
# Audio/Video Remote Control Profile (AVRCP)

## Application Programming Interface Reference Manual

**Profile Version: 1.4**

**Release: 4.0.1**
**January 10, 2014**


Louisville, KY www.stonestreetone.com

# Table Of Contents

# 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 description of all programming interfaces for the Bluetooth Audio/Video Remote Control Profile provided by Bluetopia. Chapter 2 contains a description of the programming interfaces for this profile. And, Chapter 3 contains the header file name list for the Bluetooth Audio/Video Remote Control Profile library.

## 1.1 Scope

This reference manual provides information on the Audio/Video Remote Control Profile 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

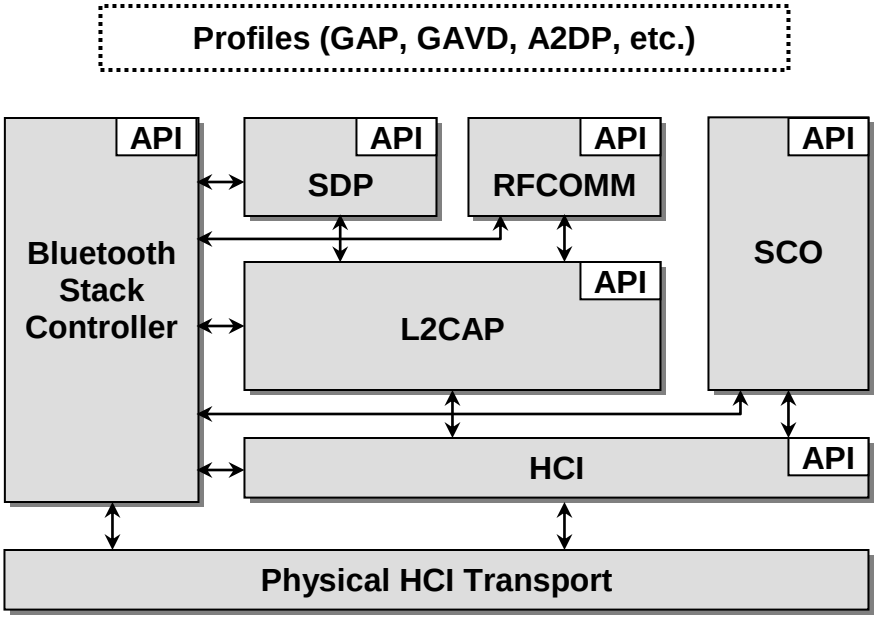


**Figure 1-1 The Stonestreet One Bluetooth Protocol Stack**

## 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, Volumes 0-4*, version 2.1 + EDR, July 26, 2007.
2. *Specification of the Bluetooth System, Bluetooth Core Specification Addendum 1*, June 26, 2008.
3. *Specification of the Bluetooth System, Volume 0, Master Table of Contents & Compliance Requirements*, version 3.0+HS, April 21, 2009.
4. *Specification of the Bluetooth System, Volume 1, Architecture and Terminology Overview*, version 3.0+HS, April 21, 2009.
5. *Specification of the Bluetooth System, Volume 2, Core System Package [Controller Volume]*, version 3.0+HS, April 21, 2009.
6. *Specification of the Bluetooth System, Volume 3, Core System Package [Host Volume]*, version 3.0+HS, April 21, 2009.
7. *Specification of the Bluetooth System, Volume 4, Host Controller Interface [Transport Layer]*, version 3.0+HS, April 21, 2009.
8. *Specification of the Bluetooth System, Volume 5, Core System Package [AMP Controller Volume]*, version 3.0+HS, April 21, 2009.
9. *Specification of the Bluetooth System, Volume 0, Master Table of Contents & Compliance Requirements*, version 4.0, June 30, 2010.
10. *Specification of the Bluetooth System, Volume 1, Architecture and Terminology Overview*, version 4.0, June 30, 2010.
11. *Specification of the Bluetooth System, Volume 2, Core System Package [BR/EDR Controller Volume]*, version 4.0, June 30, 2010.
12. *Specification of the Bluetooth System, Volume 3, Core System Package [Host Volume]*, version 4.0, June 30, 2010.
13. *Specification of the Bluetooth System, Volume 4, Host Controller Interface [Transport Layer]*, version 4.0, June 30, 2010.
14. *Specification of the Bluetooth System, Volume 5, Core System Package [AMP Controller Volume]*, version 4.0, June 30, 2010.
15. *Specification of the Bluetooth System, Volume 6, Core System Package [Low Energy Controller Volume]*, version 4.0, June 30, 2010.
16. *Bluetooth Assigned Numbers*, Bluetooth.org, April 5, 2012.
17. *Audio/Video Control Transport Protocol Specification*, version 1.3, June 26, 2008.
18. *Audio/Video Remote Control Profile*, version 1.4, June 26, 2008.

