iWRAP^3

User Guide

Version 1.3

Friday, September 26, 2008

Contents:

1. Introduction 12

2. Getting Started 14

3. iWRAP Modes 15

3.1 Command Mode 17

3.2 Data Mode 17

3.3 Multiplexing Mode 18

3.4 HFP mode 18

3.5 OPP mode 18

3.6 A2DP mode 18

3.7 AVRCP mode 18

4. Technical Details 19

5. iWRAP Command listing 21

5.1 Typographical Conventions 26

5.2 AT 27

5.3 INQUIRY 28

5.4 IC 32

5.5 NAME 34

5.6 PAIR 35

5.7 AUTH 37

5.8 CALL 39

5.9 CLOSE 43

5.10 LIST 44

5.11 SELECT 49

5.12 SDP 50

5.13 SDP ADD 52

5.14 IDENT 53

5.15 L2CAP 54

5.16 RFCOMM 55

5.17 SCO ENABLE 56

5.18 SCO OPEN 57
5.19 CLOCK 59
5.20 KILL 60
5.21 BER 61
5.22 RSSI 62
5.23 TXPOWER 63
5.24 PING 64
5.25 ECHO 65
5.26 SLEEP 66
5.27 PIO 67
5.28 VOLUME 69
5.29 TEMP 70
5.30 BATTERY 71
5.31 BYPASSUART 72
5.32 DEFRAG 73
5.33 BCSP_ENABLE 74
5.34 RESET 75
5.35 BOOT 76
5.36 TEST 77
5.37 TESTMODE 80
5.38 HELP 81
5.39 INFO 82
5.40 CONNECT 84

6. SET 85

6.2 SET BT BDADDR 87
6.3 SET BT NAME 88
6.4 SET BT CLASS 89
6.5 SET BT AUTH 90
6.6 SET BT LAP 92
6.7 SET BT PAGEMODE 94
6.8 SET BT PAIR 96

6.9 SET BT ROLE 97
6.10 SET BT SNIFF 99
6.11 SET BT POWER 101
6.12 SET BT IDENT 103
6.13 SET CONTROL AUTOCALL 105
6.14 SET CONTROL BAUD 108
6.15 SET CONTROL CD 110
6.16 SET CONTROL CONFIG 111
6.17 SET CONTROL ECHO 114
6.18 SET CONTROL ESCAPE 115
6.19 SET CONTROL INIT 117
6.20 SET CONTROL MUX 118
6.21 SET CONTROL BIND 122
6.22 SET CONTROL MSC 124
6.23 SET CONTROL GAIN 126
6.24 SET CONTROL VREGEN 127
6.25 SET CONTROL MICBIAS 129
6.26 SET CONTROL PCM 131
6.27 SET PROFILE 132
6.28 SET {link_id} ACTIVE 134
6.29 SET {link_id} MASTER 135
6.30 SET {link_id} SLAVE 136
6.31 SET {link_id} SNIFF 137
6.32 SET {link_id} MSC 138
6.33 SET {link_id} SELECT 139

7. iWRAP Events 140

7.1 CONNECT 141
7.2 INQUIRY_PARTIAL 142
7.3 NO CARRIER 143
7.4 READY 144
7.5 NAME 145

7.6 NAME ERROR 146
7.7 PAIR 147
7.8 RING 148
7.9 SYNTAX ERROR 149
7.10 AUTH 150
7.11 CLOCK 151
7.12 IDENT 152
7.13 IDENT ERROR 153
7.14 BATTERY 154
7.15 PIO 155
7.16 VOLUME 156
8. iWRAP Error Messages 157
8.1 HCI Errors 157
8.2 SDP Errors 160
8.3 RFCOMM Errors 162
9. General Information 164
9.1 Changing Parameters over RS232 with PSTool 164
9.2 Using BlueTest over RS232 165
9.3 Switching to HCI Firmware 166
9.4 Firmware Updates over SPI 167
9.5 Firmware Updates over UART 167
9.6 UART Hardware Flow Control 168
9.7 RS232 Connections Diagram 169
9.8 Supported *Bluetooth* Profiles Overview 170
9.9 Power Saving 174
9.10 *Bluetooth* profile UUIDs 175
10. Known Issues 179
11. Troubleshooting 181
12. iWRAP USAGE Examples 182
12.1 Serial Port Profile – Slave mode 182
12.2 Serial Port Profile – Master mode 184

