AUDIOAGENT (AA) – IDC777
UART COMMAND MANUAL

Ref: AAGENT-IDC777-V300x

Please contact info@iot747.com for any additional information. This document is Confidential and may contain trade secrets. It is not to be distributed, used, copied, reproduced, or modified in whole or in part, nor its contents revealed in any manner to others without the written permission of IOT747.

General Notices

IOT747 products are not authorised for use in life-support or safety-critical applications. Use in such applications is done at the sole discretion of the customer. IOT747 will not warrant the use of its devices in such applications.

While every care has been taken to ensure the accuracy of the contents of this document, IOT747 cannot accept responsibility for any errors. IOT747 reserves the right to make modifications, corrections and any other changes to its products at any time. Customers should obtain the latest information before placing orders.

IOT747 other products, services and names used in this document may have been trademarked by their respective owners. The publication of this information does not imply that any license is granted under any patent or other rights owned by IOT747.

Refer to www.iot747.com for more information. IOT747® is a trading name for Company Deep Limited.

Version History

Version	Date	Comments
1.0	19/7/2024	Initial Version

Table of Contents

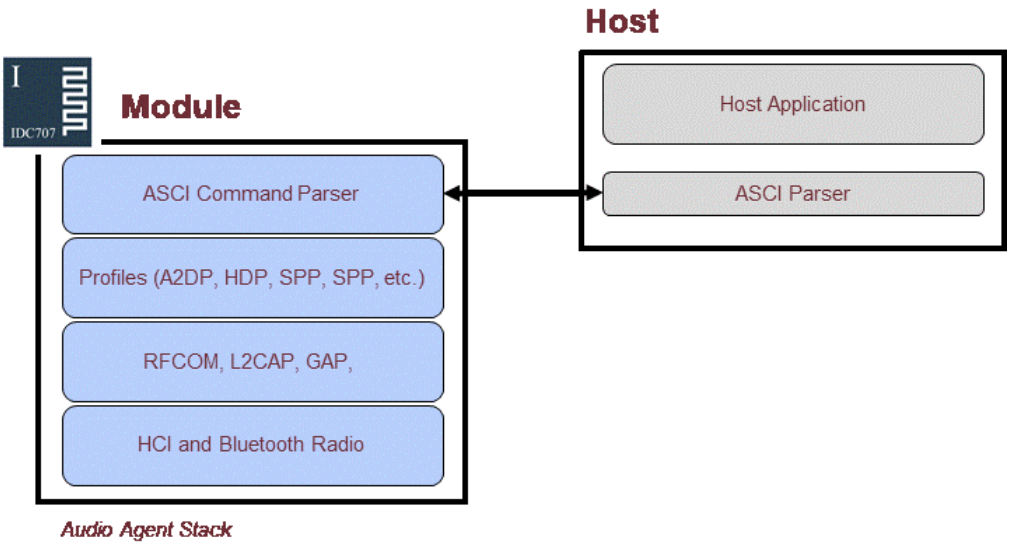
Version History 3
Introduction 5
Upgrading 6
Getting Started 7
First Commands 10
General Operation 12
Link ID Management 14
AudioAgent Commands 16
AudioAgent Configuration 60
AudioAgent Notifications 83
GPIO Functionality 94
LED Indications 95
Restoring the Default Configuration 96
Communication with Apps 97
Echo Cancellation - CVC 98
AT Commands 99
BAP QOS Configurations 103
Error codes 107
Examples 109
Wireless Audio Latencies 106
Thermistor Table 113
Terms and definitions 114

Introduction

AudioAgent is an embedded firmware running entirely on the IDC7x7 modules (ex: IDC777). It implements the Bluetooth protocol stack and many Bluetooth profiles as well. All software layers, including application software, Codecs such as aptX, AAC and aptX HD, aptX Adaptive run on the module in a protected user software execution environment.

The host system can interface to AudioAgent through one or more physical interfaces. The most common interfacing is done through the UART interface by using the ASCII commands that AudioAgent 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 hardware used.

The user can write application code to the host processor to control AudioAgent using ASCII commands or GPIO events. In this way, it is easy to develop Bluetooth enabled applications. On the IDC7x7 modules, there is also DSP processor for Audio processing.



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

1) If the host system has a processor, software can be used to control AudioAgent by using ASCII based commands or GPIO events.

2) If there is no need to control AudioAgent or the host system does not need a processor, AudioAgent can be configured to be transparent and autonomous, in which case it only accepts Bluetooth connections or automatically opens them.

3) GPIO lines that IOT747 Bluetooth modules offer can also be used together with AudioAgent to achieve additional functionality, such as Connection Detection. Audio interfaces can be used to transmit audio over a Bluetooth link.

Upgrading

For those who are starting to design and using a IDC777 Diskit, we suggest upgrading the board to the latest version of AudioAgent.

The IDC747 can be upgraded:

1) Over Bluetooth (OTA) using an Android or iOS App
2) Over the UART interface using the same Serial Interface used to control the module
3) Over the USB interface

For UART and USB, Windows tools that are available on iot747.com. For OTA, demo Apps are available on Google Play and the App Store. Source Reference Code is available on iot747.com for integration of the upgrade mechanism into User's host iOS, Android or Windows Host Software.

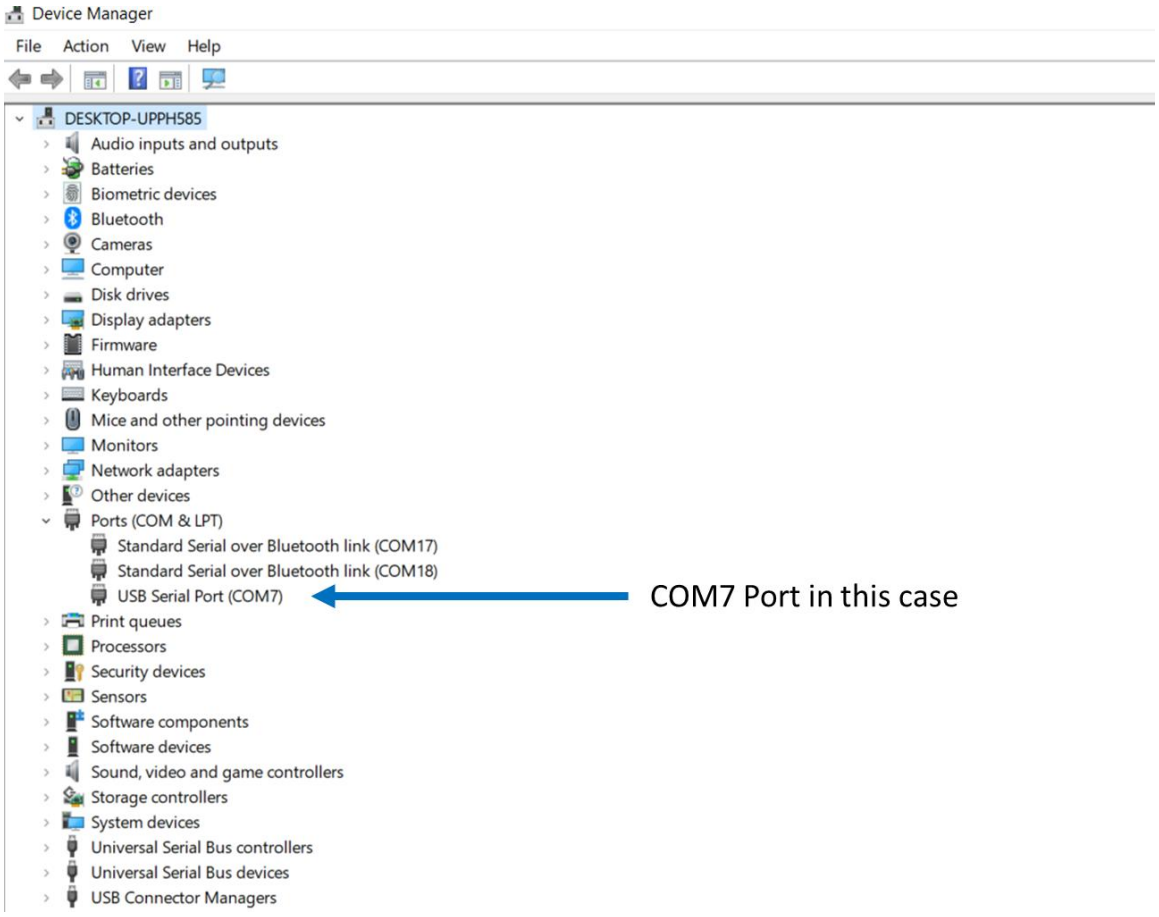
Please contact www.iot747.com on the Support or on the Forum for upgrade tool and firmware upgrades.

Getting Started

To start using AudioAgent, you need:

- A Development board (IDC777-DISKIT)
- A Computer running a serial terminal software such as Hercules.

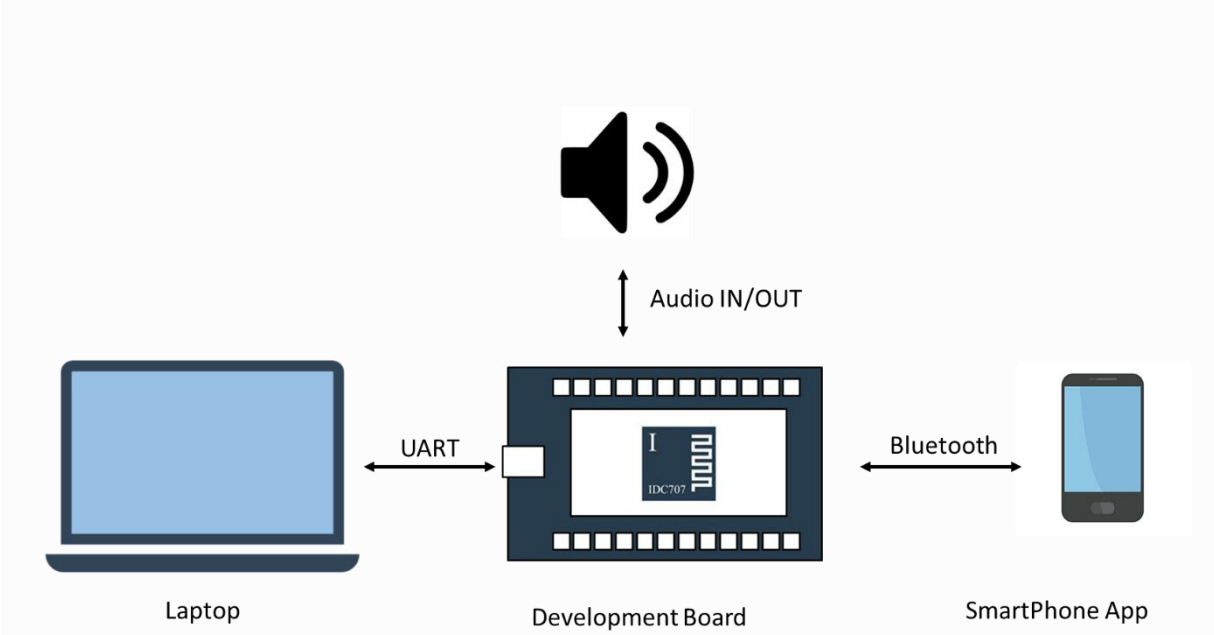
Connect the Computer (or Laptop) to the Development board using the USB cable. The Development board should enumerate as a COM port. You can see the COM number by going to Device Manager on a Windows Laptop.



By default, AudioAgent uses the following UART settings:

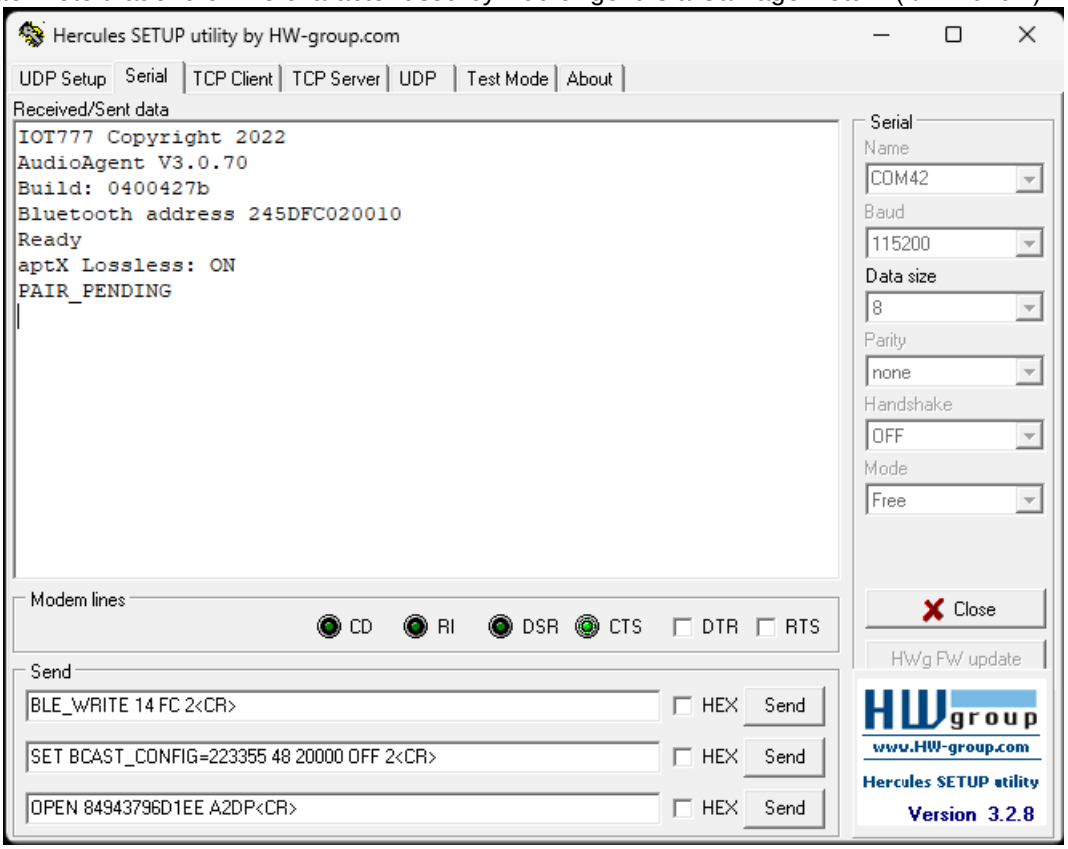
Baud rate	9600 bps
Data bits	8
Stop bits	1
Parity bit	No Parity

HW Flow Control	Disabled



Example standard configuration

Once you have configured your serial terminal and opened the COM port, you should see a prompt appear on the screen of the terminal. If you see a prompt and a “Ready”, the module is ready to operate. Note that end of line character used by AudioAgent is a Carriage Return (‘\r’ or 0x0D).



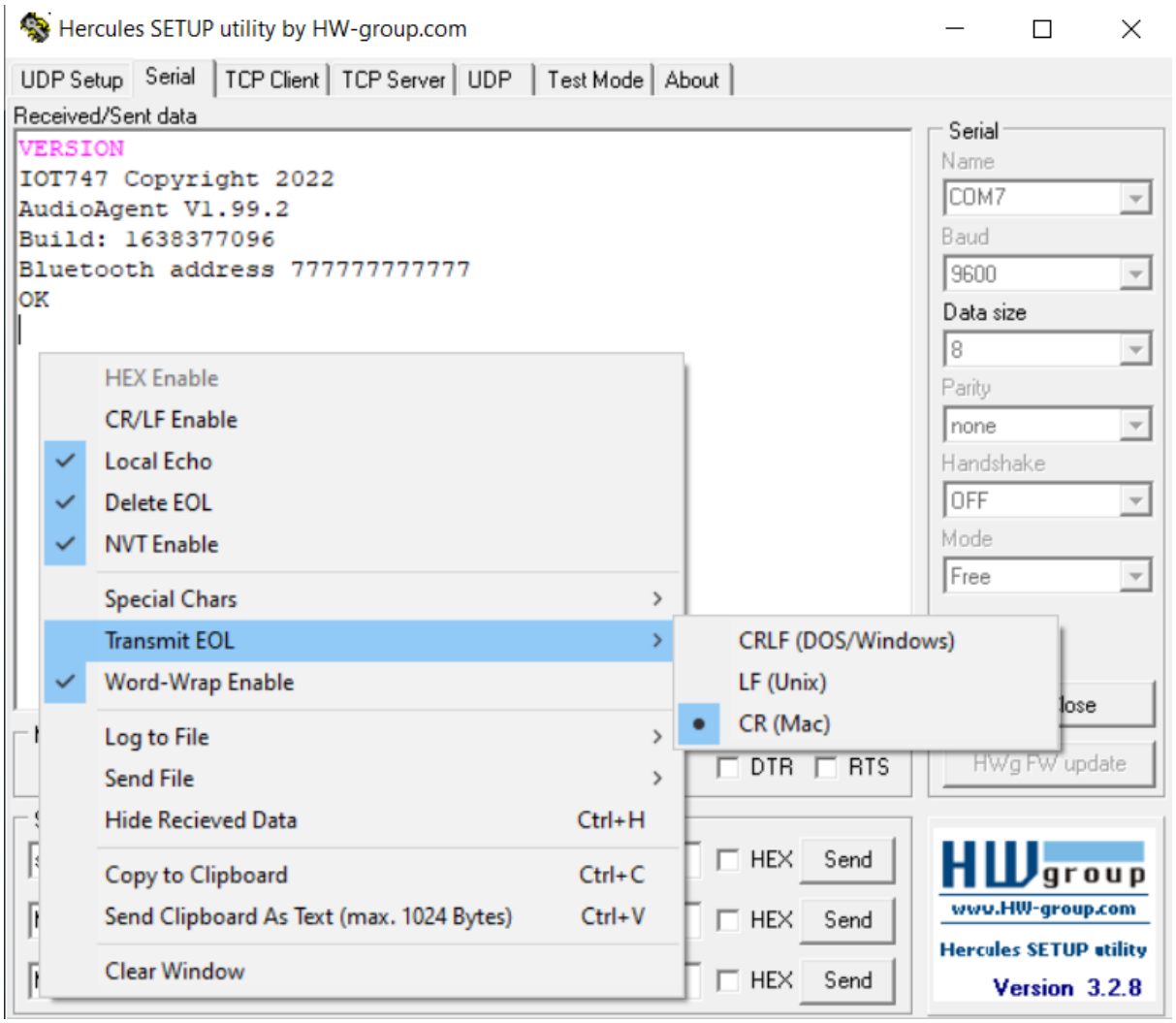
AudioAgent initial prompt

If you do not see the prompt and “Ready” appear, please check that:

- The module is powered ON and receiving power
- The UART settings on your serial terminal are the same as above
- You have opened the right COM port

Default behaviour

When AudioAgent boots with factory default settings, it is discoverable. Any Bluetooth enabled phone or other Bluetooth device can see it when scanning (discovering) other Bluetooth devices. AudioAgent will appear as ‘AA-XXXXXX’ where XXXXXX represent the last six hexadecimal digits of the device’s Bluetooth address.



AudioAgent UART Settings on Hercules

First Commands

A few basic AudioAgent use cases are below to get you started. More details and explanation are later in the manual.

The RESET command resets the module. The response will be the prompt at power ON

```
RESET
IOT777 Copyright 2022
AudioAgent V3.0.70
Build: 0400427b
Bluetooth address 245DFC020010
Ready
aptX Lossless: ON
PAIR_PENDING
```

The STATUS command gives the status of the module and whether it is connected to a device or not.

```
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[ON] ADVERTISING[ON] SCAN_UNI[OFF]
OK
```

The HELP command lists the commands available by the module. Please note that some commands are only available for certain modules. Below is just an example and you might not have all these commands available on your module.

```
HELP
ADVERTISING
AT
AVRCP_META_DATA
BATTERY_STATUS
CLOSE
…
UNPAIR
VERSION
VOLUME
WRITE
OK
```

The CONFIG command displays the configuration settings of the module. Please note that some configurations are only available for certain modules. Below is an example and you might not have all the configurations below on your module.

```
CONFIG
AUDIO=0
AUDIO_ANALOG=44100 15 10 ON OFF
AUDIO_DIGITAL=0 44100 64 100A00
AUTOCONN=0
…
BATT_CONFIG=OFF 145 4250 1500 150
IOT_SMART_UUIDS=68E3 28F0 89F7 D93C
UART_CONFIG=9600 OFF 0
OK
```

The RESTORE command resets and restores the module to factory configuration settings.

```
RESTORE
IOT777 Copyright 2022
AudioAgent V3.0.70
Build: 0400427b
Bluetooth address 245DFC020010
Ready
aptX Lossless: ON
PAIR_PENDING
```

General Operation

AudioAgent has two operating modes that define how data coming from the host (UART) is processed.

Command Mode:

In most cases and most applications, AudioAgent will be ready to receive commands over the UART interface. It then interprets them and also sends notifications over the UART in case there is activity in the Bluetooth link.

1) Commands: The host can send commands over the UART. AudioAgent will answer with a Notification. Please see the Command section for the list of commands that AudioAgent supports.

2) Notifications: When a command was issued or when there is activity over the Bluetooth Link (such as a Bluetooth connection, disconnection, etc.), AudioAgent will send a Notification over the UART. The Notifications are to notify the host of the activity or respond to the command. Please see the Notification section for the list of Notifications that can be received over UART.

3) Configurations: AudioAgent comes with Configuration parameters that affect the behaviour of the module given different circumstances or to enable/disable certain features. Please see the Configuration section for the list of the Configurations that AudioAgent supports.

Data Mode:

Data mode is used to transfer data between the host and a remote device without using commands. In this case, the module will take any Data coming from the UART and directly transfer it to the Bluetooth (BLE or SPP) link. This is used when the Host wants to transmit or receive large files through the Bluetooth Link.

To switch to data mode on an established connection, use the ENTER_DATA_MODE command with the BLE, or SPP link ID. In data mode, data is transferred seamlessly over the specified link between the host and the remote device. The host interface is exclusively dedicated to transferring data—the only command AudioAgent will parse is the escape sequence ($$$) which switches AudioAgent back to Command mode.

Additional notes:

- AudioAgent returns automatically to Command mode if a disconnection occurs with the Data mode link.
- GPIOs can be used to enter or exit Data mode (see GPIO Functionality).
- GPIOs are raised upon specific events (GPIO Functionality).

