

***Bluetooth*® A2DP AND AVRCP PROFILES**

**iWRAP APPLICATION NOTE**

Friday, 21 March 2014

Version 1.6

## **Copyright © 2000-2014 Bluegiga Technologies**

All rights reserved.

Bluegiga Technologies assumes no responsibility for any errors which may appear in this manual. Furthermore, Bluegiga Technologies reserves the right to alter the hardware, software, and/or specifications detailed here at any time without notice and does not make any commitment to update the information contained here. Bluegiga's products are not authorized for use as critical components in life support devices or systems.

The WRAP, Bluegiga Access Server, Access Point and iWRAP are registered trademarks of Bluegiga Technologies.

The *Bluetooth* trademark is owned by the *Bluetooth* SIG Inc., USA and is licensed to Bluegiga Technologies. All other trademarks listed herein are owned by their respective owners.

## VERSION HISTORY

Version	Comment
1.0	First version
1.1	Added chapter 3.5.4
1.2	Updated to reflect the changes in iWRAP5, corrected errors in chapter 3.1, detailed chapter 3.5.3
1.3	iWRAP5 release updates
1.4	Added AVRCP RAW, iOS behaviour details
1.5	AAC info added to chapter 5, generic iWRAP6 release updates in chapter 2
1.6	iWRAP6 beta release updates: AVRCP 1.5 Controller commands, AVRCP browsing

## TABLE OF CONTENTS

1	Introduction .....	5
1.1	Advanced Audio Distribution Profile .....	5
1.2	Audio/video Remote Control Profile .....	6
2	iWRAP firmware overview .....	7
3	Using A2DP and AVRCP with iWRAP .....	9
3.1	Configuration .....	9
3.1.1	A2DP Sink and AVRCP Controller .....	9
3.1.2	A2DP Source and AVRCP Target .....	10
3.1.3	Security configuration .....	11
3.2	Service discovery .....	12
3.3	Connection establishment .....	13
3.3.1	A2DP connection .....	13
3.3.2	AVRCP connection .....	14
3.4	Connection termination .....	15
3.4.1	A2DP connection .....	15
3.4.2	AVRCP connection .....	15
3.5	AVRCP browsing .....	16
3.5.1	Browsing general concepts .....	16
3.5.2	Browsing workflow .....	17
3.6	General A2DP and AVRCP information .....	19
3.6.1	A2DP commands .....	19
3.6.2	AVRCP commands .....	20
3.6.3	AVRCP PDU .....	21
3.6.4	AVRCP BROWSE .....	27
3.6.5	AVRCP {raw} detail .....	33
3.6.6	iWRAP commands related to audio .....	36
4	Power saving .....	37
5	Audio quality improvement .....	38
5.1	Enabling AAC Audio Codec for A2DP .....	38
5.2	Enabling aptX Audio Codec for A2DP .....	38
5.3	Routing the A2DP Audio to I2S (External Codec) .....	39
6	References .....	40
7	Contact Information .....	41

# 1 Introduction

This application note discusses *Bluetooth* Advanced Audio Distribution Profile (A2DP) and *Bluetooth* Audio/video Remote Control Profile (AVRCP) their advantages and how these profiles can be used. Also practical examples are given how the A2DP and AVRCP profiles are used with the iWRAP firmware.

## 1.1 Advanced Audio Distribution Profile

A2DP describes how stereo-quality audio can be streamed from a media source to a sink. The audio source is the music player and the audio sink is the wireless headset or wireless stereo speakers.

The profile defines two roles of an audio device: source and sink.

- **A2DP Source** – A device is the source when it acts as a source of a digital audio stream that is delivered to the SINK of the piconet.
- **A2DP Sink** – A device is the sink when it acts as a sink of a digital audio stream delivered from the SOURCE on the same piconet.

A2DP defines the protocols and procedures that realize distribution of audio content of high-quality in mono or stereo on ACL channels. The term “advanced audio,” therefore, should be distinguished from “*Bluetooth* audio,” which indicates distribution of narrow band voice on SCO channels as defined in the baseband specification.

A2DP profile includes mandatory support for low complexity sub-band codec (SBC) and supports optionally MPEG-1,2 Audio, MPEG-2,4 AAC, ATRAC or other codecs.

The audio data is compressed in a proper format for efficient use of the limited bandwidth. Surround sound distribution is not included in the scope of this profile.



Figure 1: Typical A2DP use case

Source: [1]

## 1.2 Audio/video Remote Control Profile

AVRCP is designed to provide a standard interface to control TVs, Hi-Fi equipment, or others to allow a single remote control (or other device) to control all the A/V equipment to which a user has access. It may be used in concert with A2DP or VDP.

Basically your action manipulates the control. You can adjust menu functions that are already commonly used, such as adjusting the brightness of your TV or hue, or a VCR timer, as well as audio functions like sound adjustments, play, pause, skip, etc.

The AVRCP defines two roles, that of a controller and target device.

- **Controller** – The controller is typically considered the remote control device.
- **Target** – The target device is the one whose characteristics are being altered.

In a “walkman” type media player scenario, the control device may be a headset that allows tracks to be skipped and the target device would be the actual medial player.

In AVRCP, the controller translates the detected user action to the A/V control signal, and then transmits it to a remote *Bluetooth* enabled device. The functions available for a conventional infrared remote controller can be realized in this protocol. The remote control described in this protocol is designed specifically for A/V control only.



Figure 2: Typical AVRCP use case

Source: [2]

## 2 iWRAP firmware overview

iWRAP is an embedded firmware running entirely on the RISC processor of WT11i, WT12, WT32, WT32i and WT41 modules. It implements the full *Bluetooth* protocol stack and many *Bluetooth* profiles. All software layers, including application software, run on the internal RISC processor in a protected user software execution environment known as a Virtual Machine (VM).

The host system can interface to iWRAP firmware through one or more physical interfaces, which are also shown in the figure below. The most common interfacing is done through the UART interface by using the ASCII commands that iWRAP firmware supports. With these ASCII commands, the host can access *Bluetooth* functionality without paying any attention to the complexity, which lies in the *Bluetooth* protocol stack. GPIO interface can be used for event monitoring and command execution. PCM, SPDIF, I2S or analog interfaces are available for audio. The available interfaces depend on the used hardware.

The user can write application code to the host processor to control iWRAP firmware using ASCII commands or GPIO events. In this way, it is easy to develop *Bluetooth* enabled applications.

On WT32 and WT32i, there is an extra DSP processor available for data/audio processing.