12.3 Dial-up Networking 185

12.4 OBEX Object Push Profile Server 186

12.5 Hands-Free Audio Gateway Connection to a Headset Device 188

12.6 Hands-Free connection to a Mobile Phone 192

12.7 iWRAP to iWRAP Serial Port Profile + SCO connection 195

12.8 Wireless IO Replacement 196

12.9 A2DP Sink 198

12.10 A2DP Source 199

12.11 AVRCP Connection 200

12.12 Human Interface Device (HID) Example 202

12.13 Over-the-Air Configuration 206

12.14 *Bluetooth* Networking with iWRAP and WRAP Access Server 207

13. Support 209

14. Related Documentation 210

15. Contact Information 211

List of Tables:

Table 1: iWRAP modes transitions explained 16

Table 2: Technical details 19

Table 3: Supported generic iWRAP commands 22

Table 4: Supported “SET BT” commands 23

Table 5: Supported link control commands 24

Table 6: Supported Bluetooth profile commands 24

Table 7: Supported “SET CONTROL” commands 25

Table 8: Power classes as defined in *Bluetooth* specification 102

Table 9: Multiplexing frame format 119

Table 10: HCI errors 159

Table 11: SDP errors 161

Table 12: RFCOMM errors 163

Table 13: UUIDs and Profiles 178

Table 14: Known issues in iWRAP 180

Table 15: Supported HFP-AG commands 189

Table 16: HFP-AG status messages 190

Table 17: Supported HFP commands 193

Table 18: HFP-AG status messages 194

Table 19: Available AVRCP commands 201

Table 20: Available HID key codes and mouse events 205

List of Figures:

Figure 1: iWRAP Stack 12

Figure 2: iWRAP boot prompt 14

Figure 3: Mode transitions 15

Figure 4: Host-iWRAP-Host communication 120

Figure 5: Host-iWRAP-remote device communications 120

Figure 6: MSC signal directions 125

Figure 7: RS232 connections 169

Figure 8: SPP slave example 182

Figure 9: SPP master example 184

Figure 10: How to open a DUN connection to a mobile phone 185

Figure 11: Receiving files through OPP 186

Figure 12: Receiving a vCard over OPP 187

Figure 13: iWRAP to headset Hands-Free connection 188

Figure 14: HFP connection to a mobile phone 192

Figure 15: Simultaneous SPP and SCO connection between two iWRAP modules 195

Figure 16: Wireless IO replacement connection 196

Figure 17: MSC signal directions 197

Figure 18: A2DP sink example 198

Figure 19: A2DP source example 199

Figure 20: AVRCP example 200

Figure 21: HID keyboard example 202

Figure 22: Raw HID keyboard report 205

Figure 23: Raw HID mouse report 205

Figure 24: Over-the-Air connection example 206

Figure 25: Configuration for multiple slaves 208

Version history

Version:	Author:	Comments:
1.0	MSa	iWRAP3 official user guide
1.1	MSa	Added “SET CONTROL CONFIG”
1.2	MSa	iWRAP3 updates
1.3	MSa	Added SET {link_id} SELECT

Terms & Abbreviations

Term or Abbreviation:	Explanation:
A2DP	Advanced Audio Distribution Profile
AIO	Analog Input Output
AVRCP	Audio/Video Remote Control Profile
BDR	Basic Data Rate
Bluetooth	Set of technologies providing audio and data transfer over short-range radio connections
bps	Bits per second
CD	Carrier Detect
DTR	Data Terminal Ready
DUN	Dial-Up Networking Profile
EDR	Enhanced Data Rate
EIR	Enhanced Inquiry Response
GPIO	General Purpose Input Output
HCI	Host Controller Interface

HFP	Hands-Free Profile
HFP-AG	Hands-Free audio Gateway
HID	Human Interface Device
iWRAP	Interface for WRAP – a trademark registered by Bluegiga Technologies
L2CAP	The Logical Link Control and Adaptation Layer Protocol
MSC	Modem Status Control
OBEX	OBject EXchange Protocol
OPP	Object Push Profile
RFCOMM	Serial cable emulation protocol; element of *Bluetooth*
SNIFF mode	*Bluetooth* low power mode
SPP	Serial Port Profile
UART	Universal Asynchronous Receiver Transmitter
UUID	Universally Unique Identifier
VM	Virtual Machine
WRAP	Wireless Remote Access Platform; Bluegiga Technologies' wireless product family

1. INTRODUCTION

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

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

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

The DSP processor is only available on WT32 *Bluetooth* module.