Switching between command mode and Data Mode:

To switch between the Data and Command modes, use the following commands:

Command	Switch From
$$$$	Data Mode→Command Mode
ENTER_DATA_MODE <link_id>	Command Mode→Data Mode

Link ID Management

Since AudioAgent allows to connect multiple profiles to multiple devices, it needs to identify particular profiles/links for some commands. This is done by Links. A Link ID is an 8-bit hexadecimal value 0xAB. In most cases, the first digit identifies a device and the second digit refers to a profile as described in the tables bellow.

Device field values

Device A	Description
0	Reserved values
1	Device 1
2	Device 2
3	Device 3

Profile field values

Profile (B)	Description
0	Advanced Audio Distribution Profile (A2DP)
1	Audio/Video Remote Control Profile (AVRCP)
2	Hands-Free Profile (AGHFP), audio gateway
3	Hands-Free Profile (HFP), hand-free device
4	Bluetooth Low Energy (BLE)
5	Serial Protocol Profile (SPP)
6	Unicast (CIS) Rx Music
7	Unicast (CIS) Rx Voice
8	Unicast (CIS) Tx Music
9	Unicast (CIS) Tx Voice
A	Auracast (BIS) Rx Channel1
B	Auracast (BIS) Rx Channel2
C	Auracast (BIS) Tx Channel1

Example:

In the example below, we have connected a mobile phone to the module. We send the command STATUS and we receive a response over UART. It is connected to 1 Device with three profiles LINK10, LINK11 and LINK13. These are A2DP, AVRCP and HFP profiles.

```
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[IDLE] ADVERTISING[ON] SCAN_UNI[OFF]
LINK 10 CONNECTED A2DP 30D875EB8C1C SUSPENDED SBC SNK 44100
LINK 11 CONNECTED AVRCP 30D875EB8C1C STOPPED
LINK 13 CONNECTED HFP 30D875EB8C1C IDLE
OK
```

Please note that the Bluetooth Low Energy (LINK 14 or LINK 24) can be considered a different device depending on the phone. So if you have one mobile phone connected over Bluetooth Low Energy and Classic Bluetooth (For example A2DP, AVRC and HFP like the example above), it will appear as LINK10,11,13 and LINK 24 (for Bluetooth Low Energy).

AudioAgent Commands

AudioAgent can be used and controlled from the host system by sending ASCII commands through the UART interface of the module.

This section explains the AudioAgent commands and their syntax. Some simple use cases and tips are also given.

The generic syntax for commands is:

COMMAND <parameter_1> <parameter_2> … <parameter_n>\r

with a space between each parameter and a Carriage Return (‘r’ or 0x0D) at the end of each command. The different commands to control the Bluetooth link are listed in alphabetical order below. Mandatory parameters are listed in “( )” optional parameters are listed in “[ ]”.

The maximum length for a command is 150 characters, if a command larger than this is provided, AudioAgent will return an error.

Please note that:

- The parser is case sensitive so the commands are in capital letters
- Commands must end with a Carriage Return (‘\r’ or 0x0D)
- AudioAgent prints OK to indicate that the command has been executed
- It is recommended to introduce a 150ms delay between commands (if possible)
- It is recommended to wait for a Command confirmation before issuing the next

Table1 below lists all the available commands and a general description of each command.

Table2 describes each command in details with examples.

Table 1: List of Commands

Command	General Description
ADVERTISING	Starts/Stops Advertising (BLE)
AVRCP_META_DATA	Requests Music Information Data (Album, Artist, etc.)
AT	Send AT Command/Response over HFP or AGHFP
BATTERY_STATUS	Sends Battery and Charger Status
BLE_GET_CHAR	BLE Central: GATT Characteristic Discovery
BCAST_ASSIST	IDC777 plays the role of a Phone and assists LE Audio Headsets
BCAST_REPORT	LE Auracast: Give Information on BIS and Subgroups
BLE_GET_DES	BLE Central: GATT Descriptors Discovery
BLE_GET_SERV	BLE Central: GATT Service Discovery
BLE_INDIC	BLE Central: Turn ON Indications
BLE_INDICATION	BLE Peripheral: GATT Indication Request
BLE_NOTIF	BLE Central: Turn ON Notifications
BLE_NOTIFICATION	BLE Peripheral: GATT Notification Request
BLE_READ	BLE Central: GATT Read Request
BLE_READ_RES	BLE_Peripheral: GATT Read Response
BLE_SET_DB	BLE Peripheral: Set Custom GATT Database
BLE_WRITE	BLE_Central: GATT Write Request
BROADCAST	Starts Broadcast Transmission (only IDC777/767)
BROADCODE	Sets the PIN code for Auracast (Only IDC777767)
CALL	Manages Voice Call
CLOSE	Connection Close Request
CONFIG	Shows Configuration registers
CONNECTABLE	Starts/Stops making the module Connectable
DISCOVERABLE	Starts/Stops Pairing Mode (Classic Bluetooth).
ENTER_DATA_MODE	Enter Data Mode

Command	General Description
HELP	Shows List of Commands (LC)
GET	Shows the value of a config parameter
INQUIRY	Search for Bluetooth Classic Devices
LIST	Shows List of Paired Devices (LPD)
MICBIAS	Controls the MIC_BIAS Pin on the module
MUSIC	Controls the Music Stream
NAME	Get Remote Device Name
OPEN	Connection Open Request
PAIR	Pair Request
PASSKEY	Pairing User Confirmation
PIO	Set GPIO State
POWER	Power ON/OFF module
PROMPT	Plays prerecorded Prompts
RESET	Resets module
RESTORE	Restores Default Configuration parameters
ROLE	Controls or Inquires about Classic Bluetooth Role (Master/Slave)
ROUTE	Audio Routing for live and test cases
RSSI	Get the RSSI for the ACL with given remote device
QUALITY	Obtain the ACL rx link quality
SCAN	Scan for BLE Devices
SCAN_UNI	Continuous Scan so Unicast Receivers can connect
SCO_OPEN	Opens a Classic SCO Audio Channel
SCO_CLOSE	Closes a Classic SCO Audio Channel
SEND	Sends Data
SEND_RAW	Send Raw Data

Command	General Description
SET	Sets configuration parameter.
SRRD	Set BLE Advertising Data
STATUS	Returns the Bluetooth state of the device
UNPAIR	Removes Devices from Pairing List (LPD)
TONES	Generate Custom Tones
VOLUME	Controls Volume
VERSION	Shows Firmware Version
WRITE	Store Configurations in Permanent Memory

Table 2: Detailed Description and Example of Commands

Command	Description
ADVERTISING	Starts/Stops Advertising (BLE) Back to Table1: List of Commands

Description: Starts/Stops BLE Advertising

Syntax: ADVERTISING <action>
Notes: The user can check in what state the module is by using the command STATUS.

Response: OK

Parameter(s):
<action>:

- ON – Enable Advertising
- OFF-Disable Advertising

Example(s):

```
ADVERTISING ON
OK
ADVERTISING OFF
OK
```

AVRCP_META_ DATA	Request Music Information Back to Table1: List of Commands

Description: Requests Information on the Music currently playing

Syntax: AVRCP_META_DATA <linkid>

Response: OK

Command	Description

Parameter(s):
<link_id>:

- 8-bit hex value (See Link ID Management)

Example(s):

```
AVRCP_META_DATA 10
OK
AVRCP_MEDIA TITLE: Give Me the Night
AVRCP_MEDIA ARTIST: Rob Hayes
AVRCP_MEDIA ALBUM: Ibiza House Anthems 2022
```

AT	Send AT Command/Resp over HFP Back to Table1: List of Commands

Description: Send an AT Command or response over a specified HFP/AGHFP link. This is useful when the user wants to sell specific (proprietary commands). Please refer to the AT Command section for an example of some standard HFP commands. Also, to receive the AT Notification, these need to be enabled in HFP_CONFIG.

Syntax: AT <link_id> <command>

Response: OK
CALL_REDIAL (or AT …)
OK

Parameter(s):
<link_id>: BLE Link Identifier

- 8-bit hex value (See Link ID Management)

<command>: AT command string

- Valid AT command (ex: AT+VTS=1)

Example(s):

```
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[IDLE] ADVERTISING[ON] SCAN_UNI[OFF]
LINK 10 CONNECTED A2DP 30D875EB8C1C SUSPENDED SBC SNK 44100
LINK 11 CONNECTED AVRCP 30D875EB8C1C PAUSED
LINK 13 CONNECTED HFP 30D875EB8C1C IDLE
OK
>>This starts a call to number 121
AT 13 ATD121;
OK
SCO_OPEN
CALL_ACTIVE HFP 13
SCO_CLOSE
CALL_IDLE HFP 13
```

BATTERY_STATUS	GATT Characteristic Discovery Back to Table1: List of Commands

Description: Sends the status of the battery and the charger.

Command	Description

Syntax: BATTERY_STATUS
Notes: The values will only be read correctly if BATT_CONFIG enables battery, charger and temperature. Note that a thermistor is needed in the hardware design to provide a reading. If this is not the case, a wring reading of the battery will cause the module to shut down for safety as battery temperature is critical. If BATT_CONFIG disables the battery, charger and temperature, a static reading will be provided (Example: Voltage=3300 and Temprature=20).

Response: BATTERY_STATUS <charger_state>
BATTERY_STATUS VOLTAGE <voltage>
BATTERY_STATUS TEMP <temp>
OK

Parameter(s):
<charger_state>:

- CHARGER NOT CONNECTED
- CHARGER CONNECTED CHARGING
- CHARGER CONNECTED NOT CHARGING

<voltage>: Integer

- Battery Instantaneous voltage reading in mV

<temp>: Integer

- Battery Instantaneous temperature reading in degree centigrade

BLE_GET_CHAR	GATT Characteristic Discovery	Back to Table1: List of Commands

Description: Discovers the characteristics of a remote GATT Server. A BLE_CHAR notification is received for each characteristic found.

Syntax: BLE_GET_CHAR <link_id> [<start handle><end_handle>]

Response: PENDING
BLE_CHAR <link_id> <type> <uuid> <handle> <properties>
OK

Parameter(s):
<link_id>: BLE Link Identifier

- 8-bit hex value (See Link ID Management)

<start_handle>: (Start discovery handle)

- 4-digit Hexadecimal format
- e.g. 2A05

<end_handle>: (End discovery handle)

- 4-digit Hexadecimal format
- e.g. 2A05

<type>: UUID type

Command	Description

- U16, U32 or U128

<uuid>: Characteristic UUID

- Hex format. ex: 2A05 (U16), 2A05-79AD (U32)

<handle>: Characteristic Handle

- 4 digit Hex format. ex: 000C

<properties>: Characteristic Properties

- Bit values
- 0x01-Broadcast
- 0x02-Read
- 0x04-Write without Response
- 0x08-Write
- 0x10-Notify
- 0x30-Indicate
- 0x40-Authenticated signed writes
- 0x80-Extended Properties

Example(s):

```
OPEN 245DFC020000 BLE 0
PENDING
OPEN_OK 14 BLE 245DFC020000
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[IDLE] ADVERTISING[ON] SCAN_UNI[OFF]
LINK 14 CONNECTED BLE 245DFC020000 MTU 250
OK
BLE_GET_CHAR 14
BLE_CHAR 14 U16 2A05 0003 20
BLE_CHAR 14 U16 2B29 0006 0A
BLE_CHAR 14 U16 2B2A 0008 02
BLE_CHAR 14 U16 2B3A 000A 02
BLE_CHAR 14 U16 2A00 000D 02
BLE_CHAR 14 U16 2A01 000F 02
BLE_CHAR 14 U128 00001101-D102-11E1-9B23-00025B00A5A5 0012 08
BLE_CHAR 14 U16 2B7E 005F 08
BLE_CHAR 14 U16 2B7F 0061 12
.......
BLE_CHAR 14 U16 2BBF 00E8 02
BLE_CHAR 14 U16 2BC0 00EA 10
BLE_CHAR 14 U16 2BC2 00ED 12
BLE_CHAR 14 U16 2BC1 00F0 12
BLE_CHAR 14 U128 06D1E5E7-79AD-4A71-8FAA-373789F7D93C 00F4 38
BLE_CHAR 14 U128 818AE306-9C5B-448D-B51A-7ADD6A5D314D 00F7 14
OK
BLE_GET_CHAR 14 0008 000B
BLE_CHAR 14 U16 2B3A 000A 02
OK
```

BCAST_ASSIST	IDC777 plays the role of a Auracast Assistant (like a phone)

Command	Description

Description: The IDC777 can play the role of a Smart Phone and assist Paired and Connected Headsets to connect to an Auracast Channel (whether it is the Auracast Channel from IDC777 or from another source). This feature can be used for example to test LE Audio headsets Auracast functionality or incase there is no phone in the scenario.

Syntax: BCAST_ASSIST <bt_address> [<broadcastid><advid>]

Response: BCAST_OK

Note: If the optional parameters are used, the IDC777 directs the Paired Headset (or Earbud) to connect to the Auracast channel with the Broadcast ID and Avdertising ID. If the option parameters are ignored, the IDC777 will direct the Headset (or Earbud) to connects to its own Auracast Stream. Note that for this to work, the IDC777 needs to be Auracasting (ex: BROADCAST ON 1)

Parameter(s):
<link_id>: BLE Link Identifier

- 8-bit hex value (See Link ID Management)

bcast_id>: (Broadcast ID)

XXXXXX (X=1 to F). Note that in production bcast_id should be a random number generated by the host to avoid 2 transmitters having the same broadcast ID

<advid>: (Advertising ID)

- 1 hex number
- e.g. D

Example(s):

```
BCAST_ASSIST 00023CAA0E29
OK
OPEN_OK 14 BLE 00023CAA0E29
BCAST_ASSIST_OK
```

BCAST_REPORT	LE Auracast: Gives information on BIS and Subgroups

Description: Once an Auracast Channel is open (BRX1 or BRX2), this command can be sent to discover how many subgroups and how many BIS are in the Auracast Streams

Syntax: BCAST_REPORT <link_id>

Response:
Subgroups: <num_subgroups>
CODEC: <codec> Num_BIS: <num_bis> Program_Info <metadata> BIS_INDEX: <bis_index> <right/left>
...

OK

Parameter(s):
<link_id>: BLE Link Identifier

- 8-bit hex value (See Link ID Management)

<num_subgroups>: (Number of subgroups)

Command	Description

- 1 to 8
- e.g. 2

<codec>: (Codec used)

- LC3

<num_bis >: (Number of BIS – mono streams)

- 1 or 2
- e.g. 2

<metadata>: (Meta Data – example: Stream name)

- max 16 characters
- e.g: Italian

<bis_index >: (Index to use to play the Stream. See MUSIC command)

- 1 to 4.

<right/left>: (Position of the Mono Audio Stream)

- MONO, FR_RIGHT or FR_LEFT

Example(s):

```
open 245DFC021563 brx1 0 d 223344
PENDING
OPEN_OK 1A BRX1 245DFC021563
bcast_report 1a
PENDING
Subgroups:4
Codec:LC3 Num_BIS:1 Program_Info:Italian BIS_Index:0x1 MONO
Codec:LC3 Num_BIS:1 Program_Info:French BIS_Index:0x2 MONO
Codec:LC3 Num_BIS:1 Program_Info:English BIS_Index:0x4 MONO
Codec:LC3 Num_BIS:1 Program_Info:German BIS_Index:0x8 MONO
OK
music 1a play 1
OK
LC3_BROADCAST_NUMBER_BIS 1
LC3_BROADCAST_STREAM_CONNECT
music 1a pause
OK
LC3_BROADCAST_STREAM_DISCONNECT
music 1a play 2
OK
music 1a pause
OK
```

BLE_GET_DES	BLE Central: GATT Characteristic Discovery

Description: Discovers the descriptors of a remote GATT Server. A BLE_DES notification is received for each characteristic found.

Syntax: BLE_GET_CHAR <link_id> <start handle><end_handle>

Command	Description

Response: PENDING
BLE_DES <link_id> <type> <uuid> <handle> <properties>
OK

Parameter(s):

<link_id>: BLE Link Identifier

- 8-bit hex value (See Link ID Management)

<start_handle>: (Start discovery handle)

- 4-digit Hexadecimal format
- e.g. 2A05

<end_handle>: (End discovery handle)

- 4-digit Hexadecimal format
- e.g. 2A05

<type>: UUID type

- U16, U32 or U128

<uuid>: Characteristic UUID

- Hex format. ex: 2A05 (U16), 2A05-79AD (U32)

<handle>: Characteristic Handle

- 4 digit Hex format. ex: 000C

<properties>: Characteristic Properties

- Bit values
- 0x01-Broadcast
- 0x02-Read
- 0x04-Write without Response
- 0x08-Write
- 0x10-Notify
- 0x30-Indicate
- 0x40-Authenticated signed writes
- 0x80-Extended Properties

Example(s):

```
OPEN 245DFC020000 BLE 0
PENDING
OPEN_OK 14 BLE 245DFC020000
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[IDLE] ADVERTISING[ON] SCAN_UNI[OFF]
LINK 14 CONNECTED BLE 245DFC020000 MTU 250
OK
BLE_GET_CHAR 14
BLE_CHAR 14 U16 2A05 0003 20
BLE_CHAR 14 U16 2B29 0006 0A
BLE_CHAR 14 U16 2B2A 0008 02
BLE_CHAR 14 U16 2B3A 000A 02
BLE_CHAR 14 U16 2A00 000D 02
BLE_CHAR 14 U16 2A01 000F 02
```

Command	Description

```
BLE_CHAR 14 U128 00001101-D102-11E1-9B23-00025B00A5A5 0012 08
BLE_CHAR 14 U16 2B7E 005F 08
BLE_CHAR 14 U16 2B7F 0061 12
.......
BLE_CHAR 14 U16 2BBF 00E8 02
BLE_CHAR 14 U16 2BC0 00EA 10
BLE_CHAR 14 U16 2BC2 00ED 12
BLE_CHAR 14 U16 2BC1 00F0 12
BLE_CHAR 14 U128 06D1E5E7-79AD-4A71-8FAA-373789F7D93C 00F4 38
BLE_CHAR 14 U128 818AE306-9C5B-448D-B51A-7ADD6A5D314D 00F7 14
OK
BLE_GET_DES 14 6 8
BLE_CHAR_DES 14 U16 2B29 0006
BLE_CHAR_DES 14 U16 2803 0007
BLE_CHAR_DES 14 U16 2B2A 0008
OK
```

BLE_GET_SERV	GATT Service Discovery	Back to Table1: List of Commands

Description: Discovers the services of a remote GATT Server. A BLE_SERV notification is received for each service found.

Syntax: BLE_GET_SERV <link_id>

Response: PENDING
BLE_SERV <link_id> <type> <uuid> <start_handle> <end_handle>
OK

Parameter(s):
<link_id>: BLE Link Identifier

- 8-bit hex value (See Link ID Management)

<type>: UUID type

- U16, U32 or U128)

<uuid>: Characteristic UUID

- Hex format. ex: 2A05 (U16), 2A05-79AD (U32)

<start_handle>: Service Start Handle

- 4 digit Hex format. ex: 000C

<end_handle>: Service End Handle

- 4 digit Hex format. ex: 000C

Example(s):

```
BLE_GET_SERV 14
OK
BLE_SERV 14 U16 1801 0001 - 000A
BLE_SERV 14 U16 1800 000B - 000F
BLE_SERV 14 U128 00001100-D102-11E1-9B23-00025B00A5A5 0010 - 0018
BLE_SERV 14 U128 0000EB10-D102-11E1-9B23-00025B00A5A5 0019 - 0022
BLE_SERV 14 U16 FD92 0023 - 002B
BLE_SERV 14 U16 180F 002C - 0030
BLE_SERV 14 U16 184D 0031 - 0034
```

Command	Description

```
BLE_SERV 14 U16 1850 0035 - 0059
BLE_SERV 14 U16 1844 005A - 0062
BLE_SERV 14 U16 1855 0063 - 0065
BLE_SERV 14 U16 1853 0066 - 0066
BLE_SERV 14 U16 184F 0067 - 006F
BLE_SERV 14 U16 184E 0070 - 0085
BLE_SERV 14 U16 1858 0086 - 008C
BLE_SERV 14 U16 1849 008D - 00C4
BLE_SERV 14 U16 184C 00C5 - 00F1
BLE_SERV 14 U128 BC2F4CC6-AAEF-4351-9034-D66268E328F0 00F2 - FFFF
OK
```

BLE_INDIC	Turns ON/OFF Indication	Back to Table1: List of Commands

Syntax: BLE_INDIC <link_id> <handle> <action>

Response: OK
BLE_INDICATION <link_id> <handle> <size> <indication>
…

Parameter(s):
<link_id>: BLE Link Identifier

- 8-bit hex value (See Link ID Management)

<handle>: Handle

- 4 digit Hex format. ex: 000C

<size>: Size of Notification in Words

- Alphanumeric

<ind >: Notification

- Ascii

Example(s):

```
BLE_INDIC 14 D ON
OK
BLE_INDICATION 14 000D 4 1658A902
BLE_NOTIFICATION 14 000D 4 1657B102
BLE_NOTIFICATION 14 000D 4 1657B102
BLE_NOTIFICATION 14 000D 4 1657B102
```

BLE_INDICATION	GATT Indication Request	Back to Table1: List of Commands

Description: Send GATT Indication request over BLE. When PENDING response is received, enter the indication <data> and then the module will send the <data> as a GATT indication and return OK.

Syntax: BLE_INDICATION <link_id> <handle> <size>

Response: PENDING

Command	Description

*(Enter <data>)*OK

Parameter(s):

<link_id>: BLE Link Identifier

- 8-bit hex value (See Link ID Management)

<handle>: Handle

- 4 digit Hex format. ex: 000C

<size>: Size of Notification in bytes

- Alphanumeric

<data>: Characteristics data

- Ascii for hex. e.g. AA

Example(s):

```
BLE_INDICATION 14 18 1
PENDING
AA
OK
```

BLE_NOTIF	Turns ON/OFF Notification	Back to Table1: List of Commands

Syntax: BLE_NOTIF <link_id> <handle> <action>

Response: OK

BLE_NOTIFICATION <link_id> <handle> <size> <indication>

…

Parameter(s):

<link_id>: BLE Link Identifier

- 8-bit hex value (See Link ID Management)

<handle>: Handle

- 4 digit Hex format. ex: 000C

<size>: Size of Notification in Words

- Alphanumeric

<ind >: Notification

- Ascii

Example(s):

```
BLE_NOTIF 14 D ON
OK
BLE_NOTIFICATION 14 000D 4 1658A902
BLE_NOTIFICATION 14 000D 4 1657B102
BLE_NOTIFICATION 14 000D 4 1657B102
BLE_NOTIFICATION 14 000D 4 1657B102
```

Command	Description
BLE_NOTIFICATION	BLE Per: GATT Notification Request Back to Table1: List of Commands

Description: Send GATT Notification request over BLE. When PENDING response is received, enter the notification <data> and then the module will send the <data> as a GATT notification and return OK.