19. *Bluetopia® Protocol Stack, Application Programming Interface Reference Manual,* version 4.0.1, August 27, 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 |
|---|---|
| AVRCP | Audio/Video Remote Control Profile (Bluetooth Profile) |
| API | Application Programming Interface |
| BD_ADDR | Bluetooth Device Address |
| BR | Basic Rate |
| BT | Bluetooth |
| EDR | Enhanced Data Rate |
| GAP | Generic Access Profile (Bluetooth Profile) |
| AVCTP | Audio/Video Control Transport Protocol |
| HS | High Speed |
| LE | Low Energy |
| LSB | Least Significant Bit |
| MSB | Most Significant Bit |

# 2. AVRCP Programming Interface

The Audio/Video Remote Control Profile programming interface defines the protocols and procedures to be used to implement AVRCP capabilities. As the Audio/Video Remote Control Profile is so heavily dependent on the Audio/Video Control Transport Protocol (AVCTP), there are no additional functions required to implement this profile. The application programmer can simply use the AVCTP API in conjunction with the constants and MACROs that are defined in the AVRCP API header file to implement the AVRCP Profile.

## 2.1 AVRCP Commands

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

| Function | Description |
|---|---|
| AVRCP_Format_Unit_Info_Command | Formats an AVRCP Unit Info Command into a provided buffer. |
| AVRCP_Format_Unit_Info_Response | Formats an AVRCP Unit Info Response into a provided buffer. |
| AVRCP_Format_Subunit_Info_Command | Formats an AVRCP Subunit Info Command into a provided buffer. |
| AVRCP_Format_Subunit_Info_Response | Formats an AVRCP Subunit Info Response into a provided buffer. |
| AVRCP_Format_Pass_Through_Command | Formats an AVRCP Pass Through Command into a provided buffer. |
| AVRCP_Format_Pass_Through_Response | Formats an AVRCP Pass Through Response into a provided buffer. |
| AVRCP_Format_Vendor_Dependent_Generic_Command | Formats an AVRCP Vendor Dependent Generic Command into a provided buffer. |
| AVRCP_Format_Vendor_Dependent_Generic_Response | Formats an AVRCP Vendor Dependent Generic Command into a provided buffer. |
| AVRCP_Format_Browsing_Channel_Generic_Message | Formats an AVRCP Browsing Channel Generic Message into a provided buffer. |
| AVRCP_Decode_Message | Parses a provided buffer containing an AVRCP Response Message. |
| AVRCP_Free_Decoded_Message | Frees the resources associated with an AVRCP Decoded Message. |

## AVRCP_Format_Unit_Info_Command

This function is a utility function that exists to format an AVRCP Unit Info Command into a provided buffer. This function accepts the size of the buffer and a pointer to a buffer to write the command data. This function returns the amount of data that was written to the buffer or negative return code on failure.

Note:

1. Passing zero and NULL for the BufferLength and Buffer parameters (respectively) instructs this function to calculate and return the number of bytes that are required to hold the specified response.

**Prototype:**

int BTPSAPI **AVRCP_Format_Unit_Info_Command** (unsigned int BluetoothStackID, unsigned int BufferLength, Byte_t *Buffer)

**Parameters:**

| | |
|---|---|
| BluetoothStackID[1] | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize. |
| BufferLength | Length of the provided buffer to which the command data will be written. |
| Buffer | Pointer to a buffer to which the command data will be written. |

**Return:**

The number of bytes written if successful.

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

BTAVRCP_ERROR_BUFFER_TOO_SMALL
BTAVCTP_ERROR_NOT_INITIALIZED
BTAVCTP_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.

## AVRCP_Format_Unit_Info_Response

This function is a utility function that exists to format an AVRCP Unit Info Response into a provided buffer. This function accepts the data to populate into the response (required), followed by the size of the buffer, and a pointer to a buffer to write the response data. This function returns the amount of data that was written to the buffer or a negative return code on failure.