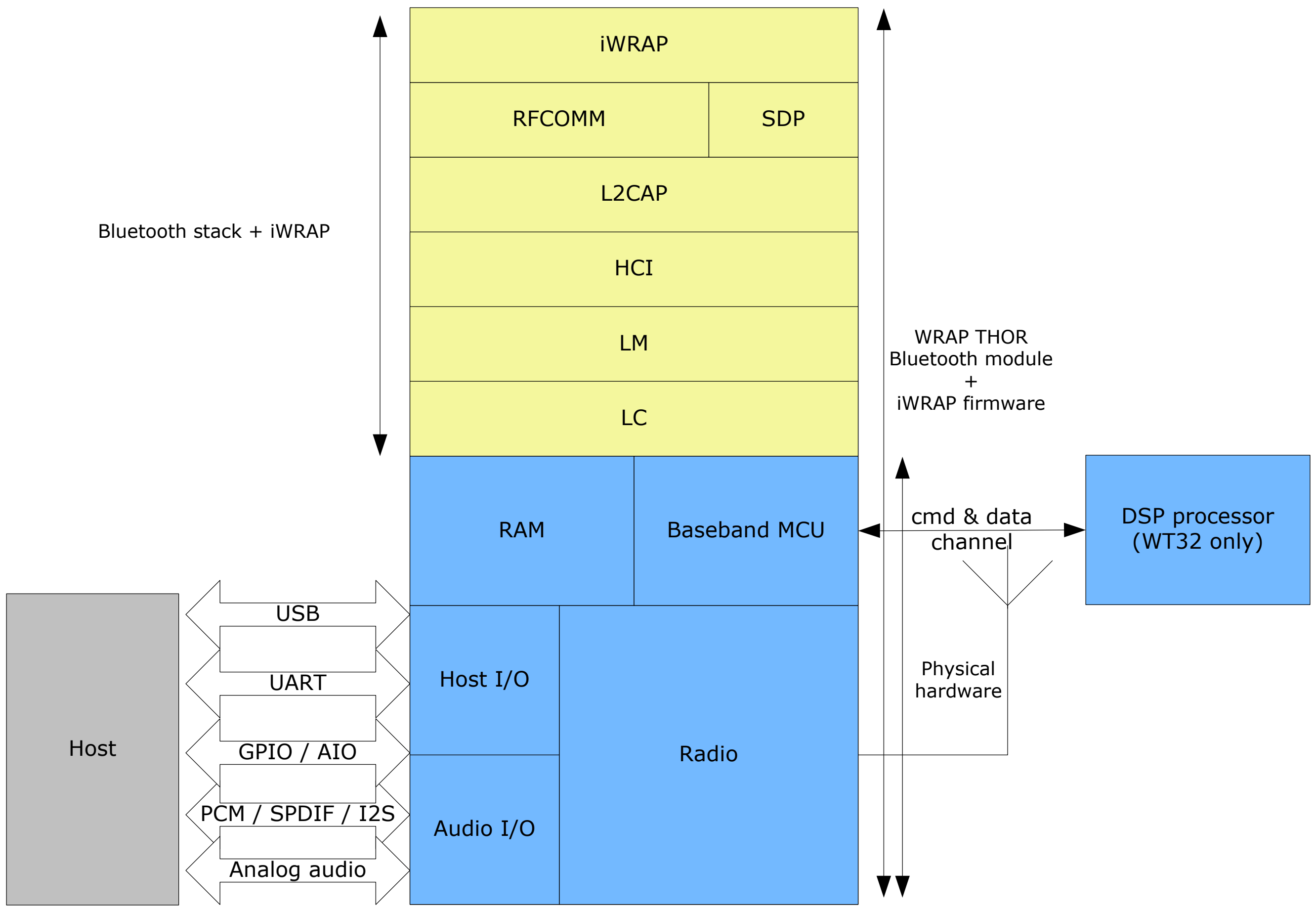


Figure 1: iWRAP Stack

In the figure above, a WRAP THOR *Bluetooth* module with iWRAP firmware could be connected to a host system for example via the UART interface.

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

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

3. GPIO lines that WRAP THOR modules offer can also be used together with iWRAP to achieve additional functionality, such as Carrier Detect or DTR signaling.