Syntax: BLE_NOTIFICATION <link_id> <handle> <size>

Response: PENDING
*(Enter <data>)*OK

Parameter(s):
<link_id>: BLE Link Identifier

- 8-bit hex value (See Link ID Management)

<handle>: Handle

- 4 digit Hex format. ex: 000C

<size>: Size of Notification in bytes

- Alphanumeric

<data>: Characteristics data

- Ascii for hex. e.g. AA

Example(s):

```
BLE_NOTIFICATION 14 18 1
PENDING
AA
OK
```

BLE_READ	BLE Central: GATT Read Request Back to Table1: List of Commands

Description: Get (Read) the value of the characteristic of a remote GATT server device.

Syntax: BLE_READ <link_id> <handle>

Response: PENDING
BLE_READ_RES <link_id> <handle> <size> <data>
…

Parameter(s):
<link_id>: BLE Link Identifier

- 8-bit hex value (See Link ID Management)

<handle>: Handle

- 4 digit Hex format. ex: 000C

<size>: Size of Notification in bytes

- Alphanumeric

<data>: Characteristic value

Command	Description

• ASCII string in Hex. eg: AA BB
 	
Example(s):
`BLE_READ 14 18`
`BLE_READ_RES 14 0018 1 AB`
`OK`	

BLE_READ_RES	BLE Peripheral:GATT Read Response Back to Table1: List of Commands

Description: Send a requested characteristic value to a remote GATT device, in response to a BLE_READ notification.
 	
Syntax: BLE_READ_RES <link_id> <handle> <size>
 	
Response: PENDING
(Enter <data>) OK
 	
Parameter(s):
<link_id>: BLE Link Identifier
• 8-bit hex value (See Link ID Management)
<handle>: Handle
• 4 digit Hex format. ex: 000C
<size>: Size of Notification in bytes
• Alphanumeric
<data>: Characteristic value
• ASCII string in Hex. eg: AA BB
 	
Example(s):
`BLE_READ_RES 14 18 1`
`PENDING`
`AA`
`OK`	

BLE_SET_DB	BLE Per: Set Custom GATT Database Back to Table1: List of Commands
Description: Set-up a custom Peripheral GATT database by specifying the database size, then sending data. This command returns a PENDING response. Enter data until the specified number of bytes has been sent and an OK response received. The data can be sent by batches separated by carriage return <CR>. **Notes:** To generate the database values, use the AA-BLE-Generator tool available at www.iot747.com. The database is not persistent after a reboot. **Syntax:** BLE_SET_DB <size>	

Command	Description

Response: PENDING
(Enter <data>)
PENDING
(Enter <data>)
…….
OK

Parameter(s):
<size>: Size of Notification in words

- Hex

<data>: Database Data

- ASCII string in Hex. eg: AA BB CC

Example(s):
Set up database with size=68 words (sent in 6x10-word batches and a 6xword batch separated by <CR>).

```
BLE_SET_DB 44
PENDING
0002 0118 3005 2003 0005 2A00 D000 6400 0002 0018
PENDING
3005 0207 0000 2A00 D400 3005 0209 0001 2A00 D400
PENDING
0010 F028 E368 62D6 3490 5143 EFAA C64C 2F00 3013
PENDING
180C 003C D9F7 8937 37AA 8F71 4AAD 79E7 E5D1 AA00
PENDING
CC01 0000 3053 180E 003C D9F7 8937 37AA 8F71 4AAD
PENDING
79E7 E5D1 BB00 CD01 0000 3093 1810 003C D9F7 8937
PENDING
37AA 8F71 4AAD 79E7 E5D1 CC00 CE01 0000
OK
```

BLE_WRITE	BLE Central: GATT Write Request	Back to Table1: List of Commands

Syntax: BLE_WRITE <link_id> <handle> <size>

Response: PENDING
(Enter <data>) OK

Write data (<size> bytes) to the specified characteristic <handle> of a remote GATT Server device (identified by <linkid>).
…

Parameter(s):
<link_id>: BLE Link Identifier

- 8-bit hex value (See Link ID Management)

<handle>: Handle

- 4 digit Hex format. ex: 000C

Command	Description

<size>: Size of Notification in bytes

- Alphanumeric

<data>: Characteristic value

- ASCII string in Hex. eg: AA BB

Example(s):

```
BLE_WRITE 14 18 1
PENDING
AA
OK
```

BROADCAST	Starts/Stops a Broadcast Transmission (Auracast)

Description: Starts a Broadcast Stream

Syntax: BROADCAST <action> <source> [pin]
Notes: Broadcast ID is set at the configuration BCAST_CONFIG. For <source>=2 (A2DP to Broadcast) only with SBC Codec (CODEC=OFF OFF OFF OFF OFF OFF). See CODEC.

Response: OK.

Parameter(s):
<action>:

- ON-Starts Broadcast
- OFF-Stops Broadcast

<source>:

- 1 – Analog or I2S (See AUDIO config)
- 2 – A2DP (You need to be receiving an A2DP Sink Audio to be able to use it)
- 3 – Auracast (You need to be receiving an Auracast Channel to be able to use it),

[pin]:

- Optional if <source=1> if the module is required to Broadcast a Private Channel
- PIN Code (4 to 16 alphanumerical characters)

Response: OK.

Example(s):

```
BROADCAST ON 1
OK
OPEN_OK 1C BTX1 245DFC020010
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[ON] ADVERTISING[ON] SCAN_UNI[OFF]
LINK 1C CONNECTED BTX1 245DFC020010 ANALOG PDEL 20000 RATE 48000 SUBGROUP 1
BISS 2
OK
```

Command	Description
BROADCODE	Sets the PIN for receiving a Private Auracast Channel

Description: Sets Pin code for a receiving a Private Broadcast Channel

Syntax: BROADCODE <link_id> <pin>

Parameter(s):
<link_id>: BLE Link Identifier

- 8-bit hex value (See Link ID Management)

<pin>:

- PIN Code (4 to 16 alphanumerical characters)

Response: OK.

Example(s):

```
OPEN 245DFC020010 BRX1 2 D 223344
OPEN_OK 1A BRX1 245DFC020010
OK
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[ON] ADVERTISING[ON] SCAN_UNI[OFF]
LINK 1A CONNECTED BRX1 245DFC020010 0x223344 IDLE
OK
MUSIC 1A PLAY
OK
BISS ENCRYPTED - ENTER CODE
BROADCODE 1 1234
OK
```

CALL	Manages Voice Call	Back to Table1: List of Commands

Description: Manages Voice Call

Syntax: CALL <link_id> <action> [<value>]

Response: OK.

Parameter(s):
<link_id>:

- 8-bit hex value (See Link ID Management)
- ALL (will disconnect all connections)

<action>:

For an HFP Link:

- ANSWER – Accept Incoming call
- CYCLE – Cycle between Calls (ex: Switching calls between 2 Handsets)
- END – Terminate Call
- ON_AG – Transfer the call Audio to AG (the phone)
- ON_HF – Transfer the call Audio to HF (the module)
- OUTGOING – Call ‘outgoing’ Request. Value parameter is required
- REJECT – Reject Incoming call

Command	Description

- TWC – Three Way Call action (See [value]

For an AGHFP Link:

- ANSWER – Notify the call has been answered from the AG side.
- END – Terminate Call
- INCOMING – Notify of an Incoming call. <value> parameter is required.
- OUTGOING – Notify 'Outgoing' call from AG side. Value parameter is required

<value>: (value depends on <action>)

- For <action>=OUTGOING – Required: Number (any combination of alphanumeric characters)
- For <action>=TWS – Required:
  - 0 - Release the held or waiting call
  - 1 - Release the active call and accept incoming/resume held
  - 2 - Hold the active call and accept incoming/resume held
  - 3 - Add the held or incoming call to a multiparty call
  - 4 - Add the held or incoming call to a multiparty call and leave the call

Example(s):

```
CALL INCOMING 13
CALL 13 ANSWER
OK
CALL_ACTIVE 13
SCO_OPEN 13
CALL 13 END
OK
SCO_CLOSE 13
CALL_END 13
```

CLOSE	Connection Close Request	Back to Table1: List of Commands

Description: Sends a connection close request to a link (specific profile for a specific device)

Syntax: CLOSE <link_id> OR ALL

Response: OK followed by a CLOSE_OK for every successful disconnection.

Parameter(s):
<link_id>:

- 8-bit hex value (See Link ID Management)
- ALL (will disconnect all connections)

Example(s):

```
STATUS
STATE CONNECTABLE OFF
LINK 10 CONNECTED A2DP 3CCD36230455
LINK 11 CONNECTED AVRCP 3CCD36230455
LINK 24 CONNECTED BLE 675DDBA8F833
OK
CLOSE 24
OK
CLOSE_OK 24 BLE
STATUS
STATE CONNECTABLE ADVERTISING
LINK 10 CONNECTED A2DP 3CCD36230455
LINK 11 CONNECTED AVRCP 3CCD36230455
```

Command	Description
OK	

CONFIG	Shows Configuration registers	Back to Table1: List of Commands

Description: Shows all configuration registers

Syntax: CONFIG

Response: <config_name>=<config_value>
....
OK

Example(s):

```
CONFIG
LOCAL_ADDR=245DFC010209
NAME=AA-010209
OK
```

Note: This is an example. Latest version of AudioAgent might have more configuration parameters. See the Configuration section for all configuration parameters available.

CONNECTABLE	Makes the module Connectable	Back to Table1: List of Commands

Description: Starts/Stops Making the Module Connectable by other Devices (Classic Bluetooth)

Syntax: CONNECTABLE <action>
Notes: The user can check in what state the module is by using the command STATUS. CONNECTABLE OFF will close all connections. When in CONNECTABLE OFF, a phoen which was previously paired will not be able to connect.

Response: OK

Parameter(s):
<action>:

- ON – The module becomes Connectable
- OFF - The module becomes Non-Connectable

Example(s):

```
CONNECTABLE ON
OK
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[IDLE] ADVERTISING[ON] SCAN_UNI[OFF]
CONNECTABLE OFF
OK
STATUS
STATE CONNECTABLE[OFF] DISCOVERABLE[IDLE] ADVERTISING[ON] SCAN_UNI[OFF]
OK
```

DISCOVERABLE	Starts/Stops Pairing Mode	Back to Table1: List of Commands

Command	Description

Description: Starts/Stops Bluetooth Pairing Mode (Classic Bluetooth)

Syntax: DISCOVERABLE <action>
Notes: The user can check in what state the module is by using the command STATUS. In Multi-Connection Mode, DISCOVERABLE ON will be rejected if a Device is already connected and is streaming Audio. The Audio should be Paused to enable Pairing. This is to make sure the quality of Audio is not affected by the radio Pairing activity. DISCOVERABLE ON will make the discoverable until the timeout kicks in (See UI_CONFIG. By default, it will stay DISCOVERABLE until a DISCOVERABLE OFF command is sent).

Response: OK

Parameter(s):
<action>:

- ON – Enable Pairing
- OFF - Disable Pairing

Example(s):

```
DISCOVERABLE OFF
OK
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[IDLE] ADVERTISING[ON] SCAN_UNI[OFF]
OK
DISCOVERABLE ON
OK
PAIR_PENDING
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[ON] ADVERTISING[ON] SCAN_UNI[OFF]
OK
```

ENTER_DATA_MODE	Enter Data Mode	Back to Table1: List of Commands

Description: Enters Data mode on a BLE or SPP link. (To exit data mode, enter the escape sequence $$$$). Note that the link has to be active for the command to work.

Syntax: ENTER_DATA_MODE <link_id>

Response: OK

<link_id>:

- 8-bit hex value (See Link ID Management)
- Must be a <link_id> for an SPP or BLE profile (ex: 14, 15)

Example(s): Enter Data Mode on Device 1 (a BLE Link)

```
ENTER_DATA_MODE 14
OK
```

HELP	Shows List of Commands (LC)	Back to Table1: List of Commands

Command	Description

Description: Shows all commands available

Syntax: HELP

Response: <command>
…
OK

Example(s):

```
HELP
ADVERTISING
AT
CLOSE
CONFIG
DISCOVERABLE
HELP
GET
LIST
MUSIC
NAME
OPEN
PAIR
POWER
RESET
RESTORE
SET
SEND
STATUS
UNPAIR
VERSION
VOLUME
WRITE
```

Note: This is an example. Latest version of AudioAgent might have more or less commands. See the Command section for all commands available.

GET	Shows the value of a config parameter Back to Table1: List of Commands

Description: Shows the value of a specific config parameter

Syntax: GET <config_name>

Response: <config_name>=<config_value>
OK

Parameter(s):
<config_name>:

- ASCII string (See Configuration)
- ALL (will disconnect all connections)

Example(s):

```
GET NAME
NAME=AA-010209
GET LOCAL_ADDR
```

Command	Description
LOCAL_ADDR=245DFC010209	
INQUIRY	Search for Bluetooth Devices

Description: Search for all Bluetooth Devices in the area

Syntax: INQUIRY [timeout]

Notes: The name of the devices found will not always appear. Sometimes, they will be UNKNOWN even though the devices have a friendly name. The host will need to use the NAME command to always het all the names of the devices found.

Response: PENDING
INQUIRY <bd_addr> “<name>” <cod> <rssi>
…
INQUIRY <bd_addr> “<name>” <cod> <rssi>
INQU_OK
OK

Parameter(s):
<timeout>:

- 1 to 20 (time in seconds to scan for devices)

Example(s):

```
INQUIRY 5
PENDING
INQUIRY 20FABB075DA5 "UNKNOWN" 240404 -54 dBm
INQUIRY 74B7E60BFE35 "WI-C100" 240404 -58 dBm
INQU_OK
```

LIST	Shows List of Paired Devices (LPD) Back to Table1: List of Commands

Description: Shows List of Paired Devices (LPD)

Syntax: LIST [keys]

Response: LIST <type of device> <bt_addr> [profile_1] … [profile_n]
…
OK

Note: The profile listed are the profiles supported by the Paired devices
The type of device is a letter specifying the type of device the module is connected to (H=Handset, S=Sink (or headset), E=Earbud, U=Unknown).

Parameter(s):
<keys>:

- ON – Display Link Keys
- OFF (Default) – Do not display Link keys

Command	Description

Example(s):

```
LIST
LIST H 30D875EB8C1C HFP A2DP AVRCP
OK
LIST ON
LIST H 30D875EB8C1C HFP A2DP AVRCP
BREDR_KEY 74B4665399D8C8EABD55319A782A1EBB
LE_LTK AF4061DFB175093F00E8DD17090FA9A2
OK
```

MICBIAS	Controls the MIC_BIAS Pin on the Module

Description: Controls the Mic_Bias_A Pin on the module. Refer to the Datasheet. This can be helpful if the host wants to control that pin directly or for test and Mic adjustment purposes.

Syntax: MICBIAS <action>

Notes: ON will get the pin to 2.1V if there is AUDIO on the line. OFF will bring it down to 0V. Default is ON.

Response: OK

Parameter(s):
<action >:

- ON or OFF

Example(s):

```
MICBIAS ON
OK
MICBIAS OFF
OK
```

MUSIC	Controls the Music Stream Back to Table1: List of Commands

Description: Controls the Bluetooth Music Stream

Syntax: MUSIC <link_id> <action> [index]

Response: OK

Notes: The Response can be followed by a notification if the music state changed (ex: A2DP_STREAM_START 10, AVRCP_PLAY 11)

Parameter(s):

<link_id>:

- 8-bit hex value (See Link ID Management)
- Must be a <link_id> for an A2DP, AVRCP or BRX1 or BRX2 profile (ex: 10, 1A, 1B).

<action>: (Only PLAY/PAUSE for BRX1 and BRX2)

- PLAY
- PAUSE
- STOP

Command	Description

- FORWARD
- BACKWARD
- FF_PRESS
- FF_RELEASE
- REW_PRESS
- REW_RELEASE

[index]: (Allows to select which Audio Stream to Play)

- 1 to 4. Only used with Auracast (link_id=1A or 1B)
- See BCAST_REPORT

Example(s):

```
MUSIC 10 PLAY
OK
A2DP_STREAM_START 10
AVRCP_PLAY 11
MUSIC 10 PAUSE
OK
AVRCP_PAUSE 11
A2DP_STREAM_SUSPEND 10
MUSIC 11 PLAY
OK
AVRCP_PLAY 11
A2DP_STREAM_START 10
MUSIC 11 PAUSE
OK
AVRCP_PAUSE 11
A2DP_STREAM_SUSPEND 10
```

NAME	Get Remote Device Name	Back to Table1: List of Commands

Description: Get the name of the remote device

Syntax: NAME <bt_addr>

Response: PENDING
NAME <bd_addr> “<name>”

Purpose: Get the name of the remote device at the specified Bluetooth address. This command can be used if the device has not reported its name during the initial INQUIRY command. If the name initially appeared as UNKNOWN from the INQUIRY command, the host can request the name to get it. This is to make sure the INQUIRY command uncovers all Bluetooth devices Discoverable.

Parameter(s):
<bd_addr>: Bluetooth Address of the device to connect to

- 12 hex value (e: 245DFC010209)

Example(s):

```
NAME EC28D37886D3
PENDING
```

Command	Description
NAME EC28D37886D3 "iPhone 12"	
OPEN	Connection Open Request Back to Table1: List of Commands

Description: Sends a connection open request to a link (specific profile for a specific device)

Syntax: OPEN <bt_addr> <profile>. [<addr_type> <adv_sid> <bcast_id> <sync>]

Response: PENDING
(OPEN_OK notification or OPEN_ERROR notification)
Note: Except for BRX1 and BRX2 (Auracast) where there is no pairing, the OPEN request will succeed only if the device with bt_addr is connectable and previously paired (i.e. in the list of Paired Devices (LPD)). If the device is not in the pairing list, the PAIR command should be used first and the device should be in Discoverable (or Pairing) mode. Also, sometimes, if you initiate an A2DP connection request, the remote device will automatically initiate an AVRCP connection request. So you will end up with both A2DP and AVRCP connections

Parameter(s):
<bd_addr>: Bluetooth Address of the device to connect to

- 12 hex value (e: 245DFC010209)

<profile>: Profile to open

- A2DP, AVRCP, AGHFP, HFP, BLE, SPP
- URXM, URXV, UTXM, UTXV, BRX1, BRX2,

<addr_type>: Address type

- 0- Private (When opening a BLE connection, you need to specify this)
- 1- Random (When opening a BLE connection, you need to specify this)
- 2- Broadcast (Only with BRX1 and BRX2)

<adv_sid>: Advertising sid (Only for BRX1 and BRX2)

- Hex parameter
- Refer to SCAN command for sid

<bcast_id>: Broadcast ID (Only for BRX1 and BRX2)

- long Hex (ex: 112233)

<pa_sync>: PA Sync (Only for Broadcast)

- 0-Open but do not play Audio yet
- 1-Open and Play Audio

Example(s):

```
>>Here we open a A2DP Profile with a previously paired phone. The phone opens
then AVRCP and HFP.

LIST
LIST H 30D875EB8C1C HFP A2DP AVRCP
```

Command	Description

```
LIST S 245DFC020000 HFP A2DP AVRCP
OK
OPEN 30D875EB8C1C A2DP
PENDING
OPEN_OK 10 A2DP 30D875EB8C1C
OPEN_OK 11 AVRCP 30D875EB8C1C
AVRCP_PAUSE 11
OPEN_OK 13 HFP 30D875EB8C1C
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[IDLE] ADVERTISING[ON] SCAN_UNI[OFF]
LINK 10 CONNECTED A2DP 30D875EB8C1C SUSPENDED SBC SNK 44100
LINK 11 CONNECTED AVRCP 30D875EB8C1C PAUSED
LINK 13 CONNECTED HFP 30D875EB8C1C IDLE
OK


>>Here we open a BLE Profile with another IDC777 module.


SCAN 5 OFF 0
SCAN 0 245DFC020000 0 <unknown> 1A -34db
SCAN 0 245DFC020000 0 <AA-020000> 00 -35db
SCAN 0 001343D823C4 0 <unknown> 06 -68db
….
SCAN 0 0A8D438319FC 1 <unknown> 00 -97db
SCAN 0 354B45644EAA 1 <unknown> 00 -95db
SCAN 0 29A6C7F8C083 1 <unknown> 00 -82db
SCAN 0 354B45644EAA 1 <unknown> 00 -100db
SCAN 0 30D875EB8C1C 0 <unknown> 1A -52db
SCAN 0 30D875EB8C1C 0 <unknown> 00 -52db
OK
OPEN 245DFC020000 BLE 0
PENDING
OPEN_OK 24 BLE 245DFC020000
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[IDLE] ADVERTISING[ON] SCAN_UNI[OFF]
LINK 10 CONNECTED A2DP 30D875EB8C1C SUSPENDED SBC SNK 44100
LINK 11 CONNECTED AVRCP 30D875EB8C1C PAUSED
LINK 13 CONNECTED HFP 30D875EB8C1C IDLE
LINK 24 CONNECTED BLE 245DFC020000 MTU 250
OK



>>Here we open an Auracast Channel and play music from it

SCAN 5 OFF 2
PENDING
SCAN 2 245DFC020000 0x223344 0xd AA-020000 -36
SCAN_OK
OPEN 245DFC020000 BRX1 2 D 223344
OPEN_OK 1A BRX1 245DFC020000
OK
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[IDLE] ADVERTISING[ON] SCAN_UNI[OFF]
LINK 1A CONNECTED BRX1 245DFC020000 0x223344 IDLE
OK
MUSIC 1A PLAY
```

Command	Description

```
OK
LC3_BROADCAST_NUMBER_BIS 2
LC3_BROADCAST_STREAM_CONNECT
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[OFF] ADVERTISING[ON] SCAN_UNI[OFF]
LINK 1A CONNECTED BRX1 245DFC020000 0x223344 STREAMING LC3 PDEL 20000 RATE
48000 ENCR: 0 SUBGROUP 1
OK
```

PAIR	Pair Request Back to Table1: List of Commands

Description: Sends a pair request to a specific device (or Bluetooth Address).