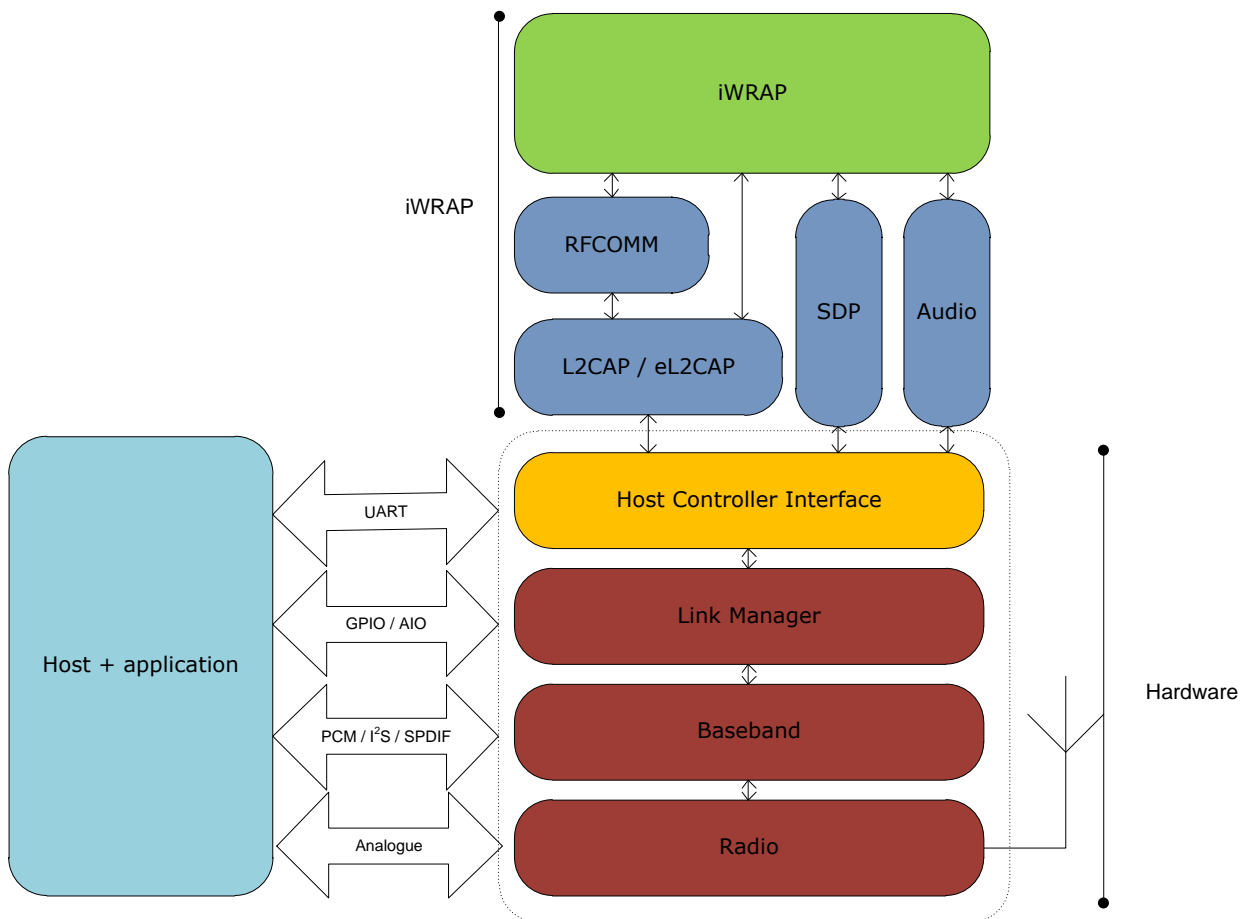


Figure 3: iWRAP Stack

In the figure above, a *Bluetooth* module with iWRAP firmware could be connected to a host system for example through the UART interface. The options are:

- If the host system has a processor, software can be used to control iWRAP by using ASCII based commands or GPIO events.
- If there is no need to control iWRAP, or the host system does not need a processor, iWRAP can be configured to be totally transparent and autonomous, in which case it only accepts connections or automatically opens them.
- GPIO lines that *Bluetooth* modules offer can also be used together with iWRAP to achieve additional functionality, such as Carrier Detect or DTR signaling.
- Audio interfaces can be used to transmit audio over a *Bluetooth* link.



## 3 Using A2DP and AVRCP with iWRAP

This chapter instructs the A2DP and AVRCP usage and configuration with the iWRAP firmware.

### 3.1 Configuration

#### 3.1.1 A2DP Sink and AVRCP Controller

Perhaps the most common use case for A2DP and AVRCP is the wireless headset with remote control capabilities. The A2DP Sink enables audio streaming from an audio source, whereas AVRCP Controller facilitates over-the-air control of the audio stream.

A2DP Sink is enabled with iWRAP command:

**“SET PROFILE A2DP SINK”**.

AVRCP controller on the other hand is enabled with command:

**“SET PROFILE AVRCP CONTROLLER”**

The mobile phones, tablets and PC to recognize the device as a wireless headset the *Bluetooth* Class of Device (CoD) must be configured to reflect the device’s capabilities. Some devices may not be able to discover and connect the A2DP unless the CoD is properly set.

The A2DP specification mandates that for the A2DP Sink, the Rendering Service bit (0x040000) is set. It is also recommended to set the Audio Service bit (0x200000) and set the Major Device Class to Audio/Video (0x000400).

The Minor Device Class should be set according to what type of device is in question, for example Headphones (0x000018) and Hi-Fi Audio Device (0x000028). The combined CoD is 0x240438.

Class of device is configured with iWRAP command:

**“SET BT CLASS {cod}”**

Finally a reset is needed to for the A2DP profile to become active.

Below is an example how to configure iWRAP as A2DP sink and AVRCP controller with a proper CoD.

```
SET PROFILE A2DP SINK
SET PROFILE AVRCP CONTROLLER
SET BT CLASS 200428
RESET
```

### 3.1.2 A2DP Source and AVRCP Target

Another very common scenario is an audio source which can be remote controlled, such as a home stereo system.

Much like in the previous example, “**SET PROFILE A2DP SOURCE**” and “**SET PROFILE AVRCP TARGET**” must be issued to enable the profiles.

As in the previous example, the Class of Device must be set properly. In the case of A2DP Source, the specification mandates the use of the Capturing Service bit (0x080000). The Audio Service (0x200000) and Audio/Video Major Device Class (0x000400) should be used, but they are not mandatory.

A common example Minor Device Class is Hi-Fi Audio Device (0x000028). The combined CoD is then 0x280428.

Finally a reset is needed for the A2DP profile to become active.

```
SET PROFILE A2DP SOURCE  
SET PROFILE AVRCP TARGET  
SET BT CLASS 200428  
RESET
```

### 3.1.3 Security configuration

To be able to pair with other *Bluetooth* enabled devices the *Bluetooth* security needs to be properly configured. iWRAP support Secure Simple Pairing (SSP) defined in *Bluetooth 2.1 + EDR* specification, and the use of it is mandatory, but PIN code pairing is also supported to enable pairing with legacy devices.

In order to enable SSP and PIN code pairing, the following configuration commands are needed:

**“SET BT SSP 3 0”** This enables SSP “just works” mode, where no PIN code entry or passkey verification is needed. To enable the optional man-in-the-middle protection, please refer to iWRAP user guide.

**“SET BT AUTH \* <pin>”** This command enables PIN code pairing, to support pairing with legacy defines. **<pin>** is the desired PIN code, which can be 1-16 alphanumeric characters.

Finally a reset is needed to for the security settings profile to become active.

Below is an example how to enable SSP just works and PIN code pairing in iWRAP.

```
SET BT SSP 3 0
SET BT AUTH * 0000
RESET
```

## 3.2 Service discovery

*Bluetooth* technology enables wireless service discovery, so you can find out the capabilities the remote device supports. Wireless service discovery uses the *Bluetooth* Service Discovery Profile (SDP).

With iWRAP the service discovery is performed with command: "**SDP {bd\_addr} {uuid}**".

<b>bd_addr</b>	<i>Bluetooth</i> device address of the remote device.
<b>uuid</b>	Universally unique identifier. Refers to the <i>Bluetooth</i> profile one wants to discover. For A2DP sink the <b>uuid</b> is 110B, for A2DP source 110A for AVRCP target 110C and for AVRCP controller 110E.