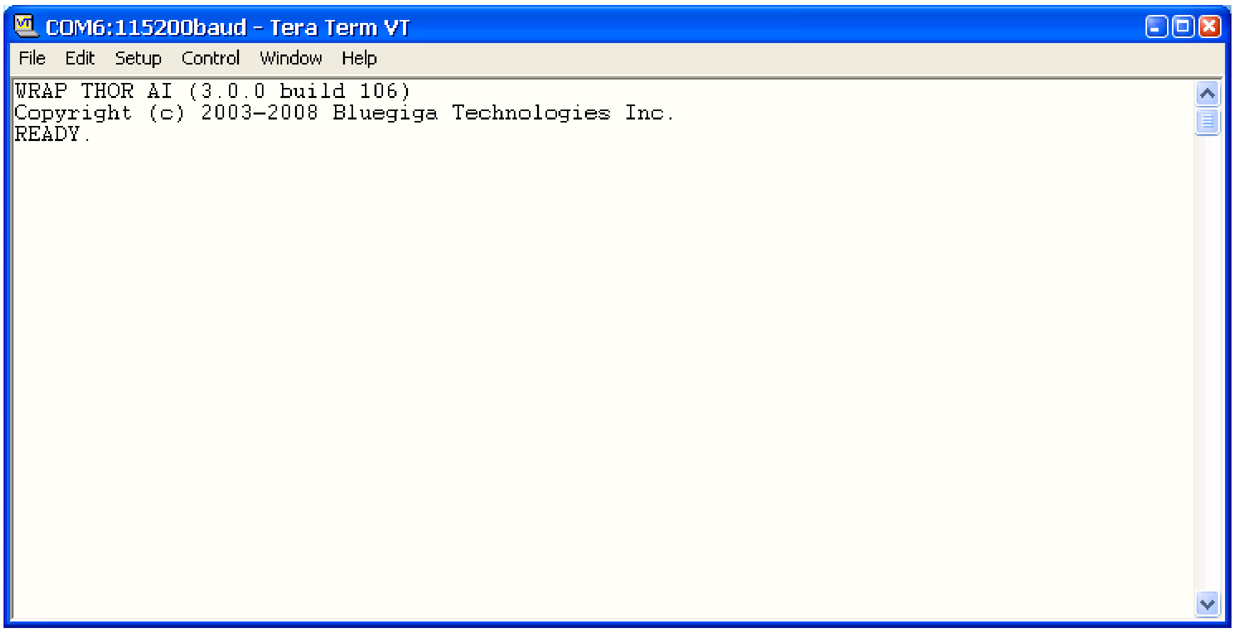
4. Audio interfaces can be used to transmit audio over a *Bluetooth* link.

2. GETTING STARTED

To start using iWRAP firmware, you can use, for example, terminal software such as *HyperTerminal*. When using the terminal software, make sure that the WRAP THOR module is connected to your PC's serial port. By default, iWRAP uses the following UART settings:

• Baud rate:	115200bps
• Data bits:	8
• Stop bits:	1
• Parity bit:	No parity
• HW Flow Control:	Enabled

When you power up your WRAP THOR module or evaluation kit, you can see the boot prompt appear on the screen of the terminal software. After the "**READY**." event iWRAP firmware is ready to be used.


