Delete_Service_Record(), and is defined as follows:

```
HDSET_Un_Register_SDP_Record(__BluetoothStackID, __HDSETPortID,
    __SDPRecordHandle) (SDP_Delete_Service_Record(__BluetoothStackID,
    __SDPRecordHandle))
```

3. The Service Name is always added at Attribute ID 0x0100. A Language Base Attribute ID List is created that specifies that 0x0100 is UTF-8 Encoded, English Language.

Prototype:

```
int BTPSAPI HDSET_Register_Audio_Gateway_SDP_Record(unsigned int
    BluetoothStackID, unsigned int HDSETPortID, char *ServiceName,
    DWord_t *SDPServiceRecordHandle)
```

Parameters:

| | |
|-------------------------------|---|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| HDSETPortID | The HDSET Port ID this command applies to. This value MUST have been obtained by calling the HDSET_Open_Audio_Gateway_Server_Port() function. |
| ServiceName | Name to appear in the SDP Database for this service. |
| SDPServiceRecordHandle | Returned handle to the SDP Database entry that may be used to remove the entry at a later time. |

Return:

Zero if successful.

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

```
BTHDSET_ERROR_NOT_INITIALIZED
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID
BTHDSET_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.

HDSET_Open_Remote_Headset_Port

This function opens a Remote Headset port on the specified remote device.

Prototype:

```
int BTPSAPI HDSET_Open_Remote_Headset_Port(unsigned int BluetoothStackID,
      BD_ADDR_t BD_ADDR, unsigned int RemoteServerPort,
      Boolean_t SupportInBandRinging, HDSET_Event_Callback_t EventCallback,
      unsigned long CallbackParameter)
```

Parameters:

| | |
|-------------------------------|--|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| BD_ADDR | Address of the Bluetooth device to connect with. |
| RemoteServerPort | Remote Server Channel ID to connect with. This must fall in the range defined by the following constants: SPP_PORT_NUMBER_MINIMUM SPP_PORT_NUMBER_MAXIMUM |
| SupportInBandRinging | Specifies whether or not the Local Audio Gateway (the entity that is connecting to the Remote Headset) supports In Band Ringing or not (TRUE if supported). Currently this parameter is ignored. |
| Event_Callback | Function to call when events occur on this port. |
| CallbackParameter | A user-defined parameter (e.g., a tag value) that will be passed back to the user in the callback function with each packet. |

Return:

Positive, non-zero if successful. If this function is successful, the return value will represent the HDSET Port ID that can be passed to all other functions that require it.

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

```
BTHDSET_ERROR_INSUFFICIENT_RESOURCES
BTHDSET_ERROR_NOT_INITIALIZED
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID
BTHDSET_ERROR_INVALID_PARAMETER
```

Possible Events:

```
etHDSET_Open_Port_Confirmation
```

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.

HDSET_Open_Remote_Audio_Gateway_Port

This function opens a Remote Audio Gateway Port on the specified Remote Device.

Prototype:

```
int BTPSAPI HDSET_Open_Remote_Audio_Gateway_Port(
    unsigned int BluetoothStackID, BD_ADDR_t BD_ADDR,
    unsigned int RemoteServerPort, Boolean_t RemoteVolumeControlSupported,
    HDSET_Event_Callback_t EventCallback, unsigned long CallbackParameter)
```

Parameters:

| | |
|-------------------------------|---|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| BD_ADDR | Address of the Bluetooth device to connect with. |
| RemoteServerPort | Remote Server Channel ID to connect with. This must fall in the range defined by the following constants: SPP_PORT_NUMBER_MINIMUM SPP_PORT_NUMBER_MAXIMUM |
| RemoteVolumeControlSupported | Specifies whether or not the Headset supports Remote Audio Volume Controls (TRUE if supported). |
| Event_Callback | Function to call when events occur on this port. |
| CallbackParameter | A user-defined parameter (e.g., a tag value) that will be passed back to the user in the callback function with each packet. |

Return:

Positive, non-zero if successful. If this function is successful, the return value will represent the HDSET Port ID that can be passed to all other functions that require it.

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