Syntax: PAIR <bd_addr>

Response: PAIR PENDING
(PAIR_OK notification or PAIR_ERROR notification)

Parameter(s):
<bd_addr>: Bluetooth Address of the device to connect to

- 12 hex value (e: 245DFC010209)

Example(s):

```
PAIR 3CCD36230455
PENDING
PAIR_OK 3CCD36230455
LIST
LIST 3CCD36230455 HFP A2DP AVRCP
OK
```

PAIR_LE	Pair Request on the BLE Channel Back to Table1: List of Commands

Description: Sends a BLE pair request to a specific device (or Bluetooth Address).

Syntax: PAIR <bd_addr> <addr_type>

Response: PAIR PENDING
(PAIR_OK notification or PAIR_ERROR notification)

Note(s): Only use PAIR_LE from the initiator of the connection (i.e. the same device that initiates the Pairing initiated the connection (OPEN <bd_address> xxx)).

Parameter(s):
<bd_addr>: Bluetooth Address of the device to connect to

- 12 hex value (e: 245DFC010209)

<addr_type>: Public or Private Address. Refer to SCAN <type> response

- 0 (Public Address)

Command	Description

- 1 (Private Address)

Example(s):

```
PAIR_LE 245DFC020000 0
PENDING
OPEN_OK 14 BLE 245DFC020000
PAIR_OK_LE 245DFC020000
```

PASSKEY	Pairing User Confirmation	Back to Table1: List of Commands

Description: After receiving a PAIR_PASSKEY reqest notification, use this command to accept/reject the pairing request

Syntax: PASSKEY <type><value>

Response: OK
(PAIR_OK notification or PAIR_ERROR notification)

Parameter(s):
<type>: Pairing Response type

- 1 – Confirmation type. <value>=1 (accept) or 0 (reject)
- 0 – Passkey type. <value>=6 digit passkey

<value>: <type> dependent

- ASCII string
- See <type> descriptions.

Example(s):

```
SET SSP_CAPS=1
OK
WRITE
OK
RESET
…Phone starts Pairing
PAIR_PASSKEY 887598BA1A7D 1 793885 (you should see 793885 on the phone.
PASSKEY 1 1
OK
… Pairing successful
```

PIO	Set GPIO State	Back to Table1: List of Commands

Description: Set General Purpose I/O (GPIO) State.
Notes: GPIO control must be disabled (via GPIO_CONFIG) before this command can be used.

Syntax: POWER <pio> <state>
Response: OK

Parameter(s):

Command	Description

<pio >: (GPIO index)

- Valid values: 2,5,7,8

Example(s):

```
PIO 8 ON
OK
(PIO8 so PIN24 on IDC747 will go high to 3V3).
```

POWER	Power ON/OFF module	Back to Table1: List of Commands

Description: Powers ON/OFF Bluetooth Module.

Syntax: POWER <action1> [<action2>]
Notes: POWER OFF will put the module into Dormant mode. A RESET or SYS_CTR will allow it to power back on. POWER OFF OFF will completely shut down the module. A SYS_CTRL is then needed to Power Back up the module. If the hardware does not have SYS_CTRL connected, the only way to Power Back up the module is to remove Power and re-apply it or to apply voltage to VCHG and VCHG_SENSE. See Datasheet. For POWER OFF (Dormant), auto_power_on needs to be set to OFF. See UI_CONFIG.

Response: OK

Parameter(s):
<action1>:

- ON or OFF

<action2>

- ON or OFF

Example(s):

```
POWER OFF OFF
OK

The module then shuts down completely and SYS_CTRL is needed to power it up
```

PROMPT	Plays Prerecorded Prompts

Description: Plays Prerecorded Prompts

Syntax: PROMPT <prompt>
Notes: Only use when AUDIO=0 0 and not when in a ROUTE command.

Response: OK

Parameter(s):
<prompt>:

- 1-Connected
- 2-Disconnected
- 3-Pairing
- 4-Pairing Failed

Command	Description

- 5-Pairing Successful
- 6-Power Off
- 7-Power On

Example(s):

```
PROMPT 7
OK
(IDC777 will play Power ON)
```

RESET	Resets module Back to Table1: List of Commands

Description: Resets module

Syntax: RESET

Response: Reset followed by boot-up Prompt
Notes: A RESET will close all Bluetooth connection and any configuration parameters that were not saved with the WRITE command will be lost.

Example(s):

```
RESET
IOT747 Copyright 2022
AudioAgent V1.99.2
Build: 1638377096
Bluetooth address 245DFC010209
Ready
```

RESTORE	Restores Default Config parameters Back to Table1: List of Commands

Description: Resets module and Restores default configuration parameters.

Syntax: RESTORE

Response: Reset followed by boot-up Prompt
Notes: A RESTORE will delete all current configuration parameters and restore factory setting. For example, the Bluetooth device name will be restored to AA-XXXXXX where XXXXXX are the last hex values of the Bluetooth Address. A RESTORE command will erase the List of Paired Devices (LPD).

Example(s):

```
RESTORE
IOT747 Copyright 2022
AudioAgent V1.99.2
Build: 1638377096
Bluetooth address 245DFC010209
Ready
GET NAME
NAME=AA-010209
```

ROLE	Controls/Inquires about Bluetooth Role (Master/Slave)

Description: Controls or Inquires about Bluetooth Role (Master/Slave)

Syntax: ROLE <linkid> [s/m]

Command	Description

Response:
ROLE_OK <bt_address> S/M

Example(s):

```
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[IDLE] ADVERTISING[ON] SCAN_UNI[OFF]
LINK 10 CONNECTED A2DP 30D875EB8C1C SUSPENDED SBC SNK 44100
LINK 11 CONNECTED AVRCP 30D875EB8C1C PAUSED
LINK 13 CONNECTED HFP 30D875EB8C1C IDLE
OK
ROLE 10
OK
ROLE_OK 30D875EB8C1C S
ROLE 10 M
OK
ROLE_NOT_ALLOWED 30D875EB8C1C S
```

ROUTE	Routing of Audio	Back to Table1: List of Commands

Description: Set routing of Audio

Syntax: ROUTE <value>

Response:
OK

Notes: In normal usage, Automatic routing is used. In Automatic routing, an HFP (call) link has a priority over an A2DP (music) link. If more than one device is connected, the first device (or currently streaming device) has priority. The Route command can only be used on its own. The ROUTE command should dbe used after a RESET. And after a ROUTE command, the module should be RESET to resume normal operation. ROUTE is usually used to test the Audio at production or help when figuring out I2S settings. Note that ROUTE 37 uses the CVC echo canceller. So it will only take the Left Channel of the signal. ROUTE 2 will take both channels.

Parameter(s):

<value>: (where to route Audio)

- 0-(Default) Automatic routing
- 1-Analog input to Analog output
- 2-Digital interface to Analog Interface (Bidirectional)
- 3-Digital input to Digital output
- 4-Digital input to Analog output
- 5-Analog input to Digital output
- 6-USB input to Analog Output (48Khz)
- 7-USB Interface to Analog Interface (Bi-directional) (16 Khz)
- 8-SPDIF Interface to Analog
- 9-Analog Interface to SPDIF
- A-Analog to SPDIF (Bidirectional)
- B-SPDIF to SPDIF (Loop back)
- 37-Similar to Route2 but the echo cancellation and noise suppression is enabled.

Example(s):

```
ROUTE 2
```

Command	Description
OK (Module acts like an I2S/Analog ADC and DAC converter)	
RSSI	Get the RSSI for the ACL with given remote device Back to Table1: List of Commands

Description: Get the RSSI for the ACL with given remote device

Syntax: RSSI <Bluetooth address>

Response: RSSI=xxx

OK

Notes: Returns the RSSI (in dBm) of the Bluetooth signal received from paired device with given Bluetooth address. If an invalid Bluetooth address is given, or the device with the given Bluetooth address is not paired to the module, an error will be returned.

Parameter(s):

<Bluetooth address>

Example(s):

```
RSSI 1869D4262A18
RSSI=-53
OK
```

QUALITY	Obtain the ACL rx link quality Back to Table1: List of Commands

Description: Obtain the ACL rx link quality

Syntax: QUALITY <Bluetooth address>

Response: RSSI=<xxx>

OK

Notes: Returns the quality of the Bluetooth signal received from paired device with given Bluetooth address. If an invalid Bluetooth address is given, or the device with the given Bluetooth address is not paired to the module, an error will be returned. The value returned will be between 0 (lowest/worst quality) to 65535 (highest/best quality).

Parameter(s):

<Bluetooth address>

Example(s):

```
QUALITY 1869D4262A18
QUALITY=65331
OK
```

SCAN	Scans for BLE Device Back to Table1: List of Commands

Command	Description

Description: Searches for BLE Devices in the area

Syntax: SCAN <time> [<format>] [<filter>]

Response:
PENDING
SCAN [<filter>] <bd_addr> <type> <name><flags><rssi>
Or SCAN_RAW <bd_addr><type><rssi><size><data>
Or SCAN <bd_addr> <bcast_id> 2 <adv_id> <name> <rssi>
Or SCAN [<filter>] <bd_addr> <bcast_id> 2 <adv_id> <name> <rssi>
…

SCAN_OK

Notes: SCAN is only used for BLE. For Classic, you should use INQUIRY command. When using SCAN, the same device might appear several times. The response will be SCAN or SCAN_RAW depending on the format. When doing a SCAN for general devices <filter>=0, there might be too many devices around. So a higher UART rate might be needed not to overflow the UART buffer. See UART_CONFIG

Parameter(s):

<timeout>: Scan duration in seconds

- Range: 0-255
- 0-Stop Scanning
- 255-Scan continuously – Only for filter=0

<format>:

- ON-SCAN_RAW format – Only for filter=0
- OFF-(Default) SCAN notifications

<filter>:

- 0 – No filter (Show all BLE devices available)
- 2 – Show only LE Audio Broadcasters (Auracast)
- 3 – Show only LE Audio Receivers (Unicast Receivers)

<bd_addr>: Bluetooth Address of the device

- 12 hex value (e: 245DFC010209)

<type>: Address type

- 0-Public
- 1-Private

<name>: Device Name or LE Broadcaster name

- ASCII value between <>. Example <AA-1234> or <UNKNOWN> if not reported

<bcast_id>: LE Broadcast ID (to be used when OPEN command is used)

- Long Hex value (ex: 112233)

<avd_id>: Advertising ID (to be used when OPEN command is used)

- Hex value (ex: F)

<flag>: Advertising flags as define in Bluetooth Specs

- Hex value

<rssi>: Signal strength in dBm

Command	Description
• ACII string (ex: -58dBm) Example(s): TBD	
SCAN_UNI	Unicast Tx to Scan for Unicast Receivers trying to connect
Description: This command enables a Unicast Transmitter (The device that initiated the connection) to be open for Receivers to open a Unicast Voice or Music Channel. It also allows the Auracast transmitter to receive Data from the Receivers (See SEND (1A or 1B) command and RECV (1C) notification). **Syntax:** SCAN_UNI <enable > [<time> <delay>] Response: OK Parameter(s): <enable>: enable scan • ON or OFF <time>: • Time for every scan (in seconds) • Value between 1 and 10. Default is 1s if no parameter is set. <delay>: • Delay netween each Scan • Value between 1 and 10. Default is 1s if no parameter is set.	
SCO_OPEN	Opens SCO Channel
Description: This command opens a Bluetooth Classic Bi-directional Audio Channel to the connected device over HFP. **Syntax:** SCO_OPEN <link_id> Response: OK Parameter(s): <link_id>: • 8-bit hex value (See Link ID Management) • Must be a <link_id> for an HFP (13) or AGHFP profile (12)	
SCO_CLOSE	Closes SCO Channel
Description: This command opens a Bluetooth Classic Ni-directional Audio Channel to the connected device over HFP. **Syntax:** SCO_CLOSE <link_id> Response: OK	

Command	Description

Parameter(s):
<link_id>:

- 8-bit hex value (See Link ID Management)
- Must be a <link_id> for an HFP (13) or AGHFP profile (12)

SEND	Sends Data	Back to Table1: List of Commands

Description: Sends Data (characters) to the BLE (Link 14), Auracast (1A, 1B or 1C) or SPP Links.

Syntax: SEND <link_id> <string>
Response: OK
Notes: SEND can be used to send any Data to another BLE or SPP device. For example, it can be used with an iOS and Android App to send data back and forth with the module. Please refer to IOT747 Scanner Sample Apps on Google Play or Apple Store. It can also be used to send Data between IDC777 modules using the BLE, SPP or Auracast Channels.

Parameter(s):
<link_id>:

- 8-bit hex value (See Link ID Management)
- Must be a <link_id> for a BLE profile (14), Auracast (1A, 1B, 1C) or SPP (15)

<string>:

- String of characters (max: 80 bytes)

Example(s):

```
>>Here we opened a BLE connection uing IOT747 Scanner and send and received
Data over the BLE Link

PAIR_PENDING
OPEN_OK 14 BLE 30D875EB8C1C
RECV 14 5 HELLO
SEND 14 HELLO2
OK

>>Here we received a Auracast channel from an IDC777 and send and received
data from the transmitter.

SCAN 5 OFF 2
PENDING
SCAN 2 245DFC020000 0x223344 0xd AA-020000 -50
SCAN_OK
OPEN 245DFC020000 BRX1 2 D 223344
OPEN_OK 2A BRX1 245DFC020000
OK
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[ON] ADVERTISING[ON] SCAN_UNI[OFF]
LINK 14 CONNECTED BLE 30D875EB8C1C MTU 250
LINK 2A CONNECTED BRX1 245DFC020000 0x223344 IDLE
OK
MUSIC 2A PLAY
OK
LC3_BROADCAST_NUMBER_BIS 2
```

Command	Description

```
LC3_BROADCAST_STREAM_CONNECT
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[ON] ADVERTISING[ON] SCAN_UNI[OFF]
LINK 14 CONNECTED BLE 30D875EB8C1C MTU 250
LINK 2A CONNECTED BRX1 245DFC020000 0x223344 STREAMING LC3 PDEL 20000 RATE
48000 ENCR: 0 SUBGROUP 1
OK
>>We received this message from the Auracast transmitter
RECV 2A 14 HELLORECEIVERS
>>We send this message to the Auracast transmitter
SEND 2A HELLOTRANSMITTER
OK

>>Here we are the Auracast transmitter and we send and receive messages.

BROADCAST ON 1
OK
OPEN_OK 1C BTX1 245DFC020000
status
STATE CONNECTABLE[ON] DISCOVERABLE[ON] ADVERTISING[ON] SCAN_UNI[OFF]
LINK 1C CONNECTED BTX1 245DFC020000 ANALOG PDEL 20000 RATE 48000 SUBGROUP 1
BISS 2
OK
>>SCAN_UNI ON means we will be scanning for messages from the receivers
SCAN_UNI ON
OK
SEND 1C HELLORECEIVERS
OK
RECV 1C 16 HELLOTRANSMITTER
```

SEND_RAW	Send Raw Data

Description: Send Raw data to a BLE or SPP link. After sending the command, a PENDING response is received. The data received after that is used to set the raw data to send. OK is returned when the expected number of bytes is received.

Syntax: SEND_RAW <link_id><size>
Response: PENDING
<data>OK

Parameter(s):
<link_id>: Link ID for BLE or SPP

- 8-bit hex value (See Link ID Management)
- Must be a <link_id> for a BLE or SPP profile (ex: 14, 15)

<size>: Number of bytes to send

- Range: 1-255

< data>: Data to send

- Raw data for ex: {68}{65}{6c}{6c}{0a} indicates five bytes with values 0x68, 0x65, 0x6c, 0x6c,0x0a are sent (hexadecimal values of the ASCII characters in “Hello”).

Command	Description

Example(s):

```
Send "1234567890" To the BLE link on device 1
In this case an App (IOT747 Scanner) connected over BLE.
OPEN_OK 14 BLE 6A288E18D114
SEND_RAW 14 10
OK
1234567890
OK
```

SET	Sets configuration parameter Back to Table1: List of Commands

Description: Sets the value of a specific config parameter

Syntax: SET <config_name>
Response: OK

Notes: This sets the configuration parameter in RAM. To avoid it being deleted at the next RESET, you should use the WRITE command to store in in Flash.

Parameter(s):
<config_name>:

- ASCII string (See Configuration)

Example(s):

```
GET NAME
NAME=AA-010203
OK
SET NAME=SPKR2
OK
GET NAME
NAME=SPKR2
OK
WRITE
OK
```

SSRD	Set Advertising Data

Description: Sets Advertising Data for BLE. After sending the command, a PENDING response is received. The data received after that is used to be added to the Advertising packets. OK is returned when the expected number of bytes is received.

Syntax: SSRD <size>
Response: PENDING
<data>OK

Notes: The <data> are the bytes of the Advertising Data. This command is used when the user needs the module to behave as a BLE peripheral and at the same time have a specific BLE advertising packet with a vendor ID for example. It is often used with BLE_SET_DB when the module has to act as a peripheral device with a proprietary (customer defined) BLE profile. Note that Swift Pair will be disabled when SRRD is used. As the Data is limited on the Advertising Packet.

Parameter(s):
<size>:

Command	Description

- Number of bytes to set
- 1-25

<data>: Data being set

- Raw data
- Ex: {07}{09}{4D} indicates bytes with values 0x07, 0x09, 0x04D are sent

Example(s):

```
SSRD 8
NAME=AA-010203
{07}{09}{4D}{79}{43}{61}{6D}{65}OK
```

STATUS	Returns the State of the device	Back to Table1: List of Commands

Description: Shows Bluetooth State and lists module active connections

Syntax: STATUS

Response: STATE CONNECTABLE [<state>] DISCOVERABLE [<state>] ADVERTISING [<state>] SCAN_UNI [<state>]

LINK <link_id> <profile> <bd_addr> <profile_info>

…

OK

<state>:

- ON-Connectable, Discoverable, Advertising or Scan_Uni (Only IDC777)
- OFF- Not Connectable, Discoverable, Advertising or Scan_Uni (Only IDC777)

<link_id>:

- 8-bit hex value (See Link ID Management)

<profile>:

- A2DP, AVRCP, AGHFP, HFP, BLE, SPP, URXM, URXV, UTXM, UTXV, BRX1, BRX2, BTX1

<bd_addr>: Bluetooth Address of the device to remove form the pairing list

- 12 hex value (e: 245DFC010209)

<profile_info>: Information about the profile

- For A2DP: <streaming> <codec> <role> <sample rate>
- <streaming>: SUSPENDED, STREAMING
- <codec>: SBC, AAC, APTX, APTX-HD, APTX-LL
- <role>: SNK (A2DP Sink), SRC (A2DP Source>
- <sample_rat>: Sampling rate in Hz
- For AVRCP <status>
- <status>: STOPPED, PLAYING, PAUSED
- For HFP <call_state> <codec>
- <call_state>: IDLE, OUTGOING, INCOMING, ACTIVE
- <codec> NB (Narrow Band), WB (Wide Band)
- <sco> If a SCO audio link is open

Command	Description

- For BLE <mtu>
- <mtu> - Size in bytes of MTU negotiated
- For URXM: <streaming> <codec> <media> <le profile> <source> <sample rate> <media_source>
- For URXV: <streaming> <codec> <media> <le profile> <source> <sample rate> <media_source>

Example(s):

```
STATUS
STATE CONNECTABLE[ON] DISCOVERABLE[IDLE] ADVERTISING[OFF]
LINK 10 CONNECTED A2DP 3CCD36230455 AAC SNK 44100
LINK 11 CONNECTED AVRCP 3CCD36230455 PAUSED
LINK 24 CONNECTED BLE 7E33C0AB936D 80
OK
OK
```

TONES	Generate Custom Tone

Description: Play a tone (a sequence of notes). Tones are defined as a sequence of notes described with several characteristics—pitch (N or TN), length (L), tempo (TE), volume (V), timbre (TI) and decay (D).

Syntax: TONE <flag> <value> [... <flag> <value>]

Response: OK

Notes:
• All characteristics can be individually set for each note, changed at any place in the tone string, or omitted (all except length) to use default values.
• A tone must have at least one note, and each note must have at least a length characteristic set (ex: “N AS7 L 1”—note pitch A sharp, octave 7, whole note).
• Notes will get mixed if an A2DP link is already playing Audio.

Parameters:

<flag> (Tone characteristic)
• TE—Tempo in quarter notes per minute.
<value> range—0–4095; default 120
• TI—Timbre <value>:
-0—Sine wave
-1—Square wave
-2—Saw-tooth wave
-3—Triangle wave
-4—Triangle wave (asymmetric)
-5—Clipped sine wave
-6—Simulates a plucked instrument
•V—Volume.
<value> range—0–255; Default 50
• D—Decay
<value>-Two-digit hexadecimal value, range 00–FF
-Interpreted as fixed point decimal number in format <digit_1>.<digit_2> (ex: 15=1.5)
-As each tone is played, its volume decreases with variable rate. Low values for this parameter

Command	Description

cause notes to decay very quickly; high values cause notes to continue with an almost-constant volume. For example: 05 (0.5) causes each note to reach zero halfway through its duration, giving a staccato feel. 20 (2.0) causes each note to reach half its initial value when the next note starts.
-Default: 20. This allows notes of the same length to be tied together with TN.

• N—Note pitch
<value> format—<note>[<flat_sharp>]<octave>
-<note>: A, B, C, D, E, F, G, or R (Rest/Pause)
-<flat_sharp>: F (Flat) or S (Sharp)
-<octave>: 0–9. (Note: Must be specified for all <note> values, including 'R", (ex: R0)
-TN—Tied note. Note is "tied" to the next note (the pitch of the note appears to change as opposed to a new note starting), and the volume continues to decay from the previous note. Note: An appropriate decay must be set for this to work.