COM6:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
WRAP THOR AI (3.0.0 build 106)
Copyright (c) 2003-2008 Bluegiga Technologies Inc.
READY.
```

Figure 2: iWRAP boot prompt

3. IWRAP MODES

iWRAP has two basic operational modes, **command mode** and **data mode**. In command mode ASCII commands can be given to iWRAP firmware to perform various actions or to change configuration settings and is the default mode when there are no *Bluetooth* connections. Data mode on the other hand is used to transmit and receive data over a *Bluetooth* link and it's only available if there is a *Bluetooth* connection. It is possible to switch between modes at any time assuming the conditions for data mode are fulfilled. The mode transitions are illustrated below.

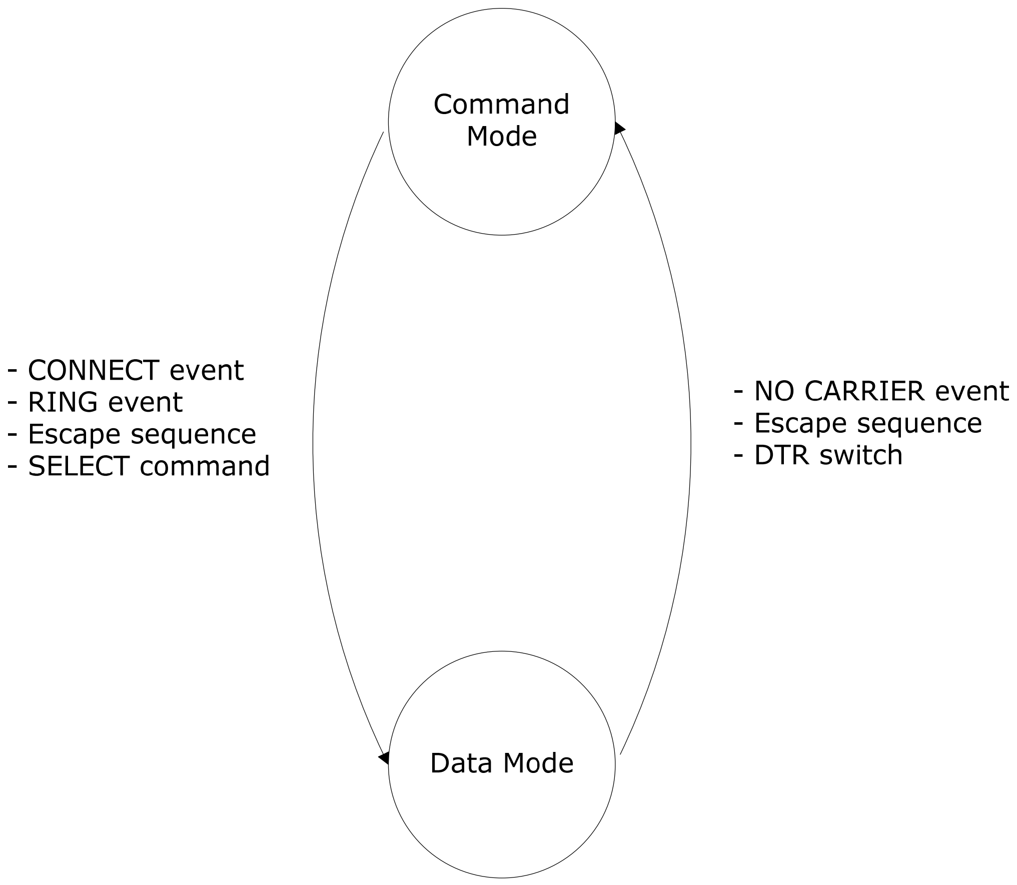


Figure 3: Mode transitions

Initial mode	**Target mode**	**Requirements for state transition**
Command Mode (no *Bluetooth* connections) In this mode, ASCII commands can be given to iWRAP.	Data Mode	A connection is successfully created by using the **CALL** command and **CONNECT** event that indicates a successful connection is received. A remote device opens a *Bluetooth* connection to iWRAP. A **RING** event that indicates a received connection is received.
Data Mode In this mode, all data is sent transparently from UART interface to *Bluetooth* connection.	Command Mode	The user switches mode by sending an escape sequence to iWRAP firmware or by toggling the DTR pin. A link is terminated (closed by the remote device or by link loss) and **NO CARRIER** event is received.
Command Mode (active connection) In this mode, ASCII commands can be given to iWRAP.	Data Mode	User switches the mode either by sending the escape sequence or command **SELECT** command.

Table 1: iWRAP modes transitions explained

The escape sequence:

The escape sequence causes the iWRAP firmware to go to the command mode from the data mode or vice versa. The escape sequence consists of three (3) escape characters that are defined by **SET CONTROL ESCAPE** command. By default the escape character is '+'.

Do not enter any character before and/or after the escape sequence for a guard time which is 1 second. Also send the escape characters individually, not as a string.

With default settings the escape sequence is:

< 1 second sleep> +++ < 1 second sleep>

When a successful state transition from data mode to command mode is made, iWRAP sends a "**READY.**" event to indicate that it's ready to receive commands.

The same escape sequence or the **SELECT** command can be used to return to data mode.

3.1 Command Mode

The command mode is the default mode when iWRAP is powered up. In command mode, ASCII commands can be entered to iWRAP to perform various functions.

Note:

- In command mode if there are active *Bluetooth* connections, the data from remote devices is buffered into iWRAP buffers.
- Because of the embedded nature of iWRAP, buffering capabilities are low and only small amounts of data can be received to buffers. The amount of data which can be buffered depends on the firmware version and the state of iWRAP. Usually, it is around 1000 bytes, but may vary radically.
- **LIST** command shows active connections, as well the amount of buffered data.

3.2 Data Mode