```
BTHDSET_ERROR_INSUFFICIENT_RESOURCES
BTHDSET_ERROR_NOT_INITIALIZED
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID
BTHDSET_ERROR_INVALID_PARAMETER
```

Possible Events:

etHDSET_Port_Open_Confirmation

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.

HDSET_Ring_Indication

This function sends a Ring Indication to the remote side.

Prototype:

```
int BTPSAPI HDSET_Ring_Indication(unsigned int BluetoothStackID,  
    unsigned int HDSETPortID)
```

Parameters:

| | |
|-------------------------------|---|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| HDSETPortID | HDSET Port ID for which the connection has been established. |

Return:

Zero if successful.

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

```
BTHDSET_ERROR_NOT_INITIALIZED  
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID  
BTHDSET_ERROR_INVALID_PARAMETER
```

Possible Events:

etHDSET_Close_Port_Indication

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.

HDSET_Send_Button_Press

This function allows the local Headset to send a button press event to the remote Audio Gateway.

Notes:

This function should be used instead of the HDSET_Accept_Incoming_Call() and HDSET_End_Call() functions. The reason is that the aforementioned functions imply a call state. Since the actual call state is handled via the Audio Gateway, the Headset does not have any mechanism to actually determine the current call state. Because of this, this function will simply issue the button press and let the Audio Gateway decide how to process the request.

Prototype:

```
int BTPSAPI HDSET_Send_Button_Press(unsigned int BluetoothStackID,  
    unsigned int HDSETPortID)
```

Parameters:

| | |
|-------------------------------|---|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| HDSETPortID | HDSET Port ID for which the connection has been established. |

Return:

Zero if successful.

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

```
BTHDSET_ERROR_NOT_INITIALIZED  
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID  
BTHDSET_ERROR_INVALID_PARAMETER
```

Possible Events:

etHDSET_Close_Port_Indication

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.

HDSET_Accept_Incoming_Call

This function accepts or rejects a HDSET incoming call request.

Notes:

This function should no longer be used. This function has been superseded via the HDSET_Send_Button_Press() function. The reason is that the aforementioned function implies a call state. Since the actual call state is handled via the Audio Gateway, the Headset does not have any mechanism to actually determine the current call state. Because of this, this function will simply issue the button press and let the Audio Gateway decide how to process the request.

Prototype:

```
int BTPSAPI HDSET_Accept_Incoming_Call(unsigned int BluetoothStackID,  
    unsigned int HDSETPortID, Boolean_t AcceptCall)
```

Parameters:

| | |
|-------------------------------|---|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| HDSETPortID | HDSET Port ID for which the connection has been established. |

AcceptCall Specifies whether or not the connection should be accepted (TRUE if accepted).

Return:

Zero if successful.

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

BTHDSET_ERROR_INVALID_OPERATION
BTHDSET_ERROR_NOT_INITIALIZED
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID
BTHDSET_ERROR_INVALID_PARAMETER

Possible Events:

etHDSET_Audio_Connection_Indication
etHDSET_Close_Port_Indication

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.

HDSET_End_Call

This function allows the local Headset to request that the remote Audio Gateway end the current call.

Notes:

This function should no longer be used. This function has been superseded via the HDSET_Send_Button_Press() function. The reason is that the aforementioned function implies a call state. Since the actual call state is handled via the Audio Gateway, the Headset does not have any mechanism to actually determine the current call state. Because of this, this function will simply issue the button press and let the Audio Gateway decide how to process the request.

Prototype:

```
int BTPSAPI HDSET_End_Call(unsigned int BluetoothStackID,  
    unsigned int HDSETPortID)
```

Parameters:

BluetoothStackID¹ Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
HDSETPortID HDSET Port ID for which the connection has been established.

Return:

Zero if successful.

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

BTHDSET_ERROR_INVALID_OPERATION
BTHDSET_ERROR_NOT_INITIALIZED
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID
BTHDSET_ERROR_INVALID_PARAMETER

Possible Events:

etHDSET_Audio_Disconnection_Indication
etHDSET_Close_Port_Indication

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.

HDSET_Close_Port

This function is used to close a HDSET Port that was previously opened by any of the following mechanisms:

- Successful call to the HDSET_Open_Remote_Headset_Port() function.
- Successful call to the HDSET_Open_Remote_Audio_Gateway_Port() function.
- Successful call to the HDSET_Open_Headset_Server_Port() function
- Successful call to the HDSET_Open_Audio_Gateway_Server_Port() function.

This function does NOT Un-Register a HDSET Server Port from the system, it ONLY disconnects any connection that is currently active on the Server Port. The HDSET_Close_Server_Port() function can be used to Un-Register the Server Port.

Prototype:

```
int BTPSAPI HDSET_Close_Port(unsigned int BluetoothStackID,  
                             unsigned int HDSETPortID)
```

Parameters:

| | |
|-------------------------------|--|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| HDSETPortID | The HDSET port to close. This is the value that was returned from one of the above Open functions. |

Return:

Zero if successful.

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

BTHDSET_ERROR_NOT_INITIALIZED
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID
BTHDSET_ERROR_INVALID_PARAMETER

Possible Events:**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.

HDSET_Set_Speaker_Gain

This function allows the local entity a mechanism of notifying the Remote entity (either Headset OR Audio Gateway) that the speaker gain has changed.

Prototype:

```
int BTPSAPI HDSET_Set_Speaker_Gain(unsigned int BluetoothStackID,  
    unsigned int HDSETPortID, unsigned int SpeakerGain)
```

Parameters:

| | |
|-------------------------------|--|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| HDSETPortID | The HDSET port to adjust speaker settings for. This is the value that was returned from one of the above Open functions. |
| SpeakerGain | The new speaker gain setting. The SpeakerGain parameter MUST be between the values of HDSET_SPEAKER_GAIN_MINIMUM and HDSET_SPEAKER_GAIN_MAXIMUM. |

Return:

Zero if successful.

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

```
BTHDSET_ERROR_INVALID_OPERATION  
BTHDSET_ERROR_NOT_INITIALIZED  
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID  
BTHDSET_ERROR_INVALID_PARAMETER
```

Possible Events:

etHDSET_Close_Port_Indication

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.

HDSET_Set_Microphone_Gain

This function allows the local entity a mechanism of notifying the Remote entity (either Headset OR Audio Gateway) that the microphone gain has changed.

Prototype:

```
int BTPSAPI HDSET_Set_Microphone_Gain(unsigned int BluetoothStackID,  
    unsigned int HDSETPortID, unsigned int MicrophoneGain)
```

Parameters:

| | |
|-------------------------------|--|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| HDSETPortID | The HDSET port to adjust microphone settings for. This is the value that was returned from one of the above Open functions. |
| MicrophoneGain | The new microphone gain setting. The MicrophoneGain parameter MUST be between the values of HDSET_MICROPHONE_GAIN_MINIMUM and HDSET_MICROPHONE_GAIN_MAXIMUM. |

Return:

Zero if successful.

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

```
BTHDSET_ERROR_INVALID_OPERATION  
BTHDSET_ERROR_NOT_INITIALIZED  
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID  
BTHDSET_ERROR_INVALID_PARAMETER
```

Possible Events:

etHDSET_Close_Port_Indication

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.

HDSET_Setup_Audio_Connection

This function allows the ability for the local Audio Gateway to establish an active audio connection to the remote Headset. This function allows the ability to specify if the audio connection is being made to satisfy in-band ringing (final parameter is specified as TRUE) or for an active call (final parameter is FALSE). The distinction is made because the Audio Gateway will automatically disconnect the connection when the audio channel is closed if the audio was brought up for an active call. This is in contrast to in-band ringing, where the control connection will remain connected if the Audio Gateway manually disconnects the audio channel (via the HDSET_Release_Audio_Connection() function).

Prototype:

```
int BTPSAPI HDSET_Setup_Audio_Connection(unsigned int BluetoothStackID,  
    unsigned int HDSETPortID, Boolean_t InBandRinging)
```

Parameters:

| | |
|-------------------------------|---|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| HDSETPortID | The HDSET port of the local Audio Gateway (client or server). This is the value that was returned from one of the above Open functions. |
| InBandRinging | Specifies whether or not the audio connection is being made to support in-band ringing (TRUE) or is being made for an actual active call (FALSE). |

Return:

Zero if successful.

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

```
BTHDSET_ERROR_INVALID_OPERATION  
BTHDSET_ERROR_NOT_INITIALIZED  
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID  
BTHDSET_ERROR_INVALID_PARAMETER
```

Possible Events:

```
etHDSET_Close_Port_Indication  
etHDSET_Audio_Connection_Indication
```

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.

HDSET_Release_Audio_Connection

This function allows the ability for the local Audio Gateway to release/disconnect an active audio connection from the remote Headset.

Prototype:

```
int BTPSAPI HDSET_Release_Audio_Connection(unsigned int BluetoothStackID,  
    unsigned int HDSETPortID)
```

Parameters:

| | |
|-------------------------------|---|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
|-------------------------------|---|

HDSETPortID The HDSET port of the local Audio Gateway (client or server). This is the value that was returned from one of the above Open functions.

Return:

Zero if successful.

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

BTHDSET_ERROR_INVALID_OPERATION
BTHDSET_ERROR_NOT_INITIALIZED
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID
BTHDSET_ERROR_INVALID_PARAMETER

Possible Events:

etHDSET_Close_Port_Indication
etHDSET_Audio_Disconnection_Indication

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.

HDSET_Send_Audio_Data

This function allows a local entity to send SCO Audio Data to the Remote entity (either Headset OR Audio Gateway).

Notes:

If this function returns BTPS_ERROR_INSUFFICIENT_BUFFER_SPACE then the application must wait for the etHDSET_Audio_Transmit_Buffer_Empty_Indication event and re-transmit the selected data.

Prototype:

int BTPSAPI **HDSET_Send_Audio_Data**(unsigned int BluetoothStackID,
 unsigned int HDSETPortID, Byte_t AudioDataLength, Byte_t *AudioData)

Parameters:

| | |
|-------------------------------|---|
| BluetoothStackID ¹ | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| HDSETPortID | The HDSET port to adjust microphone settings for. This is the value that was returned from one of the above Open functions. |
| AudioDataLength | The length (in bytes) of the audio data. |
| AudioData | Pointer to the data. |

Return:

Zero if successful.

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

BTHDSET_ERROR_INVALID_OPERATION
BTHDSET_ERROR_NOT_INITIALIZED
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID
BTHDSET_ERROR_INVALID_PARAMETER
BTPS_ERROR_INSUFFICIENT_BUFFER_SPACE

Possible Events:

etHDSET_Close_Port_Indication

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.

HDSET_Get_Server_Mode¹

This function retrieves the current Headset/Audio Gateway Server Mode for a specified Headset/Audio Gateway Server.

Prototype:

int BTPSAPI **HDSET_Get_Server_Mode**(unsigned int BluetoothStackID,
unsigned int HDSETPortID, unsigned long *ServerModeMask)

Parameters:

| | |
|-------------------------------|---|
| BluetoothStackID ² | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| HDSETPortID | The HDSET port to adjust microphone settings for. This is the value that was returned from one of the above Open functions. |
| ServerModeMask | Pointer to Server Mode variable which will receive the current Server Mode. May return one of the following: HDSET_SERVER_MODE_AUTOMATIC_ACCEPT_CONNECTION HDSET_SERVER_MODE_MANUAL_ACCEPT_CONNECTION |

Return:

Zero if successful.

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

BTHDSET_ERROR_NOT_INITIALIZED
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID
BTHDSET_ERROR_INVALID_PARAMETER

Possible Events:

etHDSET_Close_Port_Indication

Notes:

1. This function is used for Headset/Audio Gateway servers which use Bluetooth Security Mode 2.
2. 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.

HDSET_Set_Server_Mode¹

The function changes the current Headset/Audio Gateway Server Mode for a specified Headset/Audio Gateway Server.

Prototype:

```
int BTPSAPI HDSET_Set_Server_Mode(unsigned int BluetoothStackID,
    unsigned int HDSETPortID, unsigned long ServerModeMask)
```

Parameters:

| | |
|-------------------------------|---|
| BluetoothStackID ² | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize(). |
| HDSETPortID | The HDSET port to adjust microphone settings for. This is the value that was returned from one of the above Open functions. |
| ServerModeMask | Server Mode variable to change the specified Headset/Audio Gateway Server's Server Mode to. May be one of the following: HDSET_SERVER_MODE_AUTOMATIC_ACCEPT_CONNECTION HDSET_SERVER_MODE_MANUAL_ACCEPT_CONNECTION |

Return:

Zero if successful.

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

```
BTHDSET_ERROR_NOT_INITIALIZED
BTHDSET_ERROR_INVALID_BLUETOOTH_STACK_ID
BTHDSET_ERROR_INVALID_PARAMETER
```

Possible Events:

etHDSET_Close_Port_Indication

Notes:

1. This function is used for Headset/Audio Gateway servers which use Bluetooth Security Mode 2.
2. 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.

2.2 Headset Profile Event Callback Prototypes

The event callback functions mentioned in the Headset Profile open commands all accept the callback function described by the following prototype.

HDSET_Event_Callback_t

Prototype of callback function passed in one of the HDSET open commands.

Prototype:

```
void (BTPSAPI *HDSET_Event_Callback_t)(unsigned int BluetoothStackID,
    HDSET_Event_Data_t *HDSET_Event_Data, unsigned long CallbackParameter)
```

Parameters:

BluetoothStackID¹ Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().

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

```
typedef struct
{
    HDSET_Event_Type_t    Event_Data_Type;
    Word_t                Event_Data_Size;
    union
    {
        HDSET_Open_Port_Request_Indication_Data_t
            *HDSET_Open_Port_Request_Indication_Data;
        HDSET_Open_Port_Indication_Data_t        *HDSET_Open_Port_Indication_Data;
        HDSET_Open_Port_Confirmation_Data_t      *HDSET_Open_Port_Confirmation_Data;
        HDSET_Close_Port_Indication_Data_t       *HDSET_Close_Port_Indication_Data;
        HDSET_Ring_Indication_Data_t             *HDSET_Ring_Indication_Data;
        HDSET_Button_Pressed_Indication_Data_t   *HDSET_Button_Pressed_Indication_Data;
        HDSET_Speaker_Gain_Indication_Data_t     *HDSET_Speaker_Gain_Indication_Data;
        HDSET_Microphone_Gain_Indication_Data_t  *HDSET_Microphone_Gain_Indication_Data;
        HDSET_Audio_Connection_Indication_Data_t *HDSET_Audio_Connection_Indication_Data;
        HDSET_Audio_Disconnection_Indication_Data_t *HDSET_Audio_Disconnection_Indication_Data;
        HDSET_Audio_Data_Indication_Data_t       *HDSET_Audio_Data_Indication_Data;
    } Event_Data;
} HDSET_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 (e.g., tag value) that was defined in the callback registration.