Below is an example how to perform a service discovery to an A2DP sink.

### SDP 00:07:80:81:66:6f 110B

```
SDP 00:07:80:81:66:6f < I SERVICENAME S "Stereo Headset" > < I PROTOCOLDESCRIPTORLIST < < U L2CAP I 19 > < U 0019 I 100 > > >
```

SDP

<b>Stereo Headset</b>	= Service name
<b>19</b>	= L2CAP psm for A2DP sink

Below is an example how to perform a service discovery to a A2DP source.

### SDP 00:07:80:93:0c:aa 110A

```
SDP 00:07:80:93:0c:aa < I SERVICENAME S "Stereo Audio" > < I PROTOCOLDESCRIPTORLIST < < U L2CAP I 19 > < U 0019 I 100 > > >
```

SDP

Below is an example how to perform a service discovery to a AVRCP controller.

### SDP 00:07:80:93:0c:aa 110C

```
SDP 00:07:80:81:66:6f < I SERVICENAME S "A/V Controller" > < I PROTOCOLDESCRIPTORLIST < < U L2CAP I 17 > < U 0017 I 104 > > >
```

SDP

<b>A/V Controller</b>	= Service name
<b>17</b>	= L2CAP psm for A2DP source

Below is an example how to perform a service discovery to a AVRCP target..

### SDP 00:07:80:93:0c:aa 110E

```
SDP 00:07:80:93:0c:aa < I SERVICENAME S "A/V Target" > < I PROTOCOLDESCRIPTORLIST < < U L2CAP I 17 > < U 0017 I 103 > > >
```

SDP

## 3.3 Connection establishment

### 3.3.1 A2DP connection

A2DP profile requires two connections, one for a control channel and second for a data channel. However iWRAP automatically sets up both the connections so the user does not need to worry about setting up two channels.

The A2DP connection is opened, typical to iWRAP, with a **CALL** command:

**“CALL {*bd\_addr*} 19 A2DP”**

*bd\_addr*      *Bluetooth* device address of the remote device.

Below is an example how to set up an A2DP connection.

```
CALL 00:07:80:81:66:6f 19 A2DP
```

```
CALL 0
```

```
CONNECT 0 A2DP 19
```

```
CONNECT 1 A2DP 19
```

```
A2DP STREAMING START 0
```

A typical indications of outgoing call and successful connection are received (**CALL** and **CONNECT**). Two connection events are typically received indicating the establishment of control and data channels.

Also typically a third event is received indicating that A2DP streaming is started:

**“A2DP STREAMING START {*link\_id*}”**

*link\_id*      Numeric connection identifier which started streaming.

**Note:**

iWRAP allows you to set up several simultaneous A2DP connections. However, only one of the connections can be streaming audio at a time.

### 3.3.2 AVRCP connection

The AVRCP connection is opened, typical to iWRAP, with a **CALL** command:

**“CALL {bd\_addr} 17 A2DP”**

*bd\_addr* Bluetooth device address of the remote device.

Below is an example how to set up an AVRCP connection.

```
CALL 00:07:80:81:66:6f 17 AVRCP
```

```
CALL 2
```

```
CONNECT 2 AVRCP 17
```

Typical indications for outgoing call and successful connection are received (**CALL** and **CONNECT**) and in the case of when the connection establishment is not successful **CALL** and **NO CARRIER** events are received.

## 3.4 Connection termination

### 3.4.1 A2DP connection

The A2DP connection is simply closed with iWRAP command “**CLOSE {link\_id}**”. One should always terminate the first A2DP connection (control channel) and not second connection (data channel). Terminating the control channel makes iWRAP also terminate the data channel.

Below is an example of A2DP connection termination.

```
CLOSE 0  
A2DP STREAMING STOP 0  
NO CARRIER 0 ERROR 0  
NO CARRIER 1 ERROR 0
```

Typically first an indication of A2DP streaming stop is received:

“**A2DP STREAMING STOP {link\_id}**”

*link\_id*

Numeric connection identifier of link which started streaming.

Then normal **NO CARRIER** events are received for both A2DP connections.

### 3.4.2 AVRCP connection

The A2DP connection is simply closed with iWRAP command “**CLOSE {link\_id}**”.

Below is an example of AVRCP connection termination.

```
CLOSE 2  
NO CARRIER 2 ERROR 0
```

Then normal **NO CARRIER** event is received.

## 3.5 AVRCP browsing

Browsing is a feature introduced in AVRCP specification version 1.4. iWRAP6 implements AVRCP version 1.5 Controller, so browsing is supported on the Controller side.

AVRCP browsing is used to navigate the virtual file system on any media player on the Target, where browsing is supported. AVRCP browsing facilitates listing and selecting media players, navigating up and down folders and listing folder contents, retrieving information about items on the player, and searching for specific items.

### 3.5.1 Browsing general concepts

#### 3.5.1.1 Browsing channel

Browsing is done over a separate L2CAP channel, which is opened after the regular AVRCP channel has been established. All regular AVRCP commands will still be sent over the regular AVRCP channel. The browsing channel is only used for retrieving information about media on the Target device; all other actions, such as adding found items to the now playing list, will still be executed over the regular AVRCP channel.

Either the Controller or the Target can open the browsing channel, and the channel can be opened and closed on demand, or more commonly, kept open as long as the regular AVRCP link is open. The browsing channel must be closed before the regular AVRCP link is closed.

The browsing channel's connection type in the iWRAP LIST output is AVRCP, but its L2CAP PSM number is 0x001b, whereas the PSM of the regular AVRCP connection is 0x0017.

#### 3.5.1.2 UIDs and UID counter

Every item on a browsing-capable player has a unique 128-bit UID that is valid over both AVRCP channels. Any item discovered over the browsing channel can be addressed with regular AVRCP PDUs (for example, PDU 0x74, PLAY\_ITEM) by using the discovered UID.

A media item's UID may change if more media items are added to or removed from the player's virtual filesystem. The UID counter is a mechanism to keep track of UID changes. Every time the Controller sends a AVRCP BROWSE LIST or AVRCP BROWSE SEARCH browsing command, the Target returns, along with other data, the current value of the UID counter. Every Controller command that addresses media items by UID must also include the last UID counter value the Controller has received.

If the UID counter values do not match (that is, something has been added, removed, or changed on the player's virtual filesystem), the Target will reject the command with status 0x05, UIDs changed. The Controller should then repeat the LIST command it used to retrieve the UID of the item.



### 3.5.2 Browsing workflow

Since a Target can have multiple media players that support browsing, the first step is to get a list of media players and select a player for browsing.

Example: open AVRCP and AVRCP browsing links, and request the first 16 items on the media player list, then set the player with ID **0x0001** as the browsed player.

```
CALL 00:ff:00:bb:aa:cc 17 AVRCP  
CALL 0  
CONNECT 0 AVRCP 17  
@0 AVRCP BROWSE OPEN  
CONNECT 1 AVRCP 1b  
@0 AVRCP BROWSE LIST 0 0 f ff  
AVRCP_BROWSE 1 GET_FOLDER_ITEMS UIDC 0000 ITEMS 0001 < PLAYER ID 0001 TYPE 01  
SUBTYPE 00000000 PLAY_STATUS STOPPED FEATURES 0000000000b7011c0200000000000000  
NAME 0005 Music >  
@0 AVRCP BROWSE SETPLAYER 1  
AVRCP_BROWSE 1 SET_BROWSED_PLAYER UIDC 0000 COUNT 00000005 FOLDERS 00
```

**Note:**

If the AVRCP browse channel is already open (for example opened up by the phone) the **AVRCP BROWSE OPEN** will return syntax error.