Data mode is the default mode when there are one or more *Bluetooth*. In data mode, all data is sent transparently from UART interface to the *Bluetooth* link and vice versa.

Note:

- When iWRAP enters command mode, a "**READY**" event occurs, unless events are masked away with "**SET CONTROL ECHO**" command.
- DTR pin can be used instead of escape sequence to switch from data mode to command mode and it allows much faster mode switching so no guard time is needed. DTR pin can be enabled with "**SET CONTROL ESCAPE**" command.
- The escape character can also be changed by with "**SET CONTROL ESCAPE**" command.
- Carrier Detect (CD) pin can also be used to indicate either a *Bluetooth* connection or data mode. CD pin can be enabled with "**SET CONTROL CD**" command.

3.3 Multiplexing Mode

In iWRAP version 2.1.0 and newer, there is a special mode called multiplexing mode. In this mode, iWRAP does not have separate commands or data modes, but data, commands and events are all handled in one single mode. There is, however, a special protocol to separate commands and events from the actual data. This protocol must be used between the host system and iWRAP firmware.

The advantage of this multiplexing mode is that several *Bluetooth* connections can be handled simultaneously and there is no need to do time consuming data-command-data mode switching. However the downside is that the performance of iWRAP is reduced, since the firmware needs to handle the multiplexing protocol and the overhead it causes.

To learn more about multiplexing mode, please see the description of "**SET CONTROL MUX**" command.

3.4 HFP mode

iWRAP 2.2.0 and newer support *Bluetooth* Hands-Free (v.1.5) profile. This profile include a lot of control messaging and events, which is handled in the command mode i.e. when a HFP connection is opened or received no state transition occurs, but iWRAP stays in the command mode, where all HFP messaging is done. Please refer to HFP profile usage for more information.

3.5 OPP mode

IWRAP 2.1.0 and newer support *Bluetooth* Object Push Profile (OPP) profile. The operation is this profile is quite similar to HFP mode i.e. there is no separate command and data modes, but iWRAP always stays in the command mode. In OPP mode, Please refer to OPP profile usage for more information.

3.6 A2DP mode

From iWRAP3 on *Bluetooth* Advanced Audio Distribution Profile (A2DP) is supported. This profile includes also control messaging and events, which is handled in the command mode i.e. when a A2DP connection is opened or received no state transition occurs, but iWRAP stays in the command mode, where all A2DP messaging is done.

3.7 AVRCP mode

From IWRAP3 on *Bluetooth* Audio/Video Remote Control Profile (AVRCP) is supported. This profile includes also control messaging and events, which is handled in the command mode i.e. when a AVRCP connection is opened or received no state transition occurs, but iWRAP stays in the command mode, where all AVRCP messaging is done.

4. TECHNICAL DETAILS

Feature:	**Value:**
MAX simultaneous ACL connections	4
MAX simultaneous SCO connections	1
MAX data rate	600Kbps (WT12/WT11 to BT2.0 USB dongle) 500Kbps (WT12/WT11 to WT12/WT11) 450Kbps (WT12/WT11 to BT1.1-BT1.2 device)
MAX UART baud rate	921600 bps
Typical data transmission delay	10-15ms
Minimum data transmission delay	5-10ms
Typical SCO delay	30-40ms
Typical A2DP delay*	150-200ms
A2DP coding/encoding methods	SBC, MP3**, AAC**, APT-x** and FastStream**