• L—Length of note <value>:
-1—Whole note
-2—Half note
-3—Half note triplet
-4—Quarter note
-6—Quarter note triplet
-8—Eighth note
-12—Eighth note triplet
-16—Sixteenth note
-24—Sixteenth note triplet
-32—Thirty-second note
-48—Thirty-second note triplet
-64—Sixty-fourth note
-96—Sixty-fourth note triplet

<value> (<flag>-dependent value)
• See <flag> parameter descriptions for supported values.

Example(s):
• Sample tone:

```
TONE TE 400 V 64 TI 0 N C5 L 8 N R0 L 32 N E5 L 8 N R0 L 32 N G5 L 8 N R0 L 32 N B5 L 4 N R0 L 1 N C6 L 2 TN C6 L 8
OK
```

• Variable volume single note:

```
TONE V 64 N C6 L 4 V 128 N C6 L 4 V 255 N C6 L 4 V 128 N C6 L 4 V 64 N C6 L 4
OK
```

• Musical scale starting from C4 and omitting flat and sharp tones:

```
TONE V 128 TI 0 N C4 L 8 N D4 L 8 N E4 L 8 N F4 L 8 N G4 L 8 N A4 L 8 N B4 L 8 N C5 L 8
OK
```

UNPAIR	Deletes Device from Pairing List Back to Table1: List of Commands

Description: Deletes device from Pairing list

Command	Description

Syntax: UNPAIR [bd_addr]

Notes: UNPAIR without any parameters deletes the complete List of Paired Devices (LPD). If you UNPAIR from the module but not from the other device (ex: phone). You won't be able to Repair with that device. You have to remove the module (or forget on iOS) from the other device also. Also, please CLOSE ALL before using the command UNPAIR. If the device is connected when you send the command UNPAIR, it will not be deleted from the LPD or might Panic in some cases.

Response: OK

Parameter(s):
<bd_addr>: Bluetooth Address of the device to remove form the pairing list

- 12 hex value (e: 245DFC010209)

Example(s):

```
LIST
LIST 3CCD36230455 HFP A2DP AVRCP
LIST 887598BA1A7D HFP A2DP AVRCP
OK
UNPAIR 3CCD36230455
OK
LIST
LIST 887598BA1A7D HFP A2DP AVRCP
OK
```

VOLUME	Controls Volume	Back to Table1: List of Commands

Description: Sets or Gets the current volume from the Link_ID.

Syntax: VOLUME [<link_id>] [<value>]

Response: OK
Notes: VOLUME without parameters will return the current volume of all active links. For A2DP, this is between 0 and 127 in steps of 8. For HFP, it is an integer between 0 and 15 in steps of 1. VOLUME <link_id>. without a value will return the Volume of that Link in integer format. VOLUME with a value sets the Volume value. When a value is set for volume, it will be rounded to fit in the steps (steps of 16 in the case of A2DP). When sending a volume A2DP command the value sent will be rounded to the nearest 8 multiple (ex: VOLUME 10 18 will be rounded to VOLUME 10 16). For Analog In and Analog Out, the value will override and set AUDIO_ANALOG Configuration.

Parameter(s):
<link_id>:

- 8-bit hex value (See Link ID Management)
- 0 will control the Analog In volume (Input Codec amplifier)
- 1 will control the Analog Out volume (Output Codec amplifier)

<value>:

- UP (Volume will increase by 8 steps for A2DP and 1 step for HFP and others)
- DOWN (Volume will decrease by 8 steps for A2DP and 1 step for and others)
- Integer between 0 and 127 in steps of 8 for A2DP
- Integer between 0 and 15 for HFP or AGHFP
- Integer between 0 and 10 for Analog-In (link_id=0)

Command	Description

- Integer between 0 and 22 for Analog-Out (link_id=1)
- Integer between 0 and 255 for URX1, URX2, BRX1, UTX1, UTX2, BTX1

Example(s):

```
VOLUME
10 A2DP 56
OK
VOLUME 10 UP
OK
ABS_VOL 11 64
VOLUME
10 A2DP 64
OK
VOLUME 10 18
OK
ABS_VOL 11 16
```

VERSION	Shows Firmware Version	Back to Table1: List of Commands

Description: Gives information on firmware VERSION

Syntax: VERSION

Response: IOT747 Copyright xxxx
AudioAgent Vx.xx.xx
Build: xxxxxxxxx
Bluetooth address <bd_addr>

Example(s):

```
VERSION
IOT747 Copyright 2022
AudioAgent V1.99.2
Build: 1638377096
Bluetooth address 245DFC010203
OK
```

WRITE	Store Configurations in Flash	Back to Table1: List of Commands

Description: Saves all configuration values from RAM to Flash (Persistent Memory)
Notes: Please refer to the Configurations section

Syntax: WRITE

Response: OK

Example(s):

```
WRITE
OK
```

Command	Description

AudioAgent Configuration

The user can configure general parameters for the module. These parameters are stored in the RAM memory. If required, the parameters can be stored to Flash memory. When the module reboots, it will boot with the parameters that are saved to Flash memory.

There are 4 main commands to configure parameters. The commands to modify configuration parameters are described below. The generic syntax for the configuration parameters are:

SET CONFIGURATION=<parameter_1> <parameter_2> … <parameter_n>\r

with a space between each parameter and a Carriage Return (‘r’ or 0x0D) at the end of the SET command for example. Mandatory parameters are listed in “( )” optional parameters are listed in “[ ]”.

General Configuration Commands

Command	Description
CONFIG	Shows all configuration registers.
GET	Reads the value of a config parameter.
RESTORE	Restore the Configuration parameters to factory settings
SET	Sets a new value to a configuration parameter.
WRITE	Store configurations.in Flash (Permanent Memory).

Note: If the module boots up with the VOLUME-UP button pushed (PIO_3 HIGH on the IDC747 module), the factory default configurations are reset. This allows reverting to a known and working configuration state if severe problems are encountered.

Note that once modified, many configuration parameters will not take effect before a reboot: They need to be stored to Flash before rebooting with the WRITE command.

Table3 below lists all the available commands and a general description of each command.

Table4 describes each configuration parameter in details with examples.

Table 3: List of Configuration Parameters

Configuration Parameter	General Description
AUDIO	Selects the Audio interface
AUDIO_DIGITAL	Digital Audio Interface Settings
AUDIO_ANALOG	Analog Audio Interface Settings
AUTOCONN	Auto-Connection Feature
AUTO_DATA	Ability to Go Automatically in Data Mode on connection
BATT_CONFIG	Configuration of Battery Operating Zones
BCAST_CONFIG	Configuration of the Broadcast Channel (Only IDC777/767)
BCAST_META	Meta Data (name) of Auracast Subgroup BIS
BLE_CONFIG	BLE Data Configuration
BLE_CUSTOM_DB	Enables Customer Custom DB
BT_CONFIG	Bluetooth State at Boot-Up
CMD_TO	Guard Time for Escape Sequence
COD	Class Of Device
CODEC	Audio Codecs Enabled
DEEP_SLEEP	Enable/disable Chip Deepsleep
GPIO_CONFIG	Configuration for GPIO control
HFP_CONFIG	HFP and AGHFP Configuration
LINK_CONFIG	Policy for Master/Slave on Connection
LOCAL_ADDR	Bluetooth Address of Device (Read Only)
MIX_AUDIO	Mixes Audio in specific multiconnection scenarios
NAME	Bluetooth Name of Device
NAME_BCAST	Name of Broadcast Channel (Only IDC777/767)
NAME_SHORT	Bluetooth Low Energy Name of Device
MUSIC_META_DATA	Enable/Disable AVRCP Meta Data

Configuration Parameter	General Description
MIX_AUDIO	Mixes Different Audio Sources
POWERMAX	Max Transmit power in dBm
PROFILES	Bluetooth Profiles Enabled
SSP_CAPS	Secure Simple Pairing Capabilities
SPP_UUID	SSP Profile UUID
UART_CONFIG	UART Interface Settings
UCAST_CONFIG	Unicast Configuration Parameters
UI_CONFIG	Configuration of the UI (Prompts, Tones)
VOLUME_CONTROL	Management of Absolute Volume

Table 4: Detailed Description of Configurations

Configuration	Description
AUDIO	Selects the Audio Interface

Description: Selects which Audio Interface to use (Analog or Digital)

Syntax: AUDIO=<input> <output>
Note(s): If the optional parameter is not used, the output is the same as the input. Please refer to the module schematics to locate the Audio digital output Pins

Default: AUDIO=0 0

Parameter(s):
<input>: (Audio Input Interface)

- 0-Analog
- 1-Digital

<output>: (Audio Output Interface)

- 0-Analog
- 1-Digital

Reboot: Not Required

AUDIO_ANALOG	Audio Analog Interface Settings

Description: Configure the Digital Audio Interface Settings

Syntax: AUDIO_ANALOG=<input_gain> <output_gain> <mic_bias> <enable_preamp>

Note(s):
-This parameter is only taken in consideration if AUDIO=0 0 (an Analog interface is selected)
-This parameter is applied at initialisation. It can be overridden by the VOLUME command during operation (ex: VOLUME 0 2).

Default: AUDIO_ANALOG=15 15 1 OFF

Parameter(s):
<input_gain>: (Codec Input Gain)

- Decimal format (3dB steps)
- Valid range: 0-22

<output_gain>: (Codec Output Gain)

- Decimal format (3dB steps)
- Valid range: 0-22

<mic_bias>: (Mic_Bias Configuration)

- 0-Disabled
- 1-Turned on with Audio

Configuration	Description

<enable_preamp>: (Enable microphone pre-amp)

- ON-Enabled
- OFF-Disabled

Reboot: Not Required

AUDIO_DIGITAL	Digital Audio Interface Settings

Description: Configure the Digital Audio Interface Settings

Syntax: AUDIO_DIGITAL=<format><rate><param1><param2><param3>

Note(s):
-This parameter is only taken in consideration if AUDIO=1 (a Digital Interface is selected)

Default: AUDIO_DIGITAL=0 44100 64 01100100 00000000

Parameter(s):
<format>: (Digital format)

- 0-I2S
- 1-PCM
- 2-SPDIF

<rate>: (Digital sampling rate in Hertz -Samples/Second)

- For I2S: 8000, 16000, 32000, 44100 or 48000
- For PCM: 8000, 16000, 32000, 44100 or 48000
- For SPDIF: 8000, 16000, 32000, 44100 or 48000

<param1>: (rate parameter – depends on Digital format)

- For I2S: Bit Clock (BCLK) scaling factor – 64, 128 or 256
- For PCM: Bit Clock Rate
- For SPDIF: UNUSED

<param2>: (settings – depends on Digital format) - Four-byte (8 Hex) value

For I2S:

Bit Position	Description	Valid Values
First Byte	Master or Slave (1: Master)	00 or 01
Second Byte	Bits Per Sample in Hex	10, 14 or 18
Third Byte	Left Justify Bit Delay (00 MSB occurs in first SCLK period following WS transition. 01 MSB of SD Data occurs in the second SCLK period).	00 or 01
Fourth Byte	Enable 24-bit Audio Output	00 or 01

For PCM:

Bit Position	Description	Valid Values
First Byte	Master or Slave (1: Master)	00, 01
Second Byte	Sample Format (0: 13 bits in 16-bit slot, 1: 16 bits in 16-bit slot, 2: 8 bits in 16-bit slot, 3: 8 bits in 8-bit slot)	00, 01, 02, 03

Configuration	Description

Third Byte	Short Sync Enable (0: long frame sync - rising edge = start of frame, 1: short frame sync - falling edge = start of frame)	00, 01
Fourth Byte	Slot Count (0: derived from clock, 1-4: number of slots present between sync pulses)	01, 02, 03, 04

For SPDIF: UNUSED

<param3>: (settings – depends on Digital format) - Four-byte (8 Hex) value

For I2S:

Bit Position	Description	Valid Values
First Byte	Audio Atten Enable (0: 17-bit SD data is rounded down to 16 bits, 1: the attention specified by Audio Atten is applied over 24 bits with saturated rounding)	00, 01
Second Byte	Channel Polarity (0: Left when WS is high, 1: Right when WS is high)	00, 01
Third Byte	TX Start Sample (0: during low wclk phase, 1: during high wclk phase)	00, 01
Fourth Byte	RX Start Sample (0: during low wclk phase, 1: during high wclk phase)	00, 01

For PCM:

Bit Position	Description	Valid Values
First Byte	LSB First Enable (0: most significant bit first, 1: least significant bit first)	00, 01
Second Byte	Long Length Sync Enable (0: sets PCM_SYNC length to 8 PCM_CLK cycles, 1: sets length to 16 PCM_CLK cycles) N.B. Only applies if Short Sync Enable = 0.	00, 01
Third Byte	Sample Rising Edge Enable (0: sample PCM_DATA on falling edge of PCM_CLK, 1: sample PCM_DATA on rising edge of PCM_CLK)	00, 01
Fourth Byte	Audio Gain (valid range: 0-7, used to pad the end 3 bits of a 13-bit PCM sample)	01, 02, 03, 04, 05, 06, 07

For SPDIF: UNUSED

Reboot: Required

AUTOCONN	Auto-Connection Feature

Description: Enables/Disables the Auto-Connection at Boot-Up

Syntax: AUTOCONN=<type> <max_rec>
Note(s): If type=1, the module will try on RESET to reconnect to all devices on the PDL starting by the last one that was Paired (most recent device paired). For each device, it will try to reconnect MAX_REC times. Also, as it tries to re-connect, it will not be Discoverable/Connectable to by other devices. So if you boot the module with AUTOCONN enabled, you might not be able to connect to it for the time while it tries to Autoconnect (even if you have already paired with it). The IDC777 will send a UART message (RECONNECT ATTEMPT x) as it tries to reconnect. Note that after finishing to try and reconnect, the IDC777 goes back to DISCOVERABLE[IDLE]. So a command needs to be sent to make it DISCOVERABLE.

Configuration	Description
Default: AUTOCONN=0 3 Parameter(s): <type>: (Auto-Connection Feature) • 0—OFF. Auto-connection disabled. • 1—Auto-connect to all devices in the Paired Device List (PDL) <max_rec> (Max number of attempts it will try to connect with each device in the PDL) • Decimal value – 1 to 7 (7 Means unlimited attempts). **Reboot:** Required – Needs to be saved with WRITE command before RESET to take place next boot-up	
AUTO_DATA	Ability to go into Data Mode on Connection
Description: Configure module to automatically enter Data mode when BLE and/or SPP connections are established. **Syntax:** AUTO_DATA=<spp> <ble> **Default:** AUTO_DATA=OFF OFF Parameter(s): <spp>: Enter Data Mode when an SPP connection is made • OFF-Default • ON <ble> Enter Data Mode when an BLE connection is made • OFF-Default • ON **Reboot:** Not Required	
BATT_CONFIG	Battery Configuration
Description: Sets Default Battery Configuration **Syntax:** BATT_CONFIG=<battery> <charger> <temp> <critical_volt> <max_volt> <trickle current> <max_current> <max_temp> **Default:** BATT_CONFIG =OFF OFF OFF 3000 4200 10 1470 45 200 **Note(s):** If <temp> is enabled but no thermistor is connected to AIO0, the module will read a wrong value from the PIN and disable charging. The thermistor table is in the Appendix at Thermistor_Table. If <max_current> is greater than 200, the module assumes there is an external charger circuitry **Parameter(s):** <battery>: The module is being powered by battery • ON or OFF <charger >: Enables notifications when a charger is plugged in or out.	

Configuration	Description

- ON or OFF

<temp>: The module is connected to a thermistor to monitor the battery temperature

- ON or OFF

<critical_volt>: Voltage where the module will shut down.

- 1500 to 5000 (in mV)

<max_volt>: Voltage where the module will stop charging. Battery Full

- 1500 to 5000 (in mV)

<trickle current>: Charging current when the battery is below <critical_volt>

- 0 to 1500 (in mAmp)

<max_current>: Max Charging current when the battery is between <critical_volt> and <max_volt>

- 0 to 1500 (in mAmp)

<max_temp>: Max Temperature when battery will stop charging

- 0 to 120 (in degree centigrade)

<current_fast>: Max charging Current using the internal charging circuitry.

BCAST_CONFIG	Configuration of the Broadcast Channel

Description: Sets Default Broadcast configuration

Syntax: BCAST_CONFIG= <bcast_id > <bcast_cap1> <bcast_cap2> <bcast_cap3> <pdel> <pub> <bis> <subgroup>

Notes:
-For some configurations, ADVERTISING OFF command on the source (transmitter) side for some receivers to see the broadcasting channel. Note for QOS 48_6_2, 48_4_1, 48_5_1 and 48_6_1, ADVERTISING must be turned OFF on the transmitter or receiver for it to work.
-For QOS 24_x_x, if Digital Audio is used, 44100 rate should be changed to 48000. 44100 is not supported with 24 sets and Digital. For example, change AUDIO_DIGITAL=0 44100 64 00100100 00000000 to AUDIO_DIGITAL=0 48000 64 00100100 00000000
-For <subgroup>=2, QOS 48_1_1, 48_5_1, 48_6_1 and 48_6_2 is not supported. For <subgroup=3>, QOS 48_2_1, 48_2_2, 48_3_1, 48_4_1, 48_4_2 and 48_6_2 are not supported. For <subgroup=4>, 48_2_1, 48_2_2, 48_4_1, 48_4_2, 48_5_1 and 48_6_2 are not supported.
-For <subgroup=4>, The first 2 Audio mono streams (BIS1 and BIS2) are taken by what is configured in AUDIO and the 3rd and 4th Audio streams (BIS3 and BIS4)are from the other interface. For example, if AUDIO=0 0, BIS1 is Analog Left, BIS2 is Analog Right, BIS3 is Digital Left and BIS 4 is Digital Right.

Default: BCAST_CONFIG =223344 48 2 2 40000 OFF 2 1

Parameter(s):

Configuration	Description
<bcast_id>: (Broadcast ID) • XXXXXX (X=1 to F). Note that in production bcast_id should be a random number generated by the host to avoid 2 transmitters having the same broadcast ID <bcast_cap1>: (Broadcast Capability) • 16, 24, 32, 48 (Refer to Bluetooth spec BAP Profile or QOS COnfigurations) <bcast_cap2>: (Broadcast Capability) • 1, 2, 3, 4, 5, 6 (Refer to Bluetooth spec BAP or QOS COnfigurations) <bcast_cap1>: (Broadcast Capability) • 1, 2 (Refer to Bluetooth spec BAP Profile or QOS COnfigurations) <pdel>: (Pdelay – Delay in micro seconds between packet reception and Audio playback) • Between 20000 and 40000 depending on QOS configuration <pub>: (Enable Public Broadcast) • ON, OFF (Refer to Bluetooth spec) <bis> • 1, 2 (number of BIS) <subgroup> • 1, 2, 3, 4 (number of subgroups – 3 and 4 only used with bis=1) **Reboot:** Required	
BCAST_META 1, 2, 3 or 4)	Meta Data for broadcast Channels
Description: Sets the Met Date for each Auracast Subgroup **Syntax:** BCAST_META1=<name> **Default:** BCAST_META1=Italian BCAST_META2=French BCAST_META3=English BCAST_META4=German Parameter(s): <name>: (Maximum 12 bytes) • Name of Broadcast Audio Stream	
BLE_CONFIG	BLE configuration
Description: Sets Default Bluetooth Low Energy configuration	

Configuration	Description

Syntax: BLE_CONFIG= <mtu> <notification> <private> <host> <controller> <advertising_flag> <advertising_type> <bd_addr >

Default: BLE_CONFIG =250 ON OFF 2 0 1A 0 245DFC020000

Parameter(s):
<mtu>: (Maximum unit size in bytes used for GATT exchanges)

- Range: 23-250
- STATUS will give the negotiated mtu over the link

<notification>: (Notification or Indication used when using the SEND command>

- ON (Default)-Use Indication (Slower Data rate with confirmation packets)
- OFF-Use Notification (Faster Data rate with no confirmation packets)

<private>: (Enables BLE private address)

- ON-(Default)-Enabled
- OFF-Disabled

<host>:

- None – 0
- Static – 1
- Resolvable – 2
- Non-Resolvable - 3

<controller>:

- None – 0
- RPA - 1

<advertising_flag>:

BLE_FLAGS_LIMITED_DISCOVERABLE_MODE	0x01
BLE_FLAGS_GENERAL_DISCOVERABLE_MODE	0x02
BLE_FLAGS_SINGLE_MODE	0x04
BLE_FLAGS_DUAL_CONTROLLER	0x08
BLE_FLAGS_DUAL_HOST	0x10

<advertising_type>

- 0 - Undirected (Every device can see the advertisement)
- 1 – Only Allow devices in the PDL to connect (Only Bonded devices can connect)
- 2 – Directed advertising (Directed advertisement to the specific device with BT address <bd_addr>).

<bd_addr>: Bluetooth Address for directed Advertisement

- 12 hex value (e: 245DFC010209)

BLE_CUSTOM_DB	Custom Server BLE Database

Description: This enables a customer custom BLE Database. So when a Smart Phone connects for example, the IDC777 can emulate any custom Database (For example Heart Rate Monitor or any

Configuration	Description
Proprietary Push to Talk Database). This replaces the previous BLE_SET_DB command. Please contact IOT747 for more information. The proprietary Services and characteristics will then be exposed and the host can control or respond with the BLE_XXX command (In Peripheral Mode). **Syntax:** BLE_CUSTOM_DB =<value> **Default:** BLE_CUSTOM_DB=OFF Parameter(s): <value>: • ON • Default: OFF **Reboot:** Required	
BT_CONFIG	Bluetooth State at Boot-Up
Description: Sets Default Bluetooth state at boot-up and after Connection/Disconnection **Syntax:** BT_ CONFIG=<connectable> <discoverable> <advertising> **Note(s):** If BT_CONFIG=1 1 1, this means that the module will boot discoverable and advertising. But when a device connects to it and closes the connection, it will go back to IDLE mode (i.e. not DISCOVERABLE, not ADVERTISING). The commands DISCOVERABLE or ADVERTISING has to be sent to put it back in DISCOVERABLE or ADVERTISING modes. Being connectable/discoverable and advertising takes Radio bandwidth and power. So if there are multiple connection and advertising/discoverable are ON, the Audio channel might be affected. **Default:** BT_STATE_CONFIG=1 1 1 Parameter(s): <connectable>: (Allows other devices to connect) • 0—OFF. Not connectable at boot-up • 1— Connectable at boot-up <discoverable>: (Allows other devise to pair) • 0—OFF. Not discoverable at boot-up • 1—Discoverable at boot-up • 2—Always Discoverable <advertising>: (Advertising Bluetooth low Energy) • 0—OFF. Not advertising at boot-up • 1—Advertising at boot-up • 2—Always Advertising **Reboot:** Required – Needs to be saved with WRITE command before RESET to take place next boot-up	
CMD_TO	Guard Time for Escape Sequence ($$$$)
Description: Set the escape guard time. The guard time is the (empty) time between the last character and the escape sequence “$$$$”. The guard times also applies after the escape sequence to be recognised.	