After a player is selected for browsing, it's time to retrieve the folder listing and start looking for media to play.

```
@0 AVRCP BROWSE LIST 1 0 f ff  
AVRCP_BROWSE 1 GET_FOLDER_ITEMS UIDC 0000 ITEMS 0005 < FOLDER < UID  
0400000000000000 TYPE ARTISTS PLAYABLE NO NAME 0007 Artists > FOLDER < UID  
0500000000000000 TYPE TITLES PLAYABLE NO NAME 0005 Songs > FOLDER < UID  
0600000000000000 TYPE ALBUMS PLAYABLE NO NAME 0006 Albums > FOLDER < UID  
0900000000000000 TYPE ARTISTS PLAYABLE NO NAME 0009 Composers > FOLDER < UID  
0a00000000000000 TYPE GENRES PLAYABLE NO NAME 0006 Genres > >
```

Suppose we want to navigate directly to the songs folder and select some items to put in the playing queue. The songs folder contains 1509 items.

```
@0 AVRCP BROWSE PATH 0 1 0500000000000000  
AVRCP_BROWSE 1 CHANGE_PATH COUNT 000005e5
```

Next, let's list 3 media items starting from the 2<sup>nd</sup> item, and ask for all attributes.

**@0 AVRCP BROWSE LIST 1 1 3 ff**

AVRCP\_BROWSE 3 GET\_FOLDER\_ITEMS UIDC 0000 ITEMS 0003 < < MEDIA UID 9b1afa9f4c12ce59  
TYPE AUDIO NAME 0003 ABC ATTRIBUTES 01 < < TITLE NAME 0003 ABC > > > < MEDIA UID  
2bf8243352cebe67 TYPE AUDIO NAME 000a About Life ATTRIBUTES 01 < < TITLE NAME 000a About  
Life > > > < MEDIA UID b0bdea9e7ecb5224 TYPE AUDIO NAME 0008 Absorbed ATTRIBUTES 01 < <  
TITLE NAME 0008 Absorbed > > >

Now we'll start playing the song titled "ABC".

**@0 AVRCP PDU 74 1 9b1afa9f4c12ce59 0**

AVRCP 0 PLAY\_ITEM\_RSP OK

Let's search for an album called "Rain Dogs" to put in the playback queue, and then retrieve the 10 first search results.

**@0 AVRCP BROWSE SEARCH "rain dogs"**

AVRCP\_BROWSE 1 SEARCH UIDC 0000 COUNT 0000001b

**@0 AVRCP BROWSE LIST 3 0 9 0**

AVRCP\_BROWSE 1 GET\_FOLDER\_ITEMS UIDC 0000 ITEMS 000a < < MEDIA UID 0e24d548196f8e70  
TYPE AUDIO NAME 0016 Anywhere I Lay My Head ATTRIBUTES 01 < < TITLE NAME 0016 Anywhere I  
Lay My Head > > > < MEDIA UID 8942c6d007e09367 TYPE AUDIO NAME 0010 Big Black Mariah  
ATTRIBUTES 01 < < TITLE NAME 0010 Big Black Mariah > > > < MEDIA UID 884472822381fb9b TYPE  
AUDIO NAME 000a Blind Love ATTRIBUTES 01 < < TITLE NAME 000a Blind Love > > > < MEDIA UID  
06eec7217eed9522 TYPE AUDIO NAME 0011 Bride of Rain Dog ATTRIBUTES 01 < < TITLE NAME 0011  
Bride of Rain Dog > > > < MEDIA UID bab8b9091baf783a TYPE AUDIO NAME 000e Cemetery Polka  
ATTRIBUTES 01 < < TITLE NAME 000e Cemetery Polka > > > < MEDIA UID 04bb90d8c8c54d00 TYPE  
AUDIO NAME 000a Clap Hands ATTRIBUTES 01 < < TITLE NAME 000a Clap Hands > > > < MEDIA UID  
163f19ccd93ee837 TYPE AUDIO NAME 000f Diamonds & Gold ATTRIBUTES 01 < < TITLE NAME 000f  
Diamonds & Gold > > > < MEDIA UID f73b53fa79c152b0 TYPE AUDIO NAME 0016 Do You Close Your  
Eyes ATTRIBUTES 01 < < TITLE NAME 0016 Do You Close Your Eyes > > > < MEDIA UID  
97526ba94b21557c TYPE AUDIO NAME 000e Downtown Train ATTRIBUTES 01 < < TITLE NAME 000e  
Downtown Train > > > < MEDIA UID db0b9bfc5abaa800 TYPE AUDIO NAME 000f Gun Street Girl  
ATTRIBUTES 01 < < TITLE NAME 000f Gun Street Girl > > >

Let's add the first song in the search results to the playback queue.

**@0 AVRCP PDU 90 2 0e24d548196f8e70 0**

AVRCP 0 ADD\_TO\_NOW\_PLAYING\_RSP OK

## 3.6 General A2DP and AVRCP information

This chapter contains general information and tips about the iWRAP and A2DP profile for the implementers.

### 3.6.1 A2DP commands

The table below lists the possible A2DP commands. See iWRAP user guide for exact details.

Command	Function
<b>A2DP STREAMING START</b>	Starts A2DP streaming
<b>A2DP ATREAMING STOP</b>	Stops A2DP streaming

**Table 1:** Supported A2DP commands

### 3.6.2 AVRCP commands

The table below lists the possible AVRCP commands.

**Note:**

Not all devices support all commands. For example, iOS devices do not support remote volume control using **AVRCP UP**, or **AVRCP DN**, or **AVRCP MUTE** commands. This is a limitation of iOS and not an iWRAP firmware problem.

Command	Description
<b>AVRCP UP</b>	Volume up
<b>AVRCP DN</b>	Volume down
<b>AVRCP MUTE</b>	Mute
<b>AVRCP STOP</b>	Stop
<b>AVRCP PLAY</b>	Play
<b>AVRCP PAUSE</b>	Pause
<b>AVRCP REWIND</b>	Rewind
<b>AVRCP FAST_FORWARD</b>	Fast Forward
<b>AVRCP FORWARD</b>	Forward (next song)
<b>AVRCP BACKWARD</b>	Backward (previous song)
<b>AVRCP {raw}</b>	AVRCP command in raw hex mode. (See section 3.6.4) <b>Available codes:</b> 00-13, 20-2c, 30-37, 40-4b, 50-51, 71-75 and 7e
<b>AVRCP PDU {PDU_ID} [parameters]</b>	<b>PDU_ID</b> AVRCP PDU ID <b>parameters</b> AVRCP PDU parameters  Please refer to the following page for available <b>PDU_IDs</b> and <b>parameters</b> .