PIN code length	Configurable from 0 to 16 characters
Encryption length	Configurable from 0 to 128** bits
MAX simultaneous pairings	16
MAX Friendly name length	Configurable up to 248 characters
RFCOMM Packet size	Configurable from 21 to 1008
Supported *Bluetooth* profiles (iWRAP3)	GAP, SPP, Hands-Free Profile (V.1.5), A2DP, AVRCP, HID, DUN, DI, and OPP***
Supported power saving modes	Sniff and deep sleep
Bluetooth QD ID	iWRAP 3.0: B014328, iWRAP 2.2.0: B012647

Table 2: Technical details

**) Alternative coding methods (APT-x, FastStream) exist to reduce the delay to 40-60ms*

***) Custom firmware needs to be request from support@bluegiga.com*

****) Limited support*

5. IWRAP COMMAND LISTING

iWRAP can be used and controlled from the host system by sending ASCII commands through the UART interface to iWRAP.

When installed and configured, the module can be commanded from the host with the following ASCII commands:

Command:	**iWRAP version:**	**HW version:**	**Short description**
A2DP	iWRAP3.0	WT32	A2DP streaming control
AT	iWRAP 2.1.0	WT12, WT11, WT32	Attention
AUTH	iWRAP 2.2.0	WT12, WT11, WT32	Authenticates *Bluetooth* connection
BATTERY	iWRAP 3.0	WT32	Reads battery level
BCSP_ENABLE	iWRAP 3.0	WT12, WT11, WT32	Enables BCSP mode
BER	iWRAP 2.2.0	WT12, WT11, WT32	Reads Bit Error Rate
BOOT	iWRAP 2.2.0	WT12, WT11, WT32	Boots module into different modes
BYPASSUART	iWRAP 3.0	WT12, WT11, WT32	Enables UART bypass
CALL	iWRAP 2.1.0	WT12, WT11, WT32	Opens Bluetooth connections
CLOCK	iWRAP 3.0	WT12, WT11, WT32	Reads Piconet clock
CLOSE	iWRAP 2.1.0	WT12, WT11, WT32	Closes Bluetooth connections
CONNECT	iWRAP 3.0	WT12, WT11, WT32	Connects Bluetooth links
DEFRAG	iWRAP 3.0	WT12, WT11, WT32	Defrags PS key storage
ECHO	iWRAP 2.2.0	WT12, WT11, WT32	Echoes data to Bluetooth connection
HELP	iWRAP 2.2.0	WT12, WT11, WT32	Prints help
IC	iWRAP 2.2.0	WT12, WT11, WT32	Inquiry cancel
IDENT	iWRAP 3.0	WT12, WT11, WT32	Identifies a Bluetooth device
INFO	iWRAP 2.2.0	WT12, WT11, WT32	Prints firmware information
INQUIRY	iWRAP 2.1.0	WT12, WT11, WT32	Searches other Bluetooth devices

KILL	iWRAP 3.0	WT12, WT11, WT32	Kills Bluetooth connections
L2CAP	iWRAP 3.0	WT12, WT11, WT32	Sets up L2CAP psm
LIST	iWRAP 2.1.0	WT12, WT11, WT32	Lists Bluetooth connections
NAME	iWRAP 2.2.0	WT12, WT11, WT32	Does friendly name discovery
PAIR	iWRAP 3.0	WT12, WT11, WT32	Pairs with a Bluetooth device
PING	iWRAP 2.2.0	WT12, WT11, WT32	Pings a Bluetooth connection
PIO	iWRAP 3.0	WT12, WT11, WT32	Reads & Writes PIO statuses
RESET	iWRAP 2.1.0	WT12, WT11, WT32	Does a software reset
RFCOMM	iWRAP 3.0	WT12, WT11, WT32	Sets up RFCOMM channels
RSSI	iWRAP 2.2.0	WT12, WT11, WT32	Reads RSSI of a connection
SCO ENABLE	iWRAP 2.2.0	WT12, WT11, WT32	Enables SCO connections
SCO OPEN	iWRAP 2.2.0	WT12, WT11, WT32	Opens SCO connection
SDP	iWRAP 2.2.0	WT12, WT11, WT32	Browse SDP records
SDP ADD	iWRAP 2.2.0	WT12, WT11, WT32	Create SDP entries
SELECT	iWRAP 2.1.0	WT12, WT11, WT32	Selects a Bluetooth connection
SET	iWRAP 2.1.0	WT12, WT11, WT32	Lists iWRAP configuration
SLEEP	iWRAP 2.2.0	WT12, WT11, WT32	Enables deep sleep
TEMP	iWRAP 3.0	WT12, WT11, WT32	Reads internal temperature sensor
TEST	iWRAP 2.2.0	WT12, WT11, WT32	Enables self test modes
TESTMODE	iWRAP 2.2.0	WT12, WT11, WT32	Enables Bluetooth test mode
TXPOWER	iWRAP 2.2.0	WT12, WT11, WT32	Reads TX power level