Return:**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 Headset Profile Events

The possible Headset Profile events from the Bluetooth stack are listed in the table below and are described in the text that follows:

| Event | Description |
|---|---|
| etHDSSET_Open_Port_Request_Indication | Indicates that a Remote Port Open request has been made. |
| etHDSSET_Open_Port_Indication | Indicates that a Remote Port Open request has been received. |
| etHDSSET_Open_Port_Confirmation | Confirms that a Port Open request has been responded to or has encountered an error. |
| etHDSSET_Close_Port_Indication | Indicates that a port has been closed (unregistered). |
| etHDSSET_Ring_Indication | Indicates that a ring indication has been received. |
| etHDSSET_Button_Pressed_Indication | Indicates that the remote headset has pressed the button. |
| etHDSSET_Speaker_Gain_Indication | Indicates that the speaker gain has changed. |
| etHDSSET_Microphone_Gain_Indication | Indicates that the microphone gain has changed. |
| etHDSSET_Audio_Connection_Indication | Indicates that an audio connection has been created. |
| etHDSSET_Audio_Disconnection_Indication | Indicates that an audio connection has been destroyed. |
| etHDSSET_Audio_Data_Indication | Indicates that audio data is available. |
| etHDSSET_Audio_Transmit_Buffer_Empty_Indication | Indicates that the audio data buffer for the specified device has space for at least 1 more packet. |

etHDSSET_Open_Port_Request_Indication

Dispatched when a Remote Port Open Request has been made.

Return Structure:

```
typedef struct
{
    unsigned int    HDSETPortID;
    BD_ADDR_t      BD_ADDR;
}HDSET_Open_Port_Request_Indication_Data_t;
```

Event Parameters:

| | |
|-------------|---|
| HDSetPortID | Identifier of the HDSET server connection. |
| BD_ADDR | Address of the Bluetooth Device making the request. |

etHDSET_Open_Port_Indication

Indicate that a Remote Port Open request has been received.

Return Structure:

```
typedef struct
{
    unsigned int    HDSETPortID;
    BD_ADDR_t      BD_ADDR;
}HDSET_Open_Port_Indication_Data_t;
```

Event Parameters:

| | |
|-------------|---|
| HDSetPortID | Identifier of the HDSET server connection. |
| BD_ADDR | Address of the Bluetooth Device making the request. |

etHDSET_Open_Port_Confirmation

Confirm that a Port Open request has been responded to or has encounter an error.

Return Structure:

```
typedef struct
{
    unsigned int    HDSETPortID;
    unsigned int    PortOpenStatus;
} HDSET_Open_Port_Confirmation_Data_t;
```

Event Parameters:

| | |
|----------------|---|
| HDSETPortID | Identifier of the HDSET connection. |
| PortOpenStatus | One of the following possible status values: HDSET_OPEN_PORT_STATUS_SUCCESS HDSET_OPEN_PORT_STATUS_CONNECTION_TIMEOUT HDSET_OPEN_PORT_STATUS_CONNECTION_REFUSED HDSET_OPEN_PORT_STATUS_AUDIO_CONNECTION_ERROR HDSET_OPEN_PORT_STATUS_UNKNOWN_ERROR |

etHDSET_Close_Port_Indication

Indicate that a port has been closed (unregistered).

Return Structure:

```
typedef struct
{
    unsigned int  HDSETPortID;
    unsigned int  PortCloseStatus;
} HDSET_Close_Port_Indication_Data_t;
```

Event Parameters:

| | |
|-----------------|--|
| HDSETPortID | Identifier of the HDSET connection. |
| PortCloseStatus | One of the following possible status values: HDSET_CLOSE_PORT_STATUS_SUCCESS HDSET_CLOSE_PORT_STATUS_AUDIO_CONNECTION_ERROR HDSET_CLOSE_PORT_STATUS_UNKNOWN_ERROR |

etHDSET_Ring_Indication

Indicates that a ring indication has been received.

Return Structure:

```
typedef struct
{
    unsigned int  HDSETPortID;
} HDSET_Ring_Indication_Data_t;
```

Event Parameters:

| | |
|-------------|-------------------------------------|
| HDSETPortID | Identifier of the HDSET connection. |
|-------------|-------------------------------------|

etHDSET_Button_Pressed_Indication

Indicates that a button has been pressed at the headset.