Configuration	Description
Syntax: CMD_TO=<value> **Default:** CMD_TO=20 Parameter(s): <value>: • Valid 0—256. In 20ms increments • Default: 20 or 400ms **Reboot:** Not Required	
COD	Class Of Device
Description: Sets the Class of Device **Syntax:** COD=<cod> **Default:** COD=2C0404 Parameter(s): <cod > Class of Device • In Hex **Reboot:** Not Required	
CODEC	Enable/Disable SnapDragon Sound CODECs
Description: Enable or Disables SnapDragon Sound Codecs **Syntax:** CODEC =<aptx_rx> <aptxhd_rx> <aptxloss_rx> <aac_rx> <aptx_tx> <aptxhd_tx> <aptxloss_tx> <aptxlite_tx> <aptxlite_rx> **Default:** CODEC= OFF OFF OFF OFF OFF OFF OFF OFF OFF **Notes:** aptX Adaptive (or Lossless) and aptX Lite (Low Latency) requires and additional license from Qualcomm. Check with IOT747. aptX Lite is needed for ultra low latency (Gaming). Parameter(s): <aptx_rx > (Enables aptx Classic Codec in Receive mode (A2DP Sink mode) • ON-Default-Enabled • OFF-Disabled <aptxhd_rx > (Enables aptx HD Codec in Receive mode (A2DP Sink mode) • ON-Default-Enabled • OFF-Disabled <aptxloss_rx > (Enables aptx Lossless Codec in Receive mode ((A2DP Sink mode) • ON-Default-Enabled • OFF-Disabled	

Configuration	Description
<aac_rx > (Enables AAC (iOS) Codec in Receive mode ((A2DP Sink mode) • ON-Default-Enabled • OFF-Disabled <aptx_tx > (Enables aptx Classic Codec) in Transmit mode (A2DP Source) • ON-Default-Enabled • OFF-Disabled <aptxhd_tx > (Enables aptx HD Codec) in Transmit mode (A2DP Source) • ON-Default-Enabled • OFF-Disabled <aptxloss_tx > - OFF – Not implemented <aptxlite-tx > (Enables aptx Low Latency) in Unicast Transmit mode • ON-Default-Enabled • OFF-Disabled <aptxlite_rx > (Enables aptx Low Latency) in Unicast Receive mode • ON-Default-Enabled • OFF-Disabled	
GPIO_CONFIG	GPIO Configuration
Description: Enables or Disables GPIO_CONFIG. Please refer to GPIO Functionality for more details. **Syntax:** GPIO_CONFIG=<gpio_config> <pio31_enabled> <pio31_event_bitmask> **Default:** GPIO_CONFIG= ON OFF 00 Parameter(s): <gpio_config > (gpio configuration) • ON-Default-Enabled • OFF-Disabled <pio31_enabled> (is pio31 enabled) • ON- Enabled • OFF-Default-Disabled <pio31_event_bitmask> (gpio configuration) • Hexadecimal value • Bit 0 – A2DP_STREAM_START notification received • Bit 1 – A2DP_STREAM_SUSPEND notification received • Bit 2 – AVRCP_PLAY notification received • Bit 3 – AVRCP_PAUSE or AVRCP_STOP notification received • Bit 4 – CALL_INCOMING notification received • Bit 5 – CALL OUTGOING notification received • Bit 6 – LINK_LOSS notification received • Bit 7 – Data Mode exited	

Configuration	Description
• Bit 8 – High if connected, Low if not connected • Bit 9 – AVRCP_MEDIA notification received • Bit 10 – CALL_END or CALL_ACTIVE notification received **Reboot:** Not Required	
DEEP_SLEEP	Enable/Disable Chip Deepsleep
Description: Enables or Disables Module Deepsleep. **Syntax:** DEEP_SLEEP=<deepsleep_config><sniff_config> **Default:** DEEP_SLEEP=OFF ON Parameter(s): <deepsleep config> • ON-Enabled • OFF-Default-Disabled <sniff_config> • ON-Default-Enabled • OFF- Disabled **Reboot:** Required	
HFP_CONFIG	HFP/AGHFP Configuration
Description: Configures HFP and AGHFP. **Syntax:** HFP_CONFIG=<enable_aec><enable_at_notifications><aghfp_version><enable_wbs><enable_inband> <enable_outband><enable_aec_aghfp> **Default:** HFP_CONFIG= ON OFF 5 ON OFF ON OFF **Notes:** Wide Band Speech is only available from HFP Version 1.6. When using AGHFP, AEC should be usually disabled as it might interact with the AEC of the headset you are connecting to. Parameter(s): <enable_aec> (Enable Echo Cancellation Algorithm) • ON-Default-Enabled • OFF-Disabled <enable_at_notifications> (Enable AT Notification) • OFF-Default-Disabled • ON-Enabled <aghfp_version> (Version of AGHFP to use) • 0-HSP (Headset Profile) • 1-HFP version 1.0	

Configuration	Description
• 2-HFP version 1.5 • 3-HFP Version 1.6 • 4-HFP Version 1.7 • 5-HFP Version 1.8 <enable_wbs> (Enable Wide Band Speech Codec) • ON-Default-Enabled • OFF-Disabled <enable_inband> (Enable In-Band Ringing) • OFF-Default-Disabled • ON-Enabled <enable_outband> (Enable Out-Band Ringing) • ON-Default-Enabled • OFF-Disabled <enable_aec_aghfp> (Enable Echo Cancellation Algorithm in AGHFP mode) • ON-Default-Enabled • OFF-Disabled **Reboot:** Required	
LINK_CONFIG	Role Change Policy
Description: Role Change Policy. This is used in multiconnection scenarios if the IDC777 would like to automatically have the Master Role in the Bluetooth connection **Syntax:** LINK_CONFIG=<autoset><refuserolechange> **Default:** LINK_CONFIG=OFF OFF Parameter(s): <autoset> (Automatically requests MASTER role on connection) • ON-Enabled • OFF-Default-Disabled <refuserolechange > (Refuse Role Switch Request) • OFF-Default-Disabled • ON-Enabled **Reboot:** Required	
LOCAL_ADDR	Bluetooth Address of Device (Read Only)
Description: Local Device Bluetooth Address **Syntax:** LOCAL_ADDR=<bd_addr> **Note(s):** The Bluetooth Address is Unique for each module and is a read only parameter. The host cannot change it.	

Configuration	Description
Default: Not Applicable. On some modules, the Bluetooth Address is printed on a label that is placed on the module under the heading MAC-ID along with other module inform such as Build number. Parameter(s): <bd_addr> (Local Bluetooth address) • 12 hex value (e: 245DFC010209)	
MIX_AUDIO	Mixing Audio on Multiconnection
Description: This allows to mix Audio in specific multiconnection scenarios. If enabled, these only will work in isolation for the specific scenario. **Syntax:** MIX_AUDIO =<a2dpmix><hfpmix><2xaghfp> **Default:** MIX_AUDIO =OFF OFF OFF <a2dpmix> (Mixes Audio when 2 phones (or A2DP sources) are streaming music to the IDC777). • OFF-Default – Not enabled – If a Phone a streaming, the other is Paused • ON-Enabled – Both can stream and the Audio is mixed. <hfpmix> (Mixes Audio when 2 phones (or AGHFP) have an active call with IDC777). The IDC777 mic input is also then forwarded to both phones. • OFF-Default – Not enabled – If a Call is active, the other is rejected. • ON-Enabled – 2 SCO/calls can be active at the same tiem and Audio is mixed <2xaghfp> (IDC777 can have 2 AGHFP connection simultaneously with 2 headsets) • OFF-Default – Not enabled – Only one AGHFP SCo/call allowed • ON-Enabled – First AGHFP uses the Left Channels and Second AGHFP uses the right **Note:** When 2xaghfp is ON and AUDIO=0 0, CLOSE ALL cannot be used. You have to CLOSE each Link by its handle.	
MUSIC_META_DATA	Enable/Disable AVRCP Meta Data
Description: Enables/Disables AVRCP MetaData. This will show the Album title, Artist and other information every time a song is played or there is a forward or backward action. **Syntax:** MUSIC_META_DATA=<value><playback> **Default:** MUSIC_META_DATA=OFF 255 <value> (Display meta data when it changes – next song or forward/backward). • OFF-Default – Not enabled • ON-Enabled <playback> (Show positionin song every <playback> second) • Default – 255s	

Configuration	Description
MIX_AUDIO	Mixes Different Audio Sources

Description: This implements specific scenarios where the IDC777 will mix Audio coming from different sources.

Syntax: MIX_AUDIO=<mix A2DP> <mix HFP> <mix AGHFP>

Default: MIX_AUDIO=OFF OFF OFF

<mix A2DP>: ON or OFF. Is the module is connected to 2 A2DP Sources (ex: 2 Smart Phones) and one phone is already playing music.

- OFF: Default. If the other phone plays music, the phone is Paused by the module
- ON: If the other phone plays music, the IDC mixes music coming from both phones

<mix HFP>: ON or OFF. Is the module is connected to 2 AGHFP (ex: 2 Smart Phones) and one phone is already in a call.

- OFF: Default. If the other phone has a call incoming, it will not be answered
- ON: If the other phone has a call incoming, the module will mix audio from both phones

<mix AG-HFP>: ON or OFF. Is the module is connected to 2 AGHFP (ex: 2 headsets).

- OFF: Default. Only one AGHFP link can be active at the same time
- ON: The module can have 2 parallel mono audio AGHFP link. The first headset will use the left channels (IN/Out) and the second headset will use the right channel (In/Out).

Reboot: Required

POWERMAX	Max Transmit power in dBm

Description: Get/set maximum transmit power in dBm.

This Configuration is not implemented on IDC777 and IDC767. Please contact IOT747 for a custom firmware build if different power settings are required.

Syntax: POWERMAX=<value>

Default: POWERMAX=11

<value> (Maximum transmit power in dBm).

- Not implemented

Reboot: Required

NAME	Shows List of Paired Devices (LPD)

Description: Bluetooth Friendly Name of Device

Configuration	Description
Syntax: NAME=<name> **Default:** AA-XXXXXX (Where XXXXXX are the last digits of LOCAL_ADDR). Parameter(s): <name> (Device Bluetooth Friendly Name) • String of characters. Max: 32 characters. **Reboot:** Required	
NAME_BCAST	Name of Broadcast Channel (Auracast)
Description: Name under which the moduel will Broadcast Audio **Syntax:** NAME_BCAST=<name_bast> **Default:** LE-AA-XXXXXX (Where XXXXXX are the last digits of LOCAL_ADDR. Parameter(s): <name_bast> (Name of Broadcast Channel) • String of characters. Max: 12 characters. **Reboot:** Required	
NAME_SHORT	Bluetooth Low Energy Name of Device
Description: Bluetooth Low Energy Short Friendly Name of Device **Syntax:** NAME_SHORT=<name_short> **Note:** NAME_SHORT should be usually set as the same as the NAME on the LE Audio modules (IDC777) so it appears as the same module on Android 13+ Bluetooth Settings. **Default:** AA-XXXXXX (Where XXXXXX are the last digits of LOCAL_ADDR. Parameter(s): <name_short> (Device Bluetooth Low Energy Short Friendly Name) • String of characters. Max: 12 characters. **Reboot:** Required	
PROFILES	Bluetooth Profiles Enabled
Description: Bluetooth Profiles Enabled **Syntax:** PROFILES= <Multiconnection> <hfp><aghfp><a2dp_snk><a2dp_src><avrcp><ble><spp> **Default:** PROFILES=1 0 1 0 1 1 1 Parameter(s): <Multiconnection> (Enable Multiconnection)	

Configuration	Description
• ON or OFF <hfp> Max number of HFP Profiles allowed to be connected simultaneously • Range: 0 to 2 <aghfp> Max number of HFP-AG Profiles allowed to be connected simultaneously • Range: 0 to 2 <a2dp_snk> Max number of A2DP SINK Profiles allowed to be connected simultaneously • Range: 0 to 2 <a2dp_src> Max number of A2DP SOURCE HFP Profiles allowed to be connected simultaneously • Range: 0 to 2 <avrcp> Max number of AVRCP Profiles allowed to be connected simultaneously • Range: 0 to 2 <ble> Max number of BLE connections simultaneously • Range: 0 to 3 <spp> Max number of SPP connections simultaneously • Range: 0 to 3 **Reboot:** Required	
SSP_CAPS	Secure Simple Pairing Capabilities
Description: Configures Secure Simple Pairing Capabilities **Syntax:** SSP_CAPS=<io_caps> <autoearbud> **Default:** SSP_CAPS=3 OFF (Just Works – No Display – No Keyboard, No Auto Earbud detection) Parameter(s): <io_caps>: SSP Capabilities • 0- Display Only • 1-Display, Yes/No • 2-Keyboard Only • 3-No Display, No Keyboard – Just Works <autoearbud>: Detect LE earbuds automatically • ON- On pairing, discover/pair with matching earbud (CSIP: Coordinated Sets) • OFF- Do not automatically discover matching earbud. <autoearbud> should be left at OFF unless connecting Unicast to Earbuds Notes: MITM (Man In The Middle) is enabled when <io_caps> is 0,1 or 2. In this case PAIR_PASSKEY notifications are received and the user must accept or reject pairing using the PASSKEY command. **Reboot:** Not Required	
SPP UUID	SPP Profile UUID

Configuration	Description
Description: Set the SPP profile UUID (Universally Unique Identifier) **Syntax:** SPP_UUID=<uuid> **Default:** SPP_UUID=00 00 11 01 AA EF 43 51 90 34 D6 62 68 E3 28 FA Parameter(s): <uuid> (SPP UUID) • Hex format – 16 bytes **Reboot:** Required for Flow Control	
UCAST_CONFIG_TXM	Set Unicast Configuration for Music
Description: This sets the Unicast Configuration for Music/Media. **Syntax:** UCAST_CONFIG=<latency> <rate> <set_type> **Default:** UCAST_CONFIG=0 48 1 Parameter(s): <latency> (Enable Low-Latency Gaming Mode > • 0-High Reliability • 1-Gaming Mode Enabled without Back-Channel • 2-Gaming Mode Enabled with Back-Channel <rate> (Sampling Rate) • 48, 32, 24, 16 <set_type> (eg. To configure 48_1 or 48_2 set See QOS) • 1, 2	
UCAST_CONFIG_TXV	Set Unicast Configuration for Voice
Description: This sets the Unicast Configuration for Voice. **Syntax:** UCAST_CONFIG=<rate> <set_type> **Default:** UCAST_CONFIG=32 2 Parameter(s): <rate> (Sampling Rate) • 48, 32, 24, 16	

Configuration	Description
<set_type> (eg. To configure 32_1 or 32_2 set See QOS) • 1, 2	
UI_CONFIG	UI Configuration

Description: This enables the Audio Prompts (ex: “Pairing”) and different tones (Start-Up tone, Connection Tone, etc.).

Syntax:
UI_CONFIG=<prompts><tones><ring><led><upgradespp><upgradeble><discoverable><auto_poweron>

Note(s): Tones are specific and cannot be customised. Refer to TONE command to generate custom tones. This config is called ENABLE_UI in some releases.

Default: UI_CONFIG=OFF OFF ON ON ON ON 0 ON

Parameter(s):

<prompts> (Enable prompts>
- OFF-Prompts Disabled
- ON-Prompts Enabled

<tones> (Enable tones>
- OFF-Tones Disabled
- ON-Tones Enabled

<ring> (Enable ring even if tones are disabled>
- OFF-Ring Disabled
- ON-Ring Enabled

<led> (Enable Leds>
- OFF--Leds Disabled
- ON-Leds Enabled

<upgradespp> (Upgrade over SPP (Android)>
- OFF-SPP upgrade Disabled
- ON-SPP upgrade Enabled

<upgradeble> (Upgrade over BLE (iOS)>
- OFF-BLE upgrade Disabled
- ON-BLE upgrade Enabled

<discoverable> (Period that the device is discoverable>
- 0=Always Discoverable
- 1-255-In seconds. Time it stays discoverable before going IDLE (to reduce power consumption)

<auto_poweron> (Powers ON automatically when in Limbo)
- ON=Device auto powers ON when entering Limbo State (Default).
- OFF=Device does not Auto Power when entering Limbo State

See POWER command for more information

Configuration	Description
UART_CONFIG	UART Interface Settings

Description: UART Interface Settings

Syntax: UART_CONFIG=<baudrate><flow_ctrl><parity>

Default: UART_CONFIG=9600 OFF 0

Parameter(s):
<baudrate > (UART Baud rate)

- 9600,19200,38400,57600,115200,230400,460800,961200

<flow_ctrl > (Hardware Flow Control)

- OFF-Disabled – This parameter cannot be changed
- For enabling Flow Control, please contact IOT747 for a custom firmware.

<parity > (UART parity)

- 0-None
- 1-Odd parity
- 2-Even

Reboot: Required

VOLUME_CONTROL	Management of Absolute Volume

Description: The Module will receive volume notification from the connected device. This configuration allows the module to automatically apply the volume to its gain settings (Digital or Audio) or to keep the volume at maximum level and pass the volume information over the UART for the host to handle the volume change. Disabling volume control can be used for high-end Audio equipment that uses their own Amplifier to optimise SNR and Dynamic Range.

Syntax: VOLUME_CONTROL =<enable_automatic_volume > <music_volume_limit> <enable_a2dp_auracast_volume>

Default: VOLUME_CONTROL=ON 127 ON

Parameter(s):
<enable_automatic_volume>: (Enable automatic volume adjustment)

- 1- Default: Enable Automatic Volume Control – Module adjusts automatically volume
- 0-Disables Automatic Volume Control – Module does not adjust automatically volume

<music_volume_limit>: (Maximum volume for A2DP link)

- Default: 127
- 0-127- Maximum Volume on A2DP Link. The module will not apply a higher volume if if the volume requested internally or by the device connected is higher.

< enable_a2dp_auracast_volume >: (Enable automatic volume adjustment when in party mode)

- 1- Default: Enable Automatic Volume Control – Adjusts automatically volume on receivers
- 0-Disables Automatic Volume Control – Does not adjust automatically volume on receivers

This is when IDC777 received an A2DP Stream and Auracast for multi-Speaker scenario (Party Mode).

Configuration	Description
Reboot: Not Required	

AudioAgent Notifications

AudioAgent sends Notifications over the UART to notify the Host of events happening in the Bluetooth Link.

The generic syntax for Notifications are:

<event> [<link_id>] <parameters>

with a space between each parameter and a Carriage Return (‘r’ or 0x0D) at the end. S

Note: AudioAgent sends Responses to Commands over the UART as well. These are described for each Command in the Command section. We list here the Notifications that are not solicited.

Table5 below lists all the possible notifications

Table6 describes each notification in details.