**Table 2:** Available AVRCP commands

**Note:** See iWRAP user guide for more details and examples of how to use the AVRCP commands.

### 3.6.3 AVRCP PDU

#### 3.6.3.1 Syntax

**AVRCP PDU** command is used by the AVRCP Controller to send metadata request Protocol Data Units to the Target.

#### Synopsis

**AVRCP PDU** {*PDU\_ID*} [*parameters*]

PDU ID	Description and parameters
10	Get capabilities command. Query for events or Company_ID's the Target supports. <b>Parameters:</b> 2 Query supported Company_ID's. 3 Query supported events.
11	List player application settings. No parameters.
12	List possible values for a player application setting. <b>Parameters:</b> { <i>setting_id</i> } See list at the end of this command's description.
13	Get current values of player application settings. <b>Parameters:</b> { <i>number of settings</i> } Number of following parameters. Followed by: { <i>setting_id</i> } See list at the end of this command's description.
14	Set current values of player application settings. <b>Parameters:</b> { <i>number of settings</i> } Number of <i>setting_id</i> -value-pairs that follow. Followed by:

	<p><b>{setting_id} {value}</b></p> <p>See list at the end of this command's description.</p>
<b>20</b>	<p>Get attributes of the currently playing track.</p> <p><b>Parameters:</b></p> <p><b>{number of attributes}</b></p> <p>Number of attributes that follow. If zero, list all available information.</p> <p>Followed by (unless number of attributes is zero):</p> <p><b>[attribute_id]</b></p> <p>See list at the end of this command's description.</p>
<b>30</b>	<p>Get the playing status, length and position of the current track. No parameters.</p>
<b>31</b>	<p>Register notification of events. This will request the Target to notify us when a track is changed for instance.</p> <p><b>Parameters:</b></p> <p><b>{event_id}</b></p> <p>See list at the end of this command's description.</p>
<b>50</b>	<p>Set the absolute volume on the target device.</p> <p><b>Parameters:</b></p> <p><b>{volume}</b></p> <p>The requested volume level in percentage, range 0-100.</p>
<b>60</b>	<p>Set the addressed player. If the Target has multiple media players in the media player list, you can select which player should receive the AVRCP commands.</p> <p><b>Parameters:</b></p> <p><b>{Player ID}</b></p> <p>The ID of the player</p>
<b>74</b> <b>90</b>	<p>74: Start playing a track immediately.</p> <p>90: Add track to the now playing list (playback queue).</p> <p><b>Parameters:</b></p> <p><b>{scope}</b></p> <p>0: Media player list 1: Virtual filesystem 2: Search results 3: Now playing list</p> <p><b>{UID}</b></p> <p>UID of the media item. Use the browsing channel to discover UIDs.</p>

	<p><b>{UID counter}</b></p> <p>Latest UID counter value received. See section 3.5.1.2 for details.</p>
--	--

**Note:** The AVRCP profile specification contains the details of the possible PDU ID and a detailed description of the parameters.

**Note:** Not all the AVRCP commands are supported by the AVRCP target devices and most of them actually implement only a small subset of all the available commands.

### 3.6.3.2 Events

AVRCP PDU command may return the following events

Events
<b>AVRCP {PDU_ID name}_RSP [<i>parsed data</i>]</b>
<b>AVRCP RSP PDU_ID {PDU_ID}, data: [<i>unparsed data</i>]</b>
<b>AVRCP {PDU_ID name}_RSP REJ</b>

List of supported **attribute\_ids**, **event\_ids**, **setting\_ids**, and their respective **values**.

Type	ID	Explanation
Attribute	0x01	Song title
Attribute	0x02	Artist
Attribute	0x03	Album title
Attribute	0x04	Track number in numeric ASCII text
Attribute	0x05	Total number of tracks on album in numeric ASCII text
Attribute	0x06	Genre
Attribute	0x07	Playing time in milliseconds in numeric ASCII text
Setting	0x01	Equalizer setting, values: <b>0x01</b> Off <b>0x02</b> On
Setting	0x02	Repeat setting, values: <b>0x01</b> Off <b>0x02</b> Single track <b>0x03</b> All tracks <b>0x04</b> Group
Setting	0x03	Shuffle setting, values: <b>0x01</b> Off <b>0x02</b> All tracks <b>0x03</b> Group
Setting	0x04	Scan setting, values: <b>0x01</b> Off <b>0x02</b> All tracks <b>0x03</b> Group
Event	0x01	Playback status changed
Event	0x02	Track changed
Event	0x03	Track reached end
Event	0x04	Track reached start



<b>Event</b>	<b>0x05</b>	Playback position changed
<b>Event</b>	<b>0x06</b>	Battery status changed
<b>Event</b>	<b>0x07</b>	System status changed
<b>Event</b>	<b>0x08</b>	Player application setting changed
<b>Event</b>	<b>0x09</b>	Now playing changed
<b>Event</b>	<b>0x0a</b>	Available players changed
<b>Event</b>	<b>0x0b</b>	Addressed players changed
<b>Event</b>	<b>0x0c</b>	UIDs changed
<b>Event</b>	<b>0x0d</b>	Volume changed

### 3.6.3.3 Examples

Query supported AVRCP events.

#### **AVRCP PDU 10 3**

AVRCP GET\_CAPABILITIES\_RSP EVENT COUNT 3 PLAYBACK\_STATUS\_CHANGED TRACK\_CHANGED PLAYBACK\_POSITION\_CHANGED

Ask the Target about its player application settings, their possible values and change a value.

#### **AVRCP PDU 11**

AVRCP LIST\_APPLICATION\_SETTING\_ATTRIBUTES\_RSP COUNT 2 REPEAT SHUFFLE

#### **AVRCP PDU 12 2**

AVRCP LIST\_APPLICATION\_SETTING\_VALUES\_RSP COUNT 3 1 2 3

#### **AVRCP PDU 13 1 2**

AVRCP GET\_APPLICATION\_SETTING\_VALUE\_RSP COUNT 1 REPEAT OFF

#### **AVRCP PDU 14 1 2 2**

AVRCP SET\_APPLICATION\_SETTING\_VALUE\_RSP

#### **AVRCP PDU 13 1 2**

AVRCP GET\_APPLICATION\_SETTING\_VALUE\_RSP COUNT 1 REPEAT SINGLE\_TRACK

Ask the Target about the title and artist of the song that is currently playing and ask it to notify us if the playback status changes.

#### **AVRCP PDU 20 2 1 2**

AVRCP GET\_ELEMENT\_ATTRIBUTES\_RSP COUNT 2 TITLE "Cold Women and Warm Beer" ARTIST "The Black League"

#### **AVRCP PDU 31 1 1**

AVRCP REGISTER\_NOTIFICATION\_RSP INTERIM PLAYBACK\_STATUS\_CHANGED PLAYING  
(the interim response is received right after the request to confirm we were registered for notification)

AVRCP REGISTER\_NOTIFICATION\_RSP CHANGED PLAYBACK\_STATUS\_CHANGED PAUSED  
(the changed response is received when the playing status changes)