Return Structure:

```
typedef struct
{
    unsigned int  HDSETPortID;
    Boolean_t     ConnectionPresent;
} HDSET_Button_Pressed_Indication_Data_t;
```

Event Parameters:

| | |
|-------------------|--|
| HDSETPortID | Identifier of the HDSET server connection. |
| ConnectionPresent | TRUE if a connection exists. |

etHDSET_Speaker_Gain_Indication

Indicates that the speaker gain has changed.

Return Structure:

```
typedef struct
{
    unsigned int  HDSETPortID;
    unsigned int  SpeakerGain;
} HDSET_Speaker_Gain_Indication_Data_t;
```

Event Parameters:

| | |
|-------------|-------------------------------------|
| HDSETPortID | Identifier of the HDSET connection. |
| SpeakerGain | New speaker gain value. |

etHDSET_Microphone_Gain_Indication

Indicates that the microphone gain has changed.

Return Structure:

```
typedef struct
{
    unsigned int  HDSETPortID;
    unsigned int  MicrophoneGain;
} HDSET_Microphone_Gain_Indication_Data_t;
```

Event Parameters:

| | |
|----------------|-------------------------------------|
| HDSETPortID | Identifier of the HDSET connection. |
| MicrophoneGain | New microphone gain value. |

etHDSET_Audio_Connection_Indication

Indicates that an audio connection has been created.

Return Structure:

```
typedef struct
{
    unsigned int  HDSETPortID;
} HDSET_Audio_Connection_Indication_Data_t;
```

Event Parameters:

| | |
|-------------|-------------------------------------|
| HDSETPortID | Identifier of the HDSET connection. |
|-------------|-------------------------------------|

etHDSET_Audio_Disconnection_Indication

Indicates that an audio connection has been destroyed.

Return Structure:

```
typedef struct
{
    unsigned int    HDSETPortID;
} HDSET_Audio_Disconnection_Indication_Data_t;
```

Event Parameters:

HDSETPortID Identifier of the HDSET connection.

etHDSET_Audio_Data_Indication

Indicates that audio data is available.

Return Structure:

```
typedef struct
{
    unsigned int    HDSETPortID;
    Byte_t          AudioDataLength;
    Byte_t          *AudioData;
    Word_t          PacketStatus;
} HDSET_Audio_Data_Indication_Data_t;
```

Event Parameters:

HDSETPortID Identifier of the HDSET connection.

AudioDataLength Length of the data.

AudioData Pointer to the audio data.

PacketStatus The status (as reported by the baseband) of the packet. Valid valid values are one of the following:

```

HCI_SCO_FLAGS_PACKET_STATUS_MASK_
CORRECTLY_RECEIVED_
DATA
HCI_SCO_FLAGS_PACKET_STATUS_MASK_
POSSIBLY_INVALID_DATA
HCI_SCO_FLAGS_PACKET_STATUS_MASK_NO_
DATA_RECEIVED
HCI_SCO_FLAGS_PACKET_STATUS_MASK_DATA_
PARTIALLY_LOST
```

etHDSET_Audio_Transmit_Buffer_Empty_Indication

Indicates that audio transmit buffer for the specified device has space for at least 1 more packet.

Return Structure:

```
typedef struct
{
    unsigned int  HDSETPortID;
} HDSET_Audio_Transmit_Buffer_Empty_Indication_Data_t;
```

Event Parameters:

| | |
|-------------|-------------------------------------|
| HDSETPortID | Identifier of the HDSET connection. |
|-------------|-------------------------------------|

3. File Distributions

The header files that are distributed with the Bluetooth Headset Profile Library are listed in the table below.

| File | Contents/Description |
|------------|---|
| HDSETAPI.h | Bluetooth Headset Profile API definitions |
| SS1BTHDS.h | Bluetooth Headset Profile Include file |