Table 5: List of Notifications

Notification	General Description
A2DP_STREAM_START	A2DP Audio Stream Opened
A2DP_STREAM_SUSPEND	A2DP Audio Stream Suspended
ABS_VOL	AVCP Absolute Volume Change
AT	AT Command
AVRCP_BACKWARD	AVRCP Backward
AVRCP_FORWARD	AVRCP Forward
AVRCP_MEDIA	AVRCP Meta Data
AVRCP_PAUSE	AVRCP Pause
AVRCP_PLAY	AVRCP Play
AVRCP_STOP	AVRCP STOP
BLE_INDICATION	GATT Indication Received
BLE_NOTIFICATION	GATT Notification Received
BLE_READ	GATT Read Request
BLE_WRITE	GATT Write Request
CALL_ACTIVE	HFP/AGHFP Active Call Notification
CALL_DIAL	Dial Number Request from Remote HF
CALL_END	Call Termination Notification
CALL_INCOMING	Incoming Call Notification
CALL_OUTGOING	Outgoing Call Notification
CLOSE_OK	Close Connection Success
ERROR	Error
LINK_LOSS	Bluetooth Connection Lost
OPEN_OK	Open Connection Success
OPEN_ERROR	Open Connection Failure
PAIR_ERROR	Pairing Error

Notification	General Description
PAIR_OK	Pairing Successful
PAIR_PASSKEY	User Action Required to Pair
PAIR_PENDING	Pairing in Progress
RECV	Received Data Over the BLE Link
SCO_OPEN	SCO (Audio) Link Open
SCO_CLOSE	SCO (Audio) Link Closed

Table6: Detailed Description of Notifications

Notification	General Description
A2DP_STREAM_START	A2DP Audio Stream Opened
Description: A2DP Audio Stream opened. This notification appears when the remote (or local) device has requested the state of the A2DP to change. **Syntax:** A2DP_STREAM_START <link_id> Parameter(s): <link_id>: • 8-bit hex value (See Link ID Management) • Will be a <link_id> from an A2DP profile (ex: 10, 20)	
A2DP_STREAM_SUSPEND	A2DP Audio Stream Suspended
Description: A2DP Audio Stream suspended. This notification appears when the remote (or local) device has requested the state of the A2DP to change. **Syntax:** A2DP_STREAM_SUSPEND <link_id> Parameter(s): <link_id>: • 8-bit hex value (See Link ID Management) • Will be a <link_id> from an A2DP profile (ex: 10, 20)	
ABS_VOL	AVRCP Absolute Volume Change
Description: The current Absolute Volume has been changed. This notification appears when the remote (or local) device has requested a volume change. **Syntax:** ABS_VOL <link_id> <volume> Parameter(s): <link_id>: • 8-bit hex value (See Link ID Management) • Will be a <link_id> from an AVRCP profile (ex: 11, 21) <volume>: (Absolute Volume) • Decimal value between 0 and 127 for A2DP and 0 to 15 for HFP	
AT	AT Command
Description: Remote device sent an AT command or AT reply **Syntax:** AT <linkid><size><command> Parameter(s): <link_id>: • 8-bit hex value (See Link ID Management) • Will be a <link_id> from an HFP or AGHFP profile (ex: 12, 13) <size>: (length of the command) • Decimal value – Valid range: 1-65535 <command>: (AT Command) • ASCII string – AT command sent by remote device	

Notification	General Description
AVRCP_BACKWARD	AVRCP Backward

Description: AVRCP Backward has been received from the remote device
Syntax: AVRCP_BACKWARDS <link_id>
Parameter(s):
<link_id>:

- 8-bit hex value (See Link ID Management)
- Will be a <link_id> from an AVRCP profile (ex: 11, 21)

AVRCP_FORWARD	AVRCP Forward

Description: AVRCP Forward has been received from the remote device
Syntax: AVRCP_FORWARD <link_id>
Parameter(s):
<link_id>:

- 8-bit hex value (See Link ID Management)
- Will be a <link_id> from an AVRCP profile (ex: 11, 21)

AVRCP_MEDIA	AVRCP MetaData

Description: AVRCP Forward has been received from the remote device
Syntax: AVRCP_MEDIA <link_id> <type>: <data>
Parameter(s):
<link_id>:

- 8-bit hex value (See Link ID Management)
- Will be a <link_id> from an AVRCP profile (ex: 11, 21)

<type>: Meta Data type

- TITLE
- ARTIST
- ALBUM

<data>: Meta data

- ASCII string

AVRCP_PAUSE	AVRCP Pause

Description: AVRCP Pause has been received from the remote device
Syntax: AVRCP_PAUSE <link_id>
Parameter(s):
<link_id>:

- 8-bit hex value (See Link ID Management)
- Will be a <link_id> from an AVRCP profile (ex: 11, 21)

AVRCP_PLAY	AVRCP PLAY

Description: AVRCP Play has been received from the remote device

Notification	General Description
Syntax: AVRCP_PLAY <link_id> Parameter(s): <link_id>: • 8-bit hex value (See Link ID Management) • Will be a <link_id> from an AVRCP profile (ex: 11, 21)	
AVRCP_STOP	AVRCP STOP
Description: AVRCP Stop has been received from the remote device **Syntax:** AVRCP_STOP <link_id> Parameter(s): <link_id>: • 8-bit hex value (See Link ID Management) • Will be a <link_id> from an AVRCP profile (ex: 11, 21)	
BLE_INDICATION	GATT Indication Received
Description: GATT Indication received from Peripheral Device **Syntax:** BLE_INDICATION <link_id> <handle> <size> <data> **Parameter(s):** <link_id>: BLE Link Identifier • 8-bit hex value (See Link ID Management) <handle>: Handle • 4 digit Hex format. ex: 000C <size>: Size of Notification in bytes • Alphanumeric <data>: Characteristic value • ASCII string in Hex. eg: AA BB	
BLE_NOTIFICATION	GATT Notification Received
Description: GATT Notification received from Peripheral Device **Syntax:** BLE_NOTIFICATION <link_id> <handle> <size> <data> Parameter(s): <link_id>: BLE Link Identifier • 8-bit hex value (See Link ID Management) <handle>: Handle • 4 digit Hex format. ex: 000C <size>: Size of Notification in bytes	

Notification	General Description
• Alphanumeric <data>: Characteristic value • ASCII string in Hex. eg: AA BB	
BLE_READ	GATT Read Request
Description: GATT Read Request received from Central Device **Syntax:** BLE_READ <link_id> <handle> Parameter(s): <link_id>: BLE Link Identifier • 8-bit hex value (See Link ID Management) <handle>: Handle • 4 digit Hex format. ex: 000C	
BLE_WRITE	GATT Write Request
Description: GATT Write request received from Central Device **Syntax:** BLE_WRITE <link_id> <handle> <size> <data> Parameter(s): <link_id>: BLE Link Identifier • 8-bit hex value (See Link ID Management) <handle>: Handle • 4 digit Hex format. ex: 000C <size>: Size of Notification in bytes • Alphanumeric <data>: Characteristic value ASCII string in Hex. eg: AA BB	
CALL_ACTIVE	HFP/AGHFP Active Call Notification
Description: Notification of Active Call on specified HFP/AGHFP link **Syntax:** CALL_ACTIVE <linkid> **Parameter(s):** <link_id>: (Link identifier for HFP or AGHFP) • 8-bit hex value (See Link ID Management)	
CALL_DIAL	Dial Number Request from remote HFP
Description: Notification that the HFP connected wants to establish an outgoing call woth a specified number **Syntax:** CALL_DIAL <linkid><number>	

Notification	General Description
Parameter(s): <link_id>: (Link identifier for HFP or AGHFP) 8-bit hex value (See Link ID Management) <number>: (Requested number to dial) ASCI string (ex: +4412234202528)	
CALL_END	Call Termination Notification
Description: Notification of Termination of Call on specified HFP/AGHFP link **Syntax:** CALL_END <linkid> Parameter(s): <link_id>: (Link identifier for HFP or AGHFP) 8-bit hex value (See Link ID Management)	
CALL_INCOMING	Incoming Call Notification
Description: Notification of incoming call on the HFP link **Syntax:** CALL_INCOMING <linkid> Parameter(s): <link_id>: (Link identifier for HFP or AGHFP) 8-bit hex value (See Link ID Management)	
CALL_OUTGOING	Outgoing Call Notification
Description: Notification of outgoing call from the AGHFP **Syntax:** CALL_OUTGOING <linkid> Parameter(s): <link_id>: (Link identifier for HFP or AGHFP) 8-bit hex value (See Link ID Management)	
CLOSE_OK	Close Connection Success
Description: The connection of on the Link has been closed **Syntax:** CLOSE_OK <link_id> <profile> Parameter(s): <link_id>: • 8-bit hex value (See Link ID Management) • Will be a <link_id> from an AVRCP profile (ex: 11, 21) <profile>: • A2DP, AVRCP, AGHFP, HFP, BLE	
ERROR	Error
Description: An error has happened **Syntax:** ERROR <error_code>	

Notification	General Description
Parameter(s): <error_code>: • Refer to Error Codes Section	
LINK_LOSS	Bluetooth Connection Lost
Description: The Bluetooth connection has been lost and the link has closed. **Syntax:** LINK_LOSS <link_id> <profile> Parameter(s): <link_id>: • 8-bit hex value (See Link ID Management) • Will be a <link_id> from an AVRCP profile (ex: 11, 21) <profile>: • A2DP, AVRCP, AGHFP, HFP, BLE	
OPEN_OK	Open Connection Success
Description: The connection of on the Link has been closed **Syntax:** OPEN_OK <link_id> <profile><bd_addr> Parameter(s): <link_id>: • 8-bit hex value (See Link ID Management) • Will be a <link_id> from an AVRCP profile (ex: 11, 21) <profile>: • A2DP, AVRCP, AGHFP, HFP, BLE <bd_addr> (Local Bluetooth address) • 12 hex value (e: 245DFC010209)	
OPEN_ERROR	Error Opening Connection
Description: Opening Profile with remote device has failed **Syntax:** OPEN_ERROR <bd_addr> <profile> [<reason>] Parameter(s): <bd_addr> (Local Bluetooth address) • 12 hex value (e: 245DFC010209) <profile>: • A2DP, AVRCP, AGHFP, HFP, BLE <bd_addr> (Local Bluetooth address) • 0: General failure (ex: Device was there but could not connect) • 1: No connection could be established (ex: Device not there) • 2: Couldn't connect because the link key was missing. (ex: Device unpaired)	

Notification	General Description
PAIR_ERROR	Pairing Error

Description: The Pairing with remote device has failed
Syntax: PAIR_ERROR <bd_addr>
Parameter(s):
<bd_addr> (Local Bluetooth address)

- 12 hex value (e: 245DFC010209)

PAIR_OK	Pairing Successful

Description: The Pairing with remote device has succeeded
Syntax: PAIR_OK <bd_addr>
Parameter(s):
<bd_addr> (Local Bluetooth address)
12 hex value (e: 245DFC010209)

PAIR_PASSKEY	User Action Required to Pair

Description: SSP Passkey request has been received.
Use the PASSKEY command to reply to this Notification.
Syntax: PAIR_PASSKEY <bd_addr> <type> [<passkey>]
Parameter(s):
<bd_addr>: Bluetooth Address

- 12 hex value (e: 245DFC010209)

<type>: Pairing Indication type

- 0 – Passkey Requested
- 1 – Confirmation Yes/No Requested
- 2 – Display passkey

<passkey>: Passkey to Display

- Passkey to display. Passkey is not supplied for type 0 (Passkey requested).

PAIR_PENDING	Pairing in Progress

Description: The Pairing process with remote device has started
Syntax: PAIR_PENDING
Parameter(s): None

RECV	Received Data Over the BLE Link

Description: Data has been received on the BLE Link
Syntax: RECV <link_id> <size><data>
Parameter(s):
<link_id>:

- 8-bit hex value (See Link ID Management)

Notification	General Description
• Will be a <link_id> from an AVRCP profile (ex: 11, 21) <size>: (Number of characters received) • Integer number <data>: (Received Data) ASCII String	
SCO_CLOSE	SCO link closed
Description: a SCO link had been closed on the HFP/AGHFP Link **Syntax:** SCO_CLOSE <linkid> **Parameter(s):** <link_id>: (Link identifier for HFP or AGHFP) 8-bit hex value (See Link ID Management)	
SCO_OPEN	SCO link opened
Description: a SCO link had been opened on the HFP/AGHFP Link **Syntax:** SCO_OPEN <linkid> **Parameter(s):** <link_id>: (Link identifier for HFP or AGHFP) 8-bit hex value (See Link ID Management)	

GPIO Functionality

The functionality of GPIOs depends on the GPIO_CONFIG parameter. By default, GPIO_CONFIG=ON and the module can work without the need of a host processor.

The table below shows the GPIO functionality associated with each GPIO if GPIO_CONFIG=ON:

GPIO IDC777	Type	Button Timing(a)	Description
PIN23 (PIO3)	Input	Short	Volume Up
PIN5 (PIO29)	Input	Short	Volume Down
PIN24 (PIO6)	Input	Short	Answer Call (HFP) – If incoming call End Call (HFP) 0 If call active or outgoing call AVRCP (Music) Play/Pause
PIN25 (PIO5)	Input	Short	AVRCP (Music) Backwards
		Long	AVRCP (Music) Rewind
PIN7 (PIO26)	Input	Short	AVRCP (Music) Forward
		Long	AVRCP (Music) Fast-Forward

a. Short – GPIO held and released after 50ms
   Long – GPIO held and released after 1s

If GPIO_CONFIG=OFF, the table below shows the GPIO functionality

GPIO IDC777	Type	Button Timing(a)	Description
PIN8 (PIO30)	Output	N/A	High if there is at least one connection Low Otherwise
PIN25 (PIO5)	Output	N/A	High if there is a BLE Connection Low Otherwise
PIN5 (PIO29)	Output	N/A	High if there is an SPP Connection Low Otherwise
PIN4 (PIO32)	Output	N/A	High if Audio is present coming from the Bluetooth Link (i.e. on a Call or Music Streaming)
PIN6 (PIO31)	Output	N/A	User can customise this PIN – Please refer to GPIO_CONFIG
PIN7 (PIO26)	Input	Rising	Enter Data Mode (TBI)
		Falling	Exit Data Mode (TBI)
PIN23 (PIO3)	Input	Rising	Close all Connections

LED Indications

LEDs provide two indication types:

- Event Pattern – Patterns play once only.
- State Patterns– Patterns repeat.

LED indications are enabled by default (See UI_CONFIG).

On IDC777, there are 3 LEDs. But the firmware uses only LED2 (PIN15) by default (Fast Flashing if in PAIRING or slow Flashing if Connected or IDLE). LED3 (PIN16) and LED4 (PIN20) are not used.

Restoring the Default Configuration

The default configuration can be restored using either of the following:

- Maintain PIN23 (PIO3) high (Press the Volume Up button on the IDC747 Diskit) while resetting the module
- Use the RESTORE command.

Communication with Apps

The module can communicate with an Android and iOS App using Bluetooth Low Energy. The module exposes many services and characteristics used for different purposes (Device firmware upgrade, etc.). For simple Data Communications, the following Service is used.

AudioAgent Smart Service

UUID: 0xbc2f4cc6aaef43519034d66268e328f0

Characteristics (Write/Notify)

UUID: 0x06d1e5e779ad4a718faa373789f7d93c

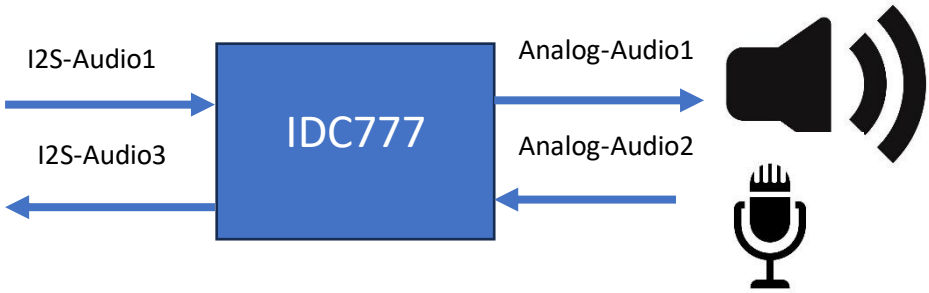
For testing and demo purposes, Sample Apps are available for download on the Apple Store and Google Play.

For integration, Source code for the AppStore and GooglePlay Applications is available. Please refer to https://www.iot747.com/software/ for the latest. Contact support@iot747.com for more information.

Echo Cancellation - CVC

IDC777 includes by default Qualcomm ® CVC echo cancellation, Automatic Gain Control and a Noise suppression algorithm. For more information, please refer to https://www.qualcomm.com/products/features/cvc-noise-cancellation-technology. CVC is automatically activated and used when a call on Bluetooth is active. It will remove noise and echo that is picked-up on the Microphone (or Line IN) to make the Audio better quality on the other end. The host can disable CVC with the config HFP_CONFIG. For example, it is recommended to disable CVC when using AGHFP (The module is the initiator and acting like a phone) since most headsets will have an echo cancellation algorithm running.

IDC777 also implements a feature that allows to use CVC even if there is no call on the Bluetooth link. This allows to use CVC on its own (for example during a wired call) using the I2S interface. Please see the diagram below and the ROUTE 37 Command.



Audio3 = Audio2 – Echo - Noise

AT Commands

Below you will find a list of some generally supported AT commands from the basic HFP profiles. These can be used with the AT command. These AT commands control the HFP and AGHFP profiles if you are not using the CALL commands. Please note that the module automatically answers when it receives some of these commands. The Syntax is AT <linkid> <ATCommand>. Note: Please refer to the Bluetooth HFP specifications for the latest commands available and Apple's Bluetooth design guidelines for the latest Apple specific commands available.

<ATCommand>	General Description
ATA	Call answer AT command.
ATDdd…dd;	Place a call to a phone number (dddddd)
ATD>nnn...	Memory Dialling
AT+CCWA	"Call Waiting notification". Within the AT+CCWA=[<n>[,<mode>[,<class>]]] command, only enabling/disabling of the Call Waiting notification unsolicited result code +CCWA , using the <n> parameter is covered.
AT+CHLD	Call hold and multiparty handling
AT+CHUP	Hang-up. Causes the AG to terminate the currently active call. This command shall have no impact on the state of any held call.
AT+CIND	Indicator update AT command.
+CIND	List of current phone indicators.
AT+CLCC	List current calls command.
+CLCC	List current calls result code
AT+COPS	AT+COPS=3,0 sets the format of the network operator string to the long format alphanumeric.
AT+CMEE	Enable the use of result code.
AT+CLIP	"Calling Line Identification notification". Enables/disables the Calling Line Identification notification unsolicited result code +CLIP.
+CLIP	"Calling Line Identification notification" unsolicited result code.
AT+CMER	Event reporting activation/deactivation AT command.

<ATCommand>	General Description
+CIEV	“Indicator events reporting” unsolicited result code.
AT+VTS	DTMF generation. Only the AT+VTS=<DTMF> command format covered.
AT+CNUM	AT+CNUM (Retrieve Subscriber Number Information)
+CNUM	Response used for sending the “Subscriber Number Information” from AG to HF.
AT+BIA	Bluetooth Indicators Activation. Command used by HF to activate / deactivate individual indicators.
AT+BINP	Requesting some specific data input from the AG4. On reception of this command the AG shall sent back requested information to the HF using the +BINP response.
AT+BLDN	Bluetooth Last Dialled Number
AT+BVRA	Bluetooth Voice Recognition Activation
+BVRA	Bluetooth Voice Recognition Activation. Unsolicited result code used to notify the HF when the voice recognition function in the AG is activated/deactivated autonomously from the AG.
AT+BRSF	(Bluetooth Retrieve Supported Features. Notifies the AG of the supported features available in the HF, and requests information about the supported features in the AG. Supported features represented as decimal value.
+BRSF	Bluetooth Retrieve Supported Features). Result code sent by the AG in response to the AT+BRSF command, used to notify the HF what features are supported in the AG. Supported features represented as decimal value.
AT+NREC	Command issued to disable any Echo Cancelling and Noise Reduction functions embedded in the AG. Only support for execution command is mandated. Neither the read nor test commands are mandatory.
AT+VGM	Gain of Microphone. Command issued by the HF to report its current microphone gain level setting to the AG. <gain> is a decimal numeric constant, relating to a particular (implementation dependent) volume level controlled by the HF. This command does not change the microphone gain of the AG; it simply indicates the current value of the microphone gain in the HF.
AT+VGS	Command issued by the HF to report its current speaker gain level setting to the AG. <gain> is a decimal numeric constant, relating to a particular (implementation dependent) volume level controlled by the HF. This

<ATCommand>	General Description
	command does not change the speaker gain of the AG; it simply indicates the current value of the speaker volume in the HF.
+VGM	Unsolicited result code issued by the AG to set the microphone gain of the HF. <gain> is a decimal numeric constant, relating to a particular (implementation dependent) volume level controlled by the HF.
+VGS	Unsolicited result code issued by the AG to set the speaker gain of the HF. <gain> is a decimal numeric constant, relating to a particular (implementation dependent) volume level controlled by the HF.
++BSIR	Unsolicited result code issued by the AG to indicate to the HF that the in-band ring tone setting has been locally changed. The HF may react accordingly by changing its own alert method.
AT+BTRH	Command issued by the HF for the “Response and Hold” feature in the AG. This specification defines the use of the set and read command. The AT+BTRH? command shall be used by the HF to query the current “Response and Hold” state of the AG
+BTRH	Result code used to notify the HF when-ever the incoming call is either put on hold or accepted or rejected. The AG shall also respond back with this response for the AT+BTRH? command from the HF.
AT+CPBS=?	Lists the phonebooks that the phone contains. (Choose phonebook storage). Returns: +CPBS: ("ME","SM","MT","ON","DC","MC","RC","EN","AD","QD","SD","FD")
+CPBS="ME"	sets the "retrieve mode" to the internal phonebook.
+CPBS="SM"	sets the "retrieve mode" to the SIM phonebook.
AT+CPBR	Describes the phonebook selected above. Gives the max number of entries the phone can contain and the maximum phone number (or email address) length and name length. You can substitute +MPBR for any +CPBR command, but the phone returns a more specific (less intelligible) response containing more fields that may act as internal “programming” flags of some sort. Returns: +CPBR: (1-1000),40,24
AT+CPBR=[beginning index],[ending index]	Returns list of numbers with the index between the two numbers entered. Denotes what TYPE of phonebook entry was selected. Returns: +CPBR: 9,"18005555555",129,"Contact Name" – 129 refers to a phone number. Returns: +CPBR: 18,"user@domain.net",128,"Contact Name" – 128 refers to an email.
AT+CPBR=[index]	Returns the specified index. Returns: +CPBR: 18,"user@domain.net",128,"Contact Name"
AT+MPBF="Name"	Searches the phonebook for the Name or string.

<ATCommand>	General Description
AT+MPBR=?	Similar to above, but more detailed result displayed. Returns: +MPBR: 1-1000,40,24,8,0-1,50,(0,2,4,6,9-30,255),(0),(0-1),(1-30),(255),25,(0-1,255),264,(0),0,0,0,0,0,0,0 ▪ 1-1000 - Number of entries that can be stored on the phonebook. ▪ 40 - Number of characters email or phone number can have. ▪ 24 - Number of characters the “friendly” name can have. ▪ 8 - Different “types” of phonebook entry (i.e. Mob, Main, Email, etc) The +CPBR command does not list anything after the 24 (as seen above), so there are times when +MPBR may be useful.
AT+MPBR=[index]	Returns: +MPBR: 18,"user@domain.net",128,"Contact Name",6,0,255,0,0,1,255,255,0,"",0,0,"","","","","","","",""
AT+CMGF=1	This tells the phone to display the entries as text rather than binary. +CMFG=0 would display the data in binary format.
AT+CPMS=?	his displays all of the locations in which the phone can save the SMS messages. Returns: +CPMS: ("MT","IM","OM","BM","DM"),("OM","DM"),("IM")
AT+CMGL=?	Returns the options on which messages you wish to display. Returns: +CMGL: ("REC UNREAD", "REC READ", "STO UNSENT", "STO SENT", "ALL")
AT+CMGL="ALL"	Selects and displays all of the SMS messages on the selected source.
AT#PMODE=1	In order to retrieve text messages and other information, Samsung phones must be in this mode.
AT#PSRMR=?	Returns the parameters to obtain text messages (Samsung). Returns: #PSRMR: (0-349)
AT+XAPL	Apple: Enables custom AT commands from a headset.
AT+IPHONEACCEV	Apple: Reports a headset state change.