## 3.6.4 AVRCP BROWSE

### 3.6.4.1 Syntax

**AVRCP BROWSE** is used to browse an AVRCP Target's virtual filesystem.

Synopsis
<b>AVRCP BROWSE</b> {cmd [params]}

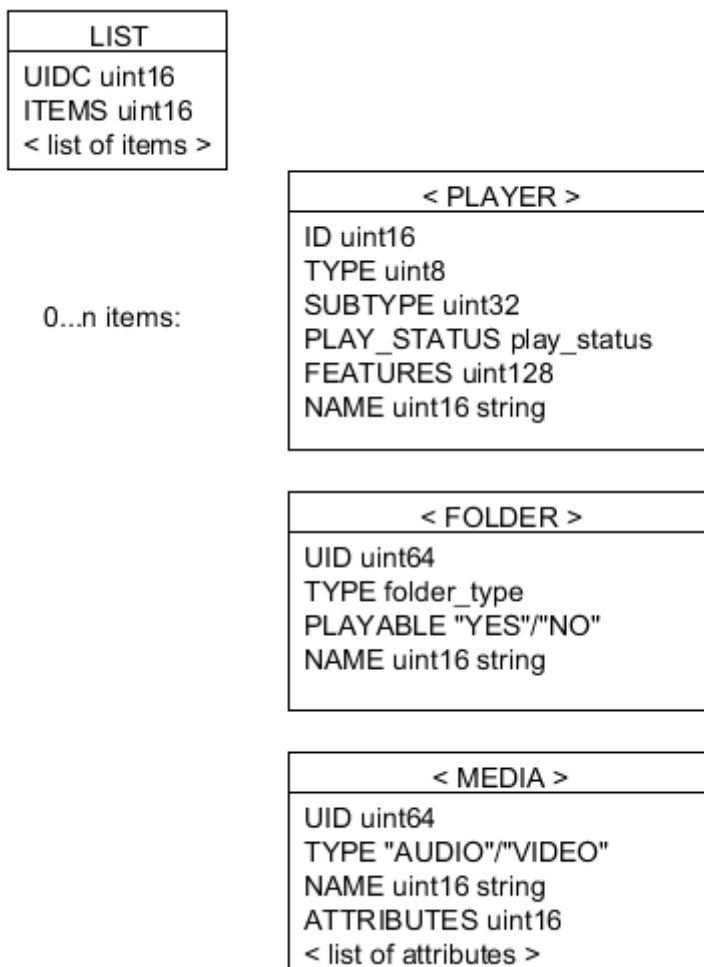
cmd	Description and parameters
<b>OPEN</b>	<p>Open the browsing channel, if it's not already present. The browsing channel can be closed with the <b>CLOSE</b> command when necessary, and it will not automatically close the regular AVRCP link. The browsing channel can be used immediately after it has opened.</p> <p><b>Parameters:</b></p> <p style="text-align: center;">None</p>
<b>LIST</b>	<p>Lists items in a given scope. The start_item and end_item parameters are used to retrieve the folder contents in chunks, if all items do not fit inside a single packet.</p> <p><b>Parameters:</b></p> <p><b>{scope}</b></p> <p style="text-align: center;">0: Media player list 1: Virtual filesystem 2: Search results 3: Now playing list</p> <p><b>{start_item}</b></p> <p style="text-align: center;">The running number of the first item inside the folder to retrieve.</p> <p><b>{end_item}</b></p> <p style="text-align: center;">The running number of the last item to retrieve.</p> <p><b>{number_of_attributes}</b></p> <p style="text-align: center;">Number attributes to retrieve for the items. 0: All attributes ff: No attributes</p> <p><b>[attribute_list]</b></p> <p style="text-align: center;">Sequence of attributes; omit if number_of_attributes is 0 or ff. The supported values for attribute IDs are listed in the table in chapter 3.6.3.2.</p>
<b>SETPLAYER</b>	<p>Selects the player to browse. A Target may have multiple players, so the player must be selected prior to browsing its virtual filesystem.</p> <p><b>Parameters:</b></p>

	<p><b>{ID}</b></p> <p>16-bit hexadecimal ID of the player</p>
<b>PATH</b>	<p>Navigate up or down inside the virtual folder structure.</p> <p><b>Parameters:</b></p> <p><b>{UID_counter}</b></p> <p>UID counter – see the description in chapter 3.5</p> <p><b>{direction}</b></p> <p>0: up – folder UID is omitted, as it’s implicitly known 1: down</p> <p><b>[folder_UID]</b></p> <p>UID of the target folder (omitted if direction is 0)</p>
<b>GET</b>	<p>Get attributes of an item</p> <p><b>Parameters:</b></p> <p><b>{scope}</b></p> <p>0: Media player list 1: Virtual filesystem 2: Search results 3: Now playing list</p> <p><b>{UID_counter}</b></p> <p>UID counter – see the description in chapter 3.5</p> <p><b>{UID}</b></p> <p>UID of the item</p> <p><b>{number_of_attributes}</b></p> <p>Number attributes to retrieve for the items. 0: All attributes ff: No attributes</p> <p><b>[attribute_list]</b></p> <p>Sequence of attributes; omit if number_of_attributes is 0 or ff. The supported values for attribute IDs are listed in the table in chapter 3.6.3.2.</p>
<b>SEARCH</b>	<p>Search for all items containing the search string. No wildcards are accepted. The search results will appear in the search results scope.</p> <p><b>Parameters:</b></p> <p><b>{“search_string”}</b></p> <p>String enclosed in quotes.</p>

### 3.6.4.2 Response data format

The only response for **AVRCP BROWSE OPEN** is the browsing channel opening or failing to open.

The response for **AVRCP BROWSE LIST** contains the current value of the UID Counter and a list of items, which can be of three distinct types: Player, Folder or Media.



For example, an AVRCP BROWSE LIST which asks for items #4, #5 and #6 in the current virtual filesystem folder, and the title and artist attribute (if applicable), response may look like this:

```

@1 AVRCP BROWSE LIST 1 4 6 2 1 2
AVRCP_BROWSE 1 GET_FOLDER_ITEMS UIDC 0042 ITEMS 0004 < FOLDER < UID
0000000000000010 TYPE TITLES PLAYABLE YES NAME 0009 "Favorites" > < FOLDER UID
0000000000000020 TYPE ARTISTS PLAYABLE NO NAME 0007 "Artists" > < MEDIA UID
000000000000003f TYPE AUDIO NAME 000c "SomeSong.mp3" ATTRIBUTES 0002 < < TITLE NAME
0008 "SomeSong" > < ARTIST NAME 000a "SomeArtist" > > >
  
```

The response for **AVRCP BROWSE SETPLAYER** returns the current UID Counter, the count of items in the current folder, and if the current folder is not the root folder (for example, a player might start browsing in a folder called /Artists/Albums/ by default), the count and names of the folders leading to the current folder from the root.

SETPlayer
UIDC uint16
COUNT uint32
FOLDERS uint8
< list of folder names >

Example: select a player with ID 2 as the browsed player. The player has 32 media items, and it is currently in the folder /Playlists/Favorite/

```
@1 AVRCP BROWSE SETPLAYER 2
AVRCP_BROWSE 1 SETPLAYER UIDC 0001 COUNT 00000020 FOLDERS 02 < NAME 0009 "Playlists"
NAME 0008 "Favorite" >
```

The response for **AVRCP BROWSE PATH** returns the count of items in the current folder.

PATH
COUNT uint32

Example: browse down into a folder with UID 0x0000000000000022, when the last received UID Counter value was 0x0042.

```
@1 AVRCP BROWSE PATH 42 1 22
AVRCP_BROWSE 1 SETPLAYER UIDC 0001 COUNT 00000020 FOLDERS 02 < NAME 0009 "Playlists"
NAME 0008 "Favorite" >
```

The response for **AVRCP BROWSE GET** returns a list of attributes.

GET
ATTRIBUTES uint16
< list of attributes >

Example: get all attributes for an item with UID 0x33 in the virtual filesystem, last received UID Counter value 0x0042.

```
@1 AVRCP BROWSE GET 1 42 33 0
AVRCP_BROWSE 1 GET ATTRIBUTES 0003 < < TITLE NAME 0004 "Song" > < ARTIST NAME 0006
"Artist" > < ALBUM NAME 0005 "Album" > >
```

The response for **AVRCP BROWSE SEARCH** returns the current UID Counter value and the number of search results that are now in the Search Results scope.