Note:

1. Passing zero and NULL for the BufferLength and Buffer parameters (respectively) instructs this function to calculate and return the number of bytes that are required to hold the specified response.

**Prototype:**

int BTPSAPI **AVRCP_Format_Unit_Info_Response** (unsigned int BluetoothStackID, AVRCP_Unit_Info_Response_Data_t *UnitInfoResponseData, unsigned int BufferLength, Byte_t *Buffer)

**Parameters:**

| | |
|---|---|
| BluetoothStackID[1] | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize. |
| UnitInfoResponseData | Pointer to a structure containing the data to write into the provided buffer. |
| BufferLength | Length of the provided buffer to which the response data will be written. |
| Buffer | Pointer to a buffer to which the response data will be written. |

**Return:**

The number of bytes written if successful.

An error code if negative; one of the following values:
BTAVRCP_ERROR_BUFFER_TOO_SMALL
BTAVCTP_ERROR_NOT_INITIALIZED
BTAVCTP_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.

## AVRCP_Format_Subunit_Info_Command

This function is a utility function that exists to format an AVRCP Subunit Info Command into a provided buffer. This function accepts the data to populate into the command (required), followed by the size of the buffer, and a pointer to a buffer to write the response data. This function returns the amount of data that was written to the buffer or a negative return code on failure.

Note:

1. Passing zero and NULL for the BufferLength and Buffer parameters (respectively) instructs this function to calculate and return the number of bytes that are required to hold the specified response.

**Prototype:**

int BTPSAPI **AVRCP_Format_Subunit_Info_Command** (unsigned int BluetoothStackID, AVRCP_Subunit_Info_Command_Data_t *SubunitInfoCommandData, unsigned int BufferLength, Byte_t *Buffer)

**Parameters:**

| | |
|---|---|
| BluetoothStackID[1] | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize. |
| SubunitInfoCommandData | Pointer to a structure containing the data to write into the provided buffer. |
| BufferLength | Length of the provided buffer to which the command data will be written. |
| Buffer | Pointer to a buffer to which the command data will be written. |

**Return:**

The number of bytes written if successful.

An error code if negative; one of the following values:
BTAVRCP_ERROR_BUFFER_TOO_SMALL
BTAVCTP_ERROR_NOT_INITIALIZED
BTAVCTP_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.

## AVRCP_Format_Subunit_Info_Response

This function is a utility function that exists to format an AVRCP Subunit Info Response into a provided buffer. This function accepts the data to populate into the response (required), followed by the size of the buffer, and a pointer to a buffer to write the response data. This function returns the amount of data that was written to the buffer or a negative return code on failure.

Note:

1. Passing zero and NULL for the BufferLength and Buffer parameters (respectively) instructs this function to calculate and return the number of bytes that are required to hold the specified response.

**Prototype:**

int BTPSAPI **AVCTP_Cleanup**(unsigned int BluetoothStackID, AVRCP_Subunit_Info_Response_Data_t *SubunitInfoResponseData, unsigned int BufferLength, Byte_t *Buffer)

**Parameters:**

| | |
|---|---|
| BluetoothStackID[1] | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize. |
| SubunitInfoResponseData | Pointer to a structure containing the data to write into the provided buffer. |
| BufferLength | Length of the provided buffer to which the response data will be written. |
| Buffer | Pointer to a buffer to which the response data will be written. |

**Return:**

The number of bytes written if successful.

An error code if negative; one of the following values:
BTAVRCP_ERROR_BUFFER_TOO_SMALL
BTAVCTP_ERROR_NOT_INITIALIZED
BTAVCTP_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.

## AVRCP_Format_Pass_Through_Command

This function is a utility function that exists to format an AVRCP Pass Through Command into a provided buffer. This function accepts the data to populate into the command (required), followed by the size of the buffer, and a pointer to a buffer to write the response data. This function returns the amount of data that was written to the buffer or a negative return code on failure.

Note:

1. Passing zero and NULL for the BufferLength and Buffer parameters (respectively) instructs this function to calculate and return the number of bytes that are required to hold the specified response.

**Prototype:**

int BTPSAPI **AVRCP_Format_Pass_Through_Command** (
unsigned int BluetoothStackID,
AVRCP_Pass_Through_Command_Data_t *PassThroughCommandData,
unsigned int BufferLength, Byte_t *Buffer)

**Parameters:**

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

| | |
|---|---|
| PassThroughCommandData | Pointer to a structure containing the data to write into the provided buffer. |
| BufferLength | Length of the provided buffer to which the command data will be written. |
| Buffer | Pointer to a buffer to which the command data will be written. |

**Return:**

The number of bytes written if successful.