BAP QOS Configurations

Low Latency

Set Name	Codec Capability / Configuration Setting (Table 3.5 and Table 3.11)	SDU_ Interval (µs)	Framing	Maximum_SDU_Size (Octets)	Retransmission_Number	Max_Transport_Latency (ms)	Presentation_Delay (µs)	Requirement	
								Unicast Client	Unicast Server
QoS Configuration settings for low latency audio data									
8_1_1	8_1	7500[1]	unframed	26[2] (27.734 kbps[3])	2[5]	8	40000[4]	C.1	C.2
8_2_1	8_2	10000[1]	unframed	30[2] (24 kbps[3])	2[5]	10	40000[4]	C.1	C.2
16_1_1	16_1	7500[1]	unframed	30[2] (32 kbps[3])	2[5]	8	40000[4]	C.1	C.2
16_2_1	16_2	10000[1]	unframed	40[2] (32 kbps[3])	2[5]	10	40000[4]	M	M
24_1_1	24_1	7500[1]	unframed	45[2] (48 kbps[3])	2[5]	8	40000[4]	C.1	C.2
24_2_1	24_2	10000[1]	unframed	60[2] (48 kbps[3])	2[5]	10	40000[4]	C.1	M
32_1_1	32_1	7500[1]	unframed	60[2] (64 kbps[3])	2[5]	8	40000[4]	C.1	C.2
32_2_1	32_2	10000[1]	unframed	80[2] (64 kbps[3])	2[5]	10	40000[4]	C.1	C.2
441_1_1	441_1	8163[6]	framed	97[2] (95.06 kbps[3])	5[5]	24	40000[4]	C.1	C.2
441_2_1	441_2	10884[6]	framed	130[2] (95.55 kbps[3])	5[5]	31	40000[4]	C.1	C.2
48_1_1	48_1	7500[1]	unframed	75[2] (80 kbps[3])	5[5]	15	40000[4]	C.1	C.2
48_2_1	48_2	10000[1]	unframed	100[2] (80 kbps[3])	5[5]	20	40000[4]	C.1	C.2
48_3_1	48_3	7500[1]	unframed	90[2] (96 kbps[3])	5[5]	15	40000[4]	C.1	C.2
48_4_1	48_4	10000[1]	unframed	120[2] (96 kbps[3])	5[5]	20	40000[4]	C.1	C.2
48_5_1	48_5	7500[1]	unframed	117[2] (124.8 kbps[3])	5[5]	15	40000[4]	C.1	C.2
48_6_1	48_6	10000[1]	unframed	155[2] (124 kbps[3])	5[5]	20	40000[4]	C.1	C.2

High Reliability

Set Name	Codec Capability / Configuration Setting (Table 3.5 and Table 3.11)	SDU_ Interval (µs)	Framing	Maximum_SDU_Size (Octets)	Retransmission_Number	Max_Transport_Latency (ms)	Presentation_Delay (µs)	Requirement	
								Unicast Client	Unicast Server
QoS Configuration settings for high-reliability audio data									
8_1_2	8_1	7500[1]	unframed	26[2] (27.734 kbps[3])	13[5]	75	40000[4]	C.1	C.2
8_2_2	8_2	10000[1]	unframed	30[2] (24 kbps[3])	13[5]	95	40000[4]	C.1	C.2
16_1_2	16_1	7500[1]	unframed	30[2] (32 kbps[3])	13[5]	75	40000[4]	C.1	C.2
16_2_2	16_2	10000[1]	unframed	40[2] (32 kbps[3])	13[5]	95	40000[4]	C.1	C.2
24_1_2	24_1	7500[1]	unframed	45[2] (48 kbps[3])	13[5]	75	40000[4]	C.1	C.2
24_2_2	24_2	10000[1]	unframed	60[2] (48 kbps[3])	13[5]	95	40000[4]	C.1	C.2
32_1_2	32_1	7500[1]	unframed	60[2] (64 kbps[3])	13[5]	75	40000[4]	C.1	C.2
32_2_2	32_2	10000[1]	unframed	80[2] (64 kbps[3])	13[5]	95	40000[4]	C.1	C.2
441_1_2	441_1	8163[6]	framed	97[2] (95.06 kbps[3])	13[5]	80	40000[4]	C.1	C.2
441_2_2	441_2	10884[6]	framed	130[2] (95.55 kbps[3])	13[5]	85	40000[4]	C.1	C.2
48_1_2	48_1	7500[1]	unframed	75[2] (80 kbps[3])	13[5]	75	40000[4]	C.1	C.2
48_2_2	48_2	10000[1]	unframed	100[2] (80 kbps[3])	13[5]	95	40000[4]	C.1	C.2
48_3_2	48_3	7500[1]	unframed	90[2] (96 kbps[3])	13[5]	75	40000[4]	C.1	C.2
48_4_2	48_4	10000[1]	unframed	120[2] (96 kbps[3])	13[5]	100	40000[4]	C.1	C.2
48_5_2	48_5	7500[1]	unframed	117[2] (124.8 kbps[3])	13[5]	75	40000[4]	C.1	C.2
48_6_2	48_6	10000[1]	unframed	155[2] (124 kbps[3])	13[5]	100	40000[4]	C.1	C.2

[1] Nominal. May be adjusted to accommodate audio clock offset and drift.

[2] Settings are based on a Unicast Client communicating with two Unicast Servers, with each Unicast Server being configured for a single Audio Channel and single block of codec frames per SDU (Service Data Unit). Different Audio_Channel_Allocation values (see Section 4.3.2) and/or a greater number of blocks of codec frames per SDU (see Section 4.3.2) would require settings to be appropriately scaled.

[3] Bit rates are calculated according to Section 3.2.5 in [7].

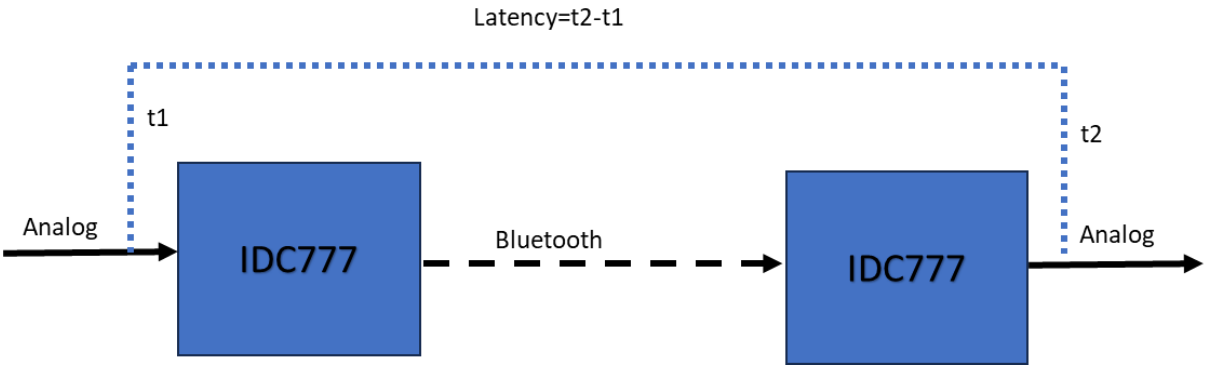
[4] For the Unicast Server, the supported Presentation_Delay range in the Codec Configured state shall include this value when the ASE is a Sink ASE.

[5] Retransmission_Number values are recommendations to the Controller, which may use different values to match desired robustness and/or bandwidth. The Host shall be capable of requesting the values listed.

[6] Effective SDU_Interval. For 44.1 kHz/7.5ms, the actual SDU_Interval is equivalent to 360 (samples per second) divided by 44100 (Sampling Frequency), which equals 8.16327 ms, and for 44.1 kHz/10 ms the actual frame duration is equal to 480 (samples per second) divided by 44100 (Sampling Frequency), which equals 10.88435. The LC3 [7] codec encodes 97 octets (for 7.5 ms/8.163 ms effective) or 130 octets (for 10 ms/10.884 ms effective) into each SDU, which arrives at the controller every 8.16327 ms or 10.88435 ms. The transmitting device assigns a time offset to each SDU and delivers the time_offset with each SDU at the receiver, as defined in Volume G, Part 7, Section 3.1 in [1]. Determination of the time_offset parameter at the transmitting device is implementation-specific. Compensation for the difference between 8.16327 ms and 8.163 ms, and/or compensation between 10.88435 ms and 10.884 ms, is implementation-specific.

Wireless Audio Latencies

We have made a few measurements between 2 IDC777-DISKITs. The measured delay is the delay between Audio Analog In and Audio Analog Out. See the diagram bellow. Note that these results are indicative as the latency will vary with the device that is being connected to. For lowest latency between 2 devices, the gaming mode can achieve 20ms delay.



Results:

Profile	Stereo	QOS	Pdelay	Latency
Auracast	Yes	48_2_2	40ms	130ms
Auracast	Yes	48_2_2	20ms	110ms
Auracast	Yes	48_2_1	40ms	80ms
Auracast	Yes	48_2_1	20ms	60ms
Unicast Media	Yes	N/A	N/A	200ms
Unicast Voice	No	N/A	N/A	45ms
Unicast Gaming APTX	Yes	N/A	N/A	20ms
A2DP (SBC Codec)	Yes	N/A	N/A	300ms
A2DP (aptx)	Yes	N/A	N/A	230ms
A2DP (aptx-hd)	Yes	N/A	N/A	230ms
HFP	No	N/A	N/A	65ms

Error codes

Error code	Description
0x0003	Unknown error

Command Errors

Error code	Description
0x0011	Command not allowed with the current configuration
0x0012	Command not found
0x0013	Wrong parameter
0x0014	Wrong number of parameters
0x0015	Command not allowed in the current state
0x0016	Device already connected
0x0017	Device not connected
0x0018	Command is too long
0x0019	Name not found
0x001A	Configuration not found

Warnings

Error code	Description
0x0100	Fail to read battery voltage
0x1003	Fail to configure cap sense
0x1004	Fail to register/unregister device
0x1005	BLE request failed
0xFF01	License key is missing
0xFF02	License key is invalid

Critical Errors

Error code	Description
0xF00X	Critical error
0xF004	Wrong config

Examples

Below you will find some examples of using the IDC747. These examples can be combined as the IDC777 allows multiple connections to multiple devices (Multiple BLE and Multiple Profile Classic Connections).

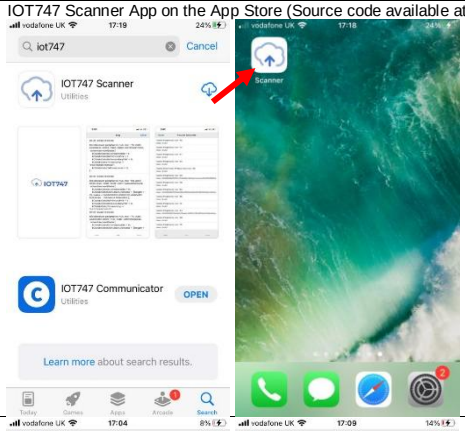
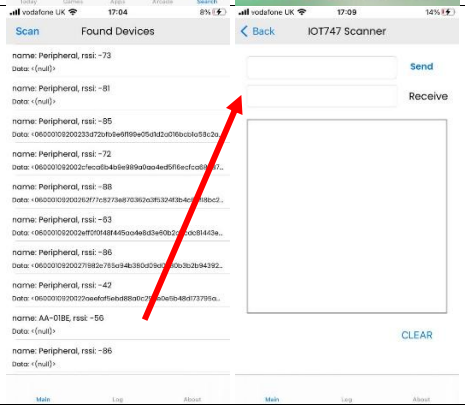
Classic: Streaming Music from IDC777 to IDC777 (or another Classic Headset) over A2DP

Description: Device A will open an A2DP Source Profile with Device B which is in A2DP Sink and Stream Audio to it		
Initial Configuration (Device A in Transmit Mode and Device B in Receive Mode)	Device A (IDC777: BT Add: xxx) *-Enable A2DP Source and AVRCP-* SET PROFILES=OFF 0 0 0 1 1 0 0 OK WRITE OK RESET IOT747 Copyright 2022 AudioAgent V2.0.26 Build: d0feae32 Bluetooth address 245DFC0001BE Ready PAIR_PENDING	Device B (IDC777: BT Add: 245DFC0001BF) *-Enable A2DP Sink and AVRCP-* SET PROFILES=OFF 0 0 1 0 1 0 0 OK WRITE OK RESET IOT747 Copyright 2022 AudioAgent V2.0.26 Build: d0feae32 Bluetooth address 245DFC0001BF Ready PAIR_PENDING
Establish Connection	*-Inquire about Devices in Proximity-* INQUIRY 5 PENDING INQUIRY 245DFC0001BF "AA-0001BF" 240404 -28 dBm INQUIRY 94E23CE8B92F "PID-LAP13" 2A010C -84 dBm …….. INQU_OK *-Pair and Connect with Device A Proximity-* PAIR 245DFC0001BF PENDING PAIR_OK 245DFC0001BF OPEN 245DFC0001BF A2DP PENDING OPEN_OK 10 A2DP OPEN_OK 11 AVRCP	*-Make Device A Discoverable-* DISCOVERABLE ON OK *-Accept Pairing and Connection-* PAIR_OK 245DFC0001BE OPEN_OK 10 A2DP OPEN_OK 11 AVRCP
Status	*-Check Connection Status-* STATUS STATE CONNECTABLE[ON] DISCOVERABLE[OFF] ADVERTISING[IDLE] LINK 10 CONNECTED A2DP 245DFC0001BF SUSPENDED SBC SRC 48000 LINK 11 CONNECTED AVRCP 245DFC0001BF STOPPED OK	*-Check Connection Status-* STATUS STATE CONNECTABLE[ON] DISCOVERABLE[IDLE] ADVERTISING[IDLE] LINK 10 CONNECTED A2DP 245DFC0001BE SUSPENDED SBC SNK 48000 LINK 11 CONNECTED AVRCP 245DFC0001BE STOPPED OK
Playback Control	 AVRCP_PLAY 11 A2DP_STREAM_START 10 *-Music Pause from Device B-* MUSIC 10 PAUSE AVRCP_PAUSE 11 OK A2DP_STREAM_SUSPEND 10 ABS_VOL 11 104 *-Set Volume to 16-* VOLUME 10 16 OK ABS_VOL 11 16 AVRCP_PLAY 11 A2DP_STREAM_START 10 AVRCP_FORWARD 11 AVRCP_BACK 11	*-Music Play from Device A-* MUSIC 10 PLAY OK A2DP_STREAM_START 10 AVRCP_PLAY 11 A2DP_STREAM_SUSPEND 10 AVRCP_PAUSE 11 *-Volume Up-* VOLUME 10 UP OK ABS_VOL 11 104 ABS_VOL 11 16 *-Music Forward -* MUSIC 10 PLAY OK A2DP_STREAM_START 10 AVRCP_PLAY 11 MUSIC 10 FORWARD OK *-Music Back -* MUSIC 10 BACKWARD OK

Classic: Establishing a Call Between IDC777 and IDC777 (or Headset) – HFP and AGHFP

Description: Configuring Device A to connect to HFP send HFP audio to Device B		
Initial Configuration	Device A (IDC747: BT Add: 245DFC0001BF) set profiles=on 0 1 0 1 1 1 1 OK write OK reset IOT777 Copyright 2022 AudioAgent V3.0.87 Build: 0400427b Bluetooth address 245DFC020000 Ready aptX Lossless: ON PAIR_PENDING	Device B (IDC747: BT Add: 245DFC010000) set profiles=on 1 0 1 0 1 1 1 OK write OK reset IOT777 Copyright 2022 AudioAgent V3.0.87 Build: 0400427b Bluetooth address 245DFC020020 Ready aptX Lossless: ON PAIR_PENDING
Establish Conn	*-Look for headsets in Pairing Mode-* inquiry 5 PENDING INQUIRY 245DFC0200FF "UNKNOWN" 240404 -43 dBm INQUIRY 245DFC020020 "UNKNOWN" 240404 -42 dBm INQU_OK *-Pair with Device B and Open HFP link-* pair 245DFC020020 PENDING PAIR_OK 245DFC020020 PAIR_OK_LE 245DFC020020 open 245DFC020020 aghfp PENDING OPEN_OK 12 AGHFP 245DFC020020 ABS_VOL 12 10 status STATE CONNECTABLE[ON] DISCOVERABLE[IDLE] ADVERTISING[ON] SCAN_UNI[OFF] LINK 12 CONNECTED AGHFP 245DFC020020 IDLE	*-Device A Paired with Device B-* PAIR_OK 245DFC020000 PAIR_OK_LE 245DFC020000 *-Device A opened HFP link with Device B-* OPEN_OK 13 HFP 245DFC020000 status STATE CONNECTABLE[ON] DISCOVERABLE[IDLE] ADVERTISING[ON] SCAN_UNI[OFF] LINK 13 CONNECTED HFP 245DFC020000 IDLE OK
Example: Call initiated from Device A, answered by Device B and ended by Device B (the headset)	*-Device A Receives a Call (Incoming Call)-* call 12 incoming 121 CALL_INCOMING 12 OK CALL_RINGING 12 CALL_RINGING 12 CALL_RINGING 12 CALL_ACTIVE 12 SCO_OPEN CALL_END 12 CALL_END 12 SCO_CLOSE	CALL_RINGING HFP 13 CALL_INCOMING HFP 13 *-Device B answers the call-* call 13 answer OK CALL_INCOMING_ENDED HFP 13 CALL_ACTIVE HFP 13 SCO_OPEN *-Device B ends the call-* call 13 end OK CALL_IDLE HFP 13 SCO_CLOSE

BLE: Connecting IDC777 to an App over BLE (to OT747 iOS Scanner App in this example)

Description: Connection to a SmartPhone and Sending/Receiving Data to an App		
Initial Configuration	Device A (IDC747: BT Add: 245DFC0001BF) *-Enable BLE Profile-* SET PROFILES=OFF 0 0 0 0 0 1 0 OK WRITE OK RESET IOT747 Copyright 2022 AudioAgent V2.0.26 Build: d0feae32 Bluetooth address 245DFC0001BF Ready PAIR_PENDING	IOT747 Scanner App on the App Store (Source code available at www.iot747.com) 
Establish Connection	*-Make Device A Advertising-* ADVERTISING ON OK *-Accept Connection -* OPEN_OK 14 BLE 637BB5A2920D	
Status	*-Check Connection Status-* STATUS STATE CONNECTABLE[ON] DISCOVERABLE[ON] ADVERTISING[OFF] LINK 14 CONNECTED BLE 637BB5A2920D MTU 250	
Playback Control	*-Receive Text from App -* RECV 14 12 Hello idc747 *-Send Text to App -* SEND 14 HELLO APP OK	

LE Audio: Auracast Transmit and Receive

Description: Configuring Device A to Auracast Audio from Analog Input and Device B to find the Auracast Channel and Play the Audio		
Initial Configuration	Device A (IDC777: BT Add: 245DFC020000) *-Restore to initial state-* restore IOT777 Copyright 2022 AudioAgent V3.0.87 Build: 0400427b Bluetooth address 245DFC020000 Ready aptX Lossless: ON PAIR_PENDING	Device B (IDC777: BT Add: 245DFC0200020) *-Restore to initial state-* restore IOT777 Copyright 2022 AudioAgent V3.0.87 Build: 0400427b Bluetooth address 245DFC020020 Ready aptX Lossless: ON PAIR_PENDING
Start Auracast Stream on Device A and Synch to it on Device B	*-Start Stream from Analog-* broadcast on 1 OK OPEN_OK 1C BTX1 245DFC020000	*-Scan for Auracast Streams and connectInquire about Devices in Proximity-* scan 5 off 2 PENDING SCAN 2 245DFC020000 0x223344 0xd AA-020000 -52 SCAN_OK open 245DFC020000 brx1 2 d 223344 OPEN_OK 1A BRX1 245DFC020000 OK music 1a play OK LC3_BROADCAST_NUMBER_BIS 2 LC3_BROADCAST_STREAM_CONNECT *-Audio should now be playing-*
Status	*-Check Connection Status-* status STATE CONNECTABLE[ON] DISCOVERABLE[ON] ADVERTISING[ON] SCAN_UNI[OFF] LINK 1C CONNECTED BTX1 245DFC020000 ANALOG PDEL 20000 RATE 48000 SUBGROUP 1 BISS 2 OK	*-Check Connection Status-* status STATE CONNECTABLE[ON] DISCOVERABLE[ON] ADVERTISING[ON] SCAN_UNI[OFF] LINK 1A CONNECTED BRX1 245DFC020000 0x223344 STREAMING LC3 PDEL 20000 RATE 48000 ENCR: 0 SUBGROUP 1 OK
Volume Control	*-Increase Volume Remotely of Device B -* volume 1c 100 OK	ABS_VOL 1A 100 *-Increase the Volume Locally on Device B -* volume 1a 50 OK

Thermistor Table

Voltage	Temp (in C)	Voltage	Temp (in C)	Voltage	Temp (in C)
1730	-40	1302	2	601	44
1725	-39	1286	3	587	45
1721	-38	1269	4	574	46
1716	-37	1252	5	561	47
1711	-36	1235	6	548	48
1706	-35	1218	7	535	49
1700	-34	1201	8	523	50
1695	-33	1183	9	511	51
1689	-32	1166	10	499	52
1682	-31	1148	11	487	53
1676	-30	1131	12	476	54
1669	-29	1113	13	465	55
1662	-28	1095	14	454	56
1654	-27	1077	15	443	57
1647	-26	1059	16	433	58
1639	-25	1041	17	423	59
1630	-24	1023	18	413	60
1622	-23	1005	19	403	61
1613	-22	988	20	394	62
1603	-21	970	21	384	63
1594	-20	952	22	375	64
1584	-19	934	23	366	65
1574	-18	917	24	358	66
1563	-17	900	25	349	67
1552	-16	882	26	341	68
1541	-15	865	27	333	69
1529	-14	848	28	325	70
1517	-13	831	29	317	71
1505	-12	814	30	310	72
1493	-11	798	31	302	73
1480	-10	781	32	295	74
1467	-9	765	33	288	75
1453	-8	749	34	282	76
1439	-7	733	35	275	77
1425	-6	718	36	269	78
1411	-5	702	37	262	79
1396	-4	687	38	256	80
1381	-3	672	39	250	81
1366	-2	657	40	244	82
1350	-1	643	41	239	83
1334	0	629	42	233	84
1318	1	614	43	228	85

Terms and definitions

Abbreviation	Description
A2DP	Advanced Audio Distribution Profile
AG	Audio Gateway
AVRCP	Audio/Video Remote Control Profile
BLE	Bluetooth Low Energy
cVc	Clear Voice Capture
DFU	Device Firmware Upgrade
HF	Hands-Free Unit
HFP	Hand-Free Profile
HID	Human Interface Device Profile
MAP	Message Access Profile
Multipoint	When more than one device is connected
PBAP	Phone Book Access Profile
PDL	Paired Device List
SPP	Serial Port Profile
WBS	Wide Band Speech