SEARCH
UIDC uint16 COUNT uint32

Example: search for all items containing the string “abcd”; 3 items match. More information about the matching items can be retrieved with AVRCP BROWSE LIST in the search results scope.

```
@1 AVRCP BROWSE SEARCH "abcd"
AVRCP_BROWSE 1 SEARCH UIDC 0042 COUNT 0003
```

### 3.6.4.3 Errors

Every browsing command with the exception of AVRCP BROWSE OPEN can return an error code, or in the case of AVRCP BROWSE SEARCH, a General Reject, if the Target does not support searching, which is an optional feature.

```
@1 AVRCP BROWSE SEARCH "abcd"
AVRCP_BROWSE 1 GENERAL_REJECT 0f
```

If a command fails for some other reason, for instance if the UIDs have changed on the Target, which happens when media files are modified on, or added to or removed from, the Target’s filesystem.

```
@1 AVRCP BROWSE GET 1 42 33 0
AVRCP_BROWSE 1 GET ERROR 05
```

The error codes are listed in the following table:

Status code	Explanation
0x00	Invalid command – the Target did not understand the command
0x01	Invalid parameter – the command was recognized, but one or more of the parameters were not
0x02	Parameter content error – a parameter was recognized, but contained invalid content, for example the Controller tried to change a supported setting to an unsupported value
0x03	Internal error – the command was understood, but could not be completed due to some internal condition

<b>0x04</b>	Success – this value will never be reported in an error
<b>0x05</b>	UIDs changed – the UID Counter issued by the Controller did not match Target's UID Counter number. The Controller should refresh its cached UID.
<b>0x06</b>	The command ID is reserved
<b>0x07</b>	Invalid direction – applicable only for AVRCP BROWSE PATH
<b>0x08</b>	Not a directory – applicable only for AVRCP BROWSE PATH
<b>0x09</b>	Does not exist – no such media item or folder found with requested UID
<b>0x0a</b>	Invalid scope
<b>0x0b</b>	Out of bounds – a parameter was too large, for example the AVRCP BROWSE LIST end_item parameter was larger than the number of items in the folder
<b>0x0c</b>	Item not playable – the Controller attempted to play a folder that is not playable
<b>0x0d</b>	Media in use – some media cannot be played if they are in use, for example a CD or DVD
<b>0x0e</b>	Now playing list full – cannot add any more items to the now playing list
<b>0x0f</b>	Search not supported – search is an optional feature, and all Targets may not have implemented it
<b>0x10</b>	Search in progress – cannot start another search before the previous search is finished
<b>0x11</b>	Invalid player ID – applicable only for AVRCP BROWSE SETPLAYER
<b>0x12</b>	Player not browseable – while the Target supports browsing, the selected player cannot be browsed
<b>0x13</b>	Player not addressed – the Target has multiple players, and does not know which player should receive the command; the Controller should use AVRCP PDU 60 to select the addressed player
<b>0x14</b>	No valid search results
<b>0x15</b>	No available players – no player can complete the request at the moment
<b>0x16</b>	Addressed player changed – the addressed player has changed for some internal reason, such as the user switching off a media player and starting another on the Target



### 3.6.5 AVRCP {raw} detail

The “**AVRCP {raw}**” command is used to send raw AVRCP press/release codes instead of using the shortcut commands such as “**AVRCP PLAY**” or “**AVRCP FAST\_FORWARD**” to accomplish the same goals. The shortcut commands are, in fact, simply automated press/release combinations of each command code. It is possible to achieve the same functionality with two unique {raw} commands, and this also gives you more control over functions which benefit from more precisely timed “press” and “release” actions, such as rewind and fast-forward.

The list of supported AVRCP command codes and their meanings are in the table below. Remember that **you must send a “release” command to correspond with every “press” command in order to ensure proper functionality.**

Each command code has a 7-bit value, ranging between 0x00 and 0x7F. The 8<sup>th</sup> bit (0x80) is a mask which is used to indicate whether it is a “press” code (8<sup>th</sup> bit cleared) or a “release” code (8<sup>th</sup> bit is set).

**Note:**

Not all devices support all command code. For example, iOS devices do not support remote volume control using **0x41**, **0x42**, or **0x43** commands. This is a limitation of iOS and not an iWRAP firmware problem.

Press code (hex)	Release code (hex)	Description
00	80	Select
01	81	D-pad up
02	82	D-pad down
03	83	D-pad left
04	84	D-pad right
05	85	D-pad up/right
06	86	D-pad down/right
07	87	D-pad up/left
08	88	D-pad down/left
09	89	Root menu
0A	8A	Setup menu
0B	8B	Contents menu
0C	8C	Favourites menu
0D	8D	Exit
<i>0x0E – 0x1F reserved</i>		
20	A0	0 number entry
21	A1	1 number entry

<b>22</b>	<b>A2</b>	2 number entry
<b>23</b>	<b>A3</b>	3 number entry
<b>24</b>	<b>A4</b>	4 number entry
<b>25</b>	<b>A5</b>	5 number entry
<b>26</b>	<b>A6</b>	6 number entry
<b>27</b>	<b>A7</b>	7 number entry
<b>28</b>	<b>A8</b>	8 number entry
<b>29</b>	<b>A9</b>	9 number entry
<b>2A</b>	<b>AA</b>	Dot ( . ) entry
<b>2B</b>	<b>AB</b>	Enter
<b>2C</b>	<b>AC</b>	Clear
<i>0x2D – 0x2F reserved</i>		
<b>30</b>	<b>B0</b>	Channel up
<b>31</b>	<b>B1</b>	Channel down
<b>32</b>	<b>B2</b>	Previous channel
<b>33</b>	<b>B3</b>	Sound select
<b>34</b>	<b>B4</b>	Input select
<b>35</b>	<b>B5</b>	Display information
<b>36</b>	<b>B6</b>	Help
<b>37</b>	<b>B7</b>	Page up
<b>38</b>	<b>B8</b>	Page down
<i>0x39 – 0x3F reserved</i>		
<b>40</b>	<b>C0</b>	Power
<b>41</b>	<b>C1</b>	Volume up
<b>42</b>	<b>C2</b>	Volume down
<b>43</b>	<b>C3</b>	Mute
<b>44</b>	<b>C4</b>	Play

45	C5	Stop
46	C6	Pause
47	C7	Record
48	C8	Rewind
49	C9	Fast-forward
4A	CA	Eject
4B	CB	Forward (next track)
4C	CC	Backward (previous track)
<i>0x4D – 0x4F reserved</i>		
50	D0	Angle
51	D1	Subtitle
<i>0x52 – 0x70 reserved</i>		
71	F1	F1 function key
72	F2	F2 function key
73	F3	F3 function key
74	F4	F4 function key
75	F5	F5 function key

Below is an example of using AVRCP {raw} to pause playback:

**AVRCP 46**  
**AVRCP C6**  
A2DP STREAMING STOP 0

Below is an example of using AVRCP {raw} to rewind the track position:

**AVRCP 48**  
*[track rewinds until next command is sent, specific behavior depends on media device]*  
**AVRCP C8**

### 3.6.6 iWRAP commands related to audio

<b>PLAY</b>	-	Command <b>PLAY</b> is used to generate tones or beeps.
<b>VOLUME</b>	-	Changes the volume level (audio output gain).
<b>SET CONTROL AUDIO</b>	-	This command controls the physical interface routing of audio on WT32. The command is also used to enable or disable multiple SCO support.
<b>SET CONTROL CODEC</b>	-	This command controls the preference of A2DP audio codecs, channel modes and sampling rates for A2DP.
<b>SET CONTROL GAIN</b>	-	<b>SET CONTROL GAIN</b> is used to control the internal input and output gain.
<b>SET CONTROL MICBIAS</b>	-	<b>SET CONTROL MICBIAS</b> controls the linear regulator that drives current through the dedicated mic bias pin.
<b>SET CONTROL PCM</b>	-	This command configures the physical PCM hardware interface settings and PCM data format.
<b>SET CONTROL RINGTONE</b>	-	Configures a ring tone which is used with HFP or HSP profile if the Audio Gateway does not provide an in-band ring tone.
<b>SET CONTROL PREAMP</b>	-	Enables or disables the 20dB microphone pre-amplifier.