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

BTAVRCP_ERROR_BUFFER_TOO_SMALL
BTAVRCP_ERROR_MESSAGE_TOO_LONG
BTAVCTP_ERROR_NOT_INITIALIZED
BTAVCTP_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.

## AVRCP_Format_Pass_Through_Response

This function is a utility function that exists to format an AVRCP Pass Through Response into a provided buffer. This function accepts the data to populate into the response (required), followed by the size of the buffer, and a pointer to a buffer to write the response data. This function returns the amount of data that was written to the buffer or a negative return code on failure.

Note:

1. Passing zero and NULL for the BufferLength and Buffer parameters (respectively) instructs this function to calculate and return the number of bytes that are required to hold the specified response.

**Prototype:**

int BTPSAPI **AVRCP_Format_Pass_Through_Response** (unsigned int BluetoothStackID,
AVRCP_Pass_Through_Response_Data_t *PassThroughResponseData,
unsigned int BufferLength, Byte_t *Buffer)

**Parameters:**

| | |
|---|---|
| BluetoothStackID[1] | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize. |
| PassThroughResponseData | Pointer to a structure containing the data to write into the provided buffer. |

| | |
|---|---|
| BufferLength | Length of the provided buffer to which the response data will be written. |
| Buffer | Pointer to a buffer to which the response data will be written. |

**Return:**

The number of bytes written if successful.

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

BTAVRCP_ERROR_BUFFER_TOO_SMALL
BTAVRCP_ERROR_MESSAGE_TOO_LONG
BTAVCTP_ERROR_NOT_INITIALIZED
BTAVCTP_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.

## AVRCP_Format_Vendor_Dependent_Generic_Command

This function is a utility function that exists to format an AVRCP Vendor Dependent Generic Command into a provided buffer. This function accepts the data to populate into the command (required), followed by the size of the buffer, and a pointer to a buffer to write the response data. This function returns the amount of data that was written to the buffer or a negative return code on failure.

Note:

1. Passing zero and NULL for the BufferLength and Buffer parameters (respectively) instructs this function to calculate and return the number of bytes that are required to hold the specified response.

**Prototype:**

int BTPSAPI **AVRCP_Format_Vendor_Dependent_Generic_Command** (
unsigned int BluetoothStackID,
AVRCP_Vendor_Dependent_Generic_Command_Data_t
*VendorDependentCommandData, unsigned int BufferLength, Byte_t *Buffer)

**Parameters:**

| | |
|---|---|
| BluetoothStackID[1] | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize. |
| VendorDependentCommandData | Pointer to a structure containing the data to write into the provided buffer. |
| BufferLength | Length of the provided buffer to which the command data will be written. |
| Buffer | Pointer to a buffer to which the command data will be written. |

**Return:**

The number of bytes written if successful.

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

BTAVRCP_ERROR_BUFFER_TOO_SMALL
BTAVRCP_ERROR_MESSAGE_TOO_LONG
BTAVCTP_ERROR_NOT_INITIALIZED
BTAVCTP_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.

## AVRCP_Format_Vendor_Dependent_Generic_Response

This function is a utility function that exists to format an AVRCP Vendor Dependent Generic Response into a provided buffer. This function accepts the data to populate into the response (required), followed by the size of the buffer, and a pointer to a buffer to write the response data. This function returns the amount of data that was written to the buffer or a negative return code on failure.

Note:

1. Passing zero and NULL for the BufferLength and Buffer parameters (respectively) instructs this function to calculate and return the number of bytes that are required to hold the specified response.

**Prototype:**

int BTPSAPI **AVRCP_Format_Vendor_Dependent_Generic_Response** (
unsigned int BluetoothStackID,
AVRCP_Vendor_Dependent_Generic_Response_Data_t
*VendorDependentResponseData, unsigned int BufferLength, Byte_t *Buffer)

**Parameters:**

| | |
|---|---|
| BluetoothStackID[1] | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize. |
| VendorDependentResponseData | Pointer to a structure containing the data to write into the provided buffer. |
| BufferLength | Length of the provided buffer to which the response data will be written. |
| Buffer | Pointer to a buffer to which the response data will be written. |

**Return:**

The number of bytes written if successful.

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

BTAVRCP_ERROR_BUFFER_TOO_SMALL
BTAVRCP_ERROR_MESSAGE_TOO_LONG
BTAVCTP_ERROR_NOT_INITIALIZED
BTAVCTP_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.

## AVRCP_Format_Browsing_Channel_Generic_Message

This function is a utility function that exists to format an AVRCP Browsing Channel Generic Message into a provided buffer. This function accepts the data to populate into the response (required), followed by the size of the buffer, and a pointer to a buffer to write the response data. This function returns the amount of data that was written to the buffer or a negative return code on failure.

Note:

1. Passing zero and NULL for the BufferLength and Buffer parameters (respectively) instructs this function to calculate and return the number of bytes that are required to hold the specified response.

**Prototype:**

int BTPSAPI **AVRCP_Format_Browsing_Channel_Generic_Message** (
unsigned int BluetoothStackID,
AVRCP_Browsing_Channel_Generic_Message_Data_t
*BrowsingChannelGenericMessageData, unsigned int BufferLength, Byte_t *Buffer)

**Parameters:**

| | |
|---|---|
| BluetoothStackID[1] | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize. |
| BrowsingChannelGenericMessageData | Pointer to a structure containing the data to write into the provided buffer. |
| BufferLength | Length of the provided buffer to which the response data will be written. |
| Buffer | Pointer to a buffer to which the response data will be written. |

**Return:**

The number of bytes written if successful.

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

BTAVRCP_ERROR_BUFFER_TOO_SMALL
BTAVRCP_ERROR_MESSAGE_TOO_LONG
BTAVCTP_ERROR_NOT_INITIALIZED

BTAVCTP_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.

## AVRCP_Decode_Message

This function parses a provided AVRCP Response Message. This function takes an AVRCP message and creates a populated structure containing the message data. This function returns zero if successful or a negative error code if there was an error.

Note:

1. If this function returns success (zero) then the caller MUST call the AVRCP_Free_Decoded_Message() function with the decoded Message buffer that was pass to this function. This will guarantee that all resouces for the message will be freed.
2. The decoded message will have pointers in it that point to memory that is located within the Message Buffer pass to this function (MessageData). This message data cannot be freed or altered until after the AVRCP_Free_Decoded_Message() function is called.

**Prototype:**

int BTPSAPI **AVRCP_Decode_Message** (unsigned int BluetoothStackID,
Boolean_t BrowsingChannelMessage, Boolean_t Response,
unsigned int MessageLength, Byte_t *MessageData,
AVRCP_Message_Information_t *MessageInformation)

**Parameters:**

| | |
|---|---|
| BluetoothStackID[1] | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize. |
| BrowsingChannelMessage | Set if the message to decode is a Browse Channel message. |
| Response | Set if the message to decode is a Response message. |
| MessageLength | Length of the message in bytes. |
| MessageData | Pointer to a buffer that contains the message to decode. |
| MessageInformation | Pointer to a structure that will contain the parsed message data. |

**Return:**

Zero if successful.

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

BTAVCTP_ERROR_INVALID_PARAMETER
BTAVRCP_ERROR_UNABLE_TO_DECODE_MESSAGE
BTAVRCP_ERROR_UNABLE_TO_DECODE_VENDOR_DEPENDENT

**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.

## AVRCP_Free_Decoded_Message

This function frees the resources associated with a decoded AVRCP message. This is the data that was populated into the AVRCP_Message_Information_t buffer that was passed to the AVRCP_Decode_Message() function. This function frees all embedded resources except the AVRCP_Message_Information_t structure itself which is owned by the caller. This function returns zero if successful or a negative error code if there was an error.

**Prototype:**

int BTPSAPI **AVRCP_Free_Decoded_Message** (unsigned int BluetoothStackID, AVRCP_Message_Information_t *MessageInformation)

**Parameters:**

| | |
|---|---|
| BluetoothStackID[1] | Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize. |
| MessageInformation | Pointer to an AVRCP_Message_Information_t structure to free. |

**Return:**

Zero if successful.

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

# 3. File Distributions

The header files that are distributed with the Bluetooth Audio/Video Remote Control Profile Library are listed in the table below.

| File | Contents/Description |
|---|---|
| AVRCPAPI.h | Bluetooth Audio/Video Remote Control Profile API definitions |
| AVRTypes.h | Bluetooth Audio/Video Remote Control Profile Type definitions |
| SS1BTAVR.h | Bluetooth Audio/Video Remote Control Profile Include file |