Please refer to iWRAP user guide for exact documentation of the listed commands.

**NOTE:**

The pre-amplifier **MUST** be turned off in A2DP source device to prevent the low frequencies being cut out.

## 4 Power saving

iWRAP offers two power saving options. Sniff mode, which can be used to save power for active *Bluetooth* connections and deep sleep mode which puts the internal processor into a reduced duty cycle mode. Please refer to iWRAP user guide for more information about sniff and deep sleep modes. One should be very careful when using sniff mode with an active A2DP connection. Sniff mode will reduce the data transmission rate and therefore affect the robustness of an audio link. When sniff mode is used with A2DP audio “clipping” is more likely to occur than without power saving.

One should also know that when *Bluetooth* connections are in active mode i.e. no power saving in use the master device uses 3-4 times less power than a slave device. Therefore for battery powered applications it might be useful to configure the device as a master rather than a slave.

## 5 Audio quality improvement

The sub-band coding (SBC) used to encode or decode the A2DP audio is a lossy audio codec and some of the audio data is lost when it's transmitted over a *Bluetooth* link. For some applications like high end headsets and headphones or audio amplifiers this may not be acceptable.

Alternative audio codecs can be used to encode or decode the audio stream and provide better quality or lower latency *Bluetooth* audio. Such codes are for example MP3, AAC or aptX. MP3 codec is however not generally supported by *Bluetooth* enabled cell phones, tablets and PCs. AAC codec is supported in the latest generation of Apple devices. aptX codec on the other hand has received the widest market adoption and is supported by my many cell phones, tablets, PCs and headsets.

AAC codec support is available by default from iWRAP6 onwards in all firmware builds (and also upon request for iWRAP5 audio firmware builds). However, the usage of the codec in the end product requires a separate license and it has an extra license fee. For more information, please see <http://www.vialicensing.com/licensing/aac-fees.aspx>.

aptX codec provides a nearly lossless audio quality and therefore is ideal to be used in Hi-Fi audio applications and products where high quality *Bluetooth* audio is needed. iWRAP supports aptX codec, but because it's a 3<sup>rd</sup> party codec and has an extra license fee, it's not included in the generic firmware builds. For more information about aptX codec, please contact [sales@bluegiga.com](mailto:sales@bluegiga.com).

### 5.1 Enabling AAC Audio Codec for A2DP

One of the optional audio codecs with Bluetooth A2DP is AAC. AAC is supported at the moment by Apple iOS devices and provides better audio quality compared to the regular SBC codec. WT32i supports the AAC audio codec and it can be enabled with the following iWRAP commands.

```
SET CONTROL CODEC AAC JOINT_STEREO 44100 0
SET CONTROL CODEC SBC JOINT_STEREO 44100 1
```

The first command enables the AAC codec in stereo mode and 44.1kHz sampling rate and priority 0 (highest). The second command enables the mandatory SBC codec in stereo mode and also 44.1kHz sampling rate and priority 1 (lower). With this configuration AAC is the preferred codec, but if the remote device does not support it SBC will be used instead.

**Note:** iWRAP contains AAC technology which incorporates intellectual property owned by numerous third parties. Supply of this product does not convey a license under the relevant intellectual property of those third parties nor imply any right to use this product in any finished end user or ready-to-use final product. An independent license for such use is required. For details, please visit <http://www.vialicensing.com>.

### 5.2 Enabling aptX Audio Codec for A2DP

aptX is another optional audio codec for Bluetooth A2DP. aptX is supported by newer Android devices and it also provides better quality audio compared to the regular SBC codec. WT32i. The aptX audio codec can be enabled in special builds with the following iWRAP commands.

```
SET CONTROL CODEC APTX JOINT_STEREO 44100 0
SET CONTROL CODEC SBC JOINT_STEREO 44100 1
```

**Note:** aptX requires a special license and is only included in a separate iWRAP build. To enable it, you need to flash the aptX build and add a valid license key. Please contact Bluegiga technical support (<http://www.bluegiga.com/support>) for further information on how to obtain a license key.

### 5.3 Routing the A2DP Audio to I2S (External Codec)

iWRAP6 adds basic support for configuring an external I2S audio codec without requiring an external MCU. The commands controlling this are “**SET CONTROL AUDIO**” and “**SET CONTROL EXTCODEC**”.

The WT32i development kit contains an external I2S audio codec (Texas Instruments TLV320AIC32IRHB), which connects to the WT32i’s I2S interface as well the I2C interface. To try out the external codec, the following iWRAP configurations are needed.

```
SET CONTROL AUDIO INTERNAL I2S EVENT 32
```

```
SET CONTROL EXTCODEC PRE 30 18 0200918000008A103700010000800000FF07C7C787C78 30 04  
25C01400 30 0a 28C00000000000008000 30 03 330F00 30 05 3F00800F00 30 04 6500A200
```

```
RESET
```

The first command routes the A2DP audio to the I2S interface instead of the DAC and configures I2S to use 32 bits per sample to suit the clocking requirements of the TI codec. The second command contains the codec configuration parameters that iWRAP send to the TI codec over I2C interface when A2DP streaming is enabled or disabled.

## 6 References

- [1] The *Bluetooth* SIG, A2DP Profile overview, URL:  
<http://Bluetooth.com/Bluetooth/Technology/Works/A2DP.htm>
  
- [2] The *Bluetooth* SIG, AVRCP Profile overview, URL:  
<http://Bluetooth.com/Bluetooth/Technology/Works/AVRCP.htm>



## 7 Contact Information

**Sales:** [sales@bluegiga.com](mailto:sales@bluegiga.com)

**Technical support:** <http://www.bluegiga.com/support>

**Orders:** [orders@bluegiga.com](mailto:orders@bluegiga.com)

**WWW:** [www.bluegiga.com](http://www.bluegiga.com)

### **Head Office / Finland:**

Phone: +358-9-4355 060

Fax: +358-9-4355 0660

Sinikalliontie 5A

02630 ESPOO

FINLAND

### **Postal address / Finland:**

P.O. BOX 120

02631 ESPOO

FINLAND

### **Sales Office / USA:**

Phone: +1 770 291 2181

Fax: +1 770 291 2183

Bluegiga Technologies, Inc.

3235 Satellite Boulevard, Building 400, Suite 300

Duluth, GA, 30096, USA

### **Sales Office / Hong-Kong:**

Bluegiga Technologies, Ltd.

Phone: +852 3182 7321