VOLUME	iWRAP 3.0	WT32	Changes volume level

Table 3: Supported generic iWRAP commands

Command:	iWRAP version:	HW version:	Short description
SET BT BDADDR	iWRAP 2.1.0	WT12, WT11, WT32	Read BD_ADDR
SET BT NAME	iWRAP 2.1.0	WT12, WT11, WT32	Change friendly name
SET BT CLASS	iWRAP 2.1.0	WT12, WT11, WT32	Set Class-of-Device
SET BT AUTH	iWRAP 2.1.0	WT12, WT11, WT32	Set PIN code
SET BT IDENT	iWRAP 3.0	WT12, WT11, WT32	Set DI data
SET BT LAP	iWRAP 2.2.0	WT12, WT11, WT32	Set inquiry access code
SET BT OPP	iWRAP 2.2.0	WT12, WT11, WT32	Enable OPP profile
SET BT PAGEMODE	iWRAP 2.1.0	WT12, WT11, WT32	Set page mode and timeout
SET BT PAIR	iWRAP 2.1.0	WT12, WT11, WT32	Manage pairings
SET BT POWER	iWRAP 2.2.0	WT12, WT11, WT32	Set TX power levels
SET BT ROLE	iWRAP 2.1.0	WT12, WT11, WT32	Set role and supervision timeout
SET BT SNIFF	iWRAP 2.2.0	WT12, WT11, WT32	Manage automatic sniff mode

Table 4: Supported "SET BT" commands

Command:	iWRAP version:	HW version:	Short description
SET {link_id} MASTER	iWRAP 2.1.0	WT12, WT11, WT32	Set Bluetooth link to master
SET {link_id} SLAVE	iWRAP 2.1.0	WT12, WT11, WT32	Set Bluetooth link to slave
SET {link_id} MSC	iWRAP 2.2.0	WT12, WT11, WT32	Set Bluetooth link MSC status
SET {link_id} SELECT	iWRAP 3.0	WT12, WT11, WT32	Set Bluetooth link to active status
SET {link_id} ACTIVE	iWRAP 2.1.0	WT12, WT11, WT32	Disable Bluetooth link power saving
SET {link_id} SNIFF	iWRAP 2.1.0	WT12, WT11, WT32	Enable Sniff mode on a Bluetooth link
SET {link_id} PARK	only iWRAP 2.2.0	WT12, WT11, WT32	Enable Park state on a Bluetooth link

Table 5: Supported link control commands

Command:	iWRAP version:	HW version:	Short description
PROFILE SPP	iWRAP 2.1.0	WT12, WT11, WT32	Enable / disable SPP profile
PROFILE HFP	iWRAP 2.1.0	WT12, WT11, WT32	Enable / disable HFP profile
PROFILE HFP-AG	iWRAP 2.1.0	WT12, WT11, WT32	Enable / disable HFP profile (AG)
PROFILE HID	iWRAP 2.1.0	WT12, WT11, WT32	Enable / disable HID profile
PROFILE OPP	iWRAP 3.0	WT12, WT11, WT32	Enable / disable OPP profile
PROFILE A2DP	iWRAP 2.2.0	WT32	Enable / disable A2DP profile
PROFILE OTA	iWRAP 3.0.0	WT12, WT11, WT32	Enable / disable OTA profile

Table 6: Supported Bluetooth profile commands

Command:	iWRAP version:	HW version:	Short description
CONTROL AUTOCALL	iWRAP 2.1.0	WT12, WT11, WT32	Manage automatic call
CONTROL BAUD	iWRAP 2.1.0	WT12, WT11, WT32	Change UART baud rate
CONTROL BIND	iWRAP 2.2.0	WT12, WT11, WT32	Manage GPIO bindings
CONTROL CD	iWRAP 2.1.0	WT12, WT11, WT32	Manage Carrier Detect signal
CONTROL CONFIG	iWRAP 2.1.0	WT12, WT11, WT32	Manage configuration bits
CONTROL ECHO	iWRAP 2.1.0	WT12, WT11, WT32	Manage echo mode
CONTROL GAIN	iWRAP 3.0	WT32	Manage ADC and DAC gains
CONTROL INIT	iWRAP 2.1.0	WT12, WT11, WT32	Manage start-up command
CONTROL MICBIAS	iWRAP 3,0	WT32	Control MIC bias settings
CONTROL MSC	iWRAP 2.2.0	WT12, WT11, WT32	Manage MSC functionality
CONTROL MUX	iWRAP 2.2.0	WT12, WT11, WT32	Manage MUX mode
CONTROL PCM	iWRAP 3.0	WT12, WT11, WT32	Manage PCM settings
CONTROL VREGEN	iWRAP 3.0	WT32	Manage VREG_EN functionality

Table 7: Supported "SET CONTROL" commands

NOTE:

- The parser is not case sensitive!
- ASCII interface 0.0.2 does not accept backspaces, but version 2.0.0 and later do.
- iWRAP commands must end with a line feed **"\n"** character.

5.1 Typographical Conventions

The ASCII commands and their usage are described further in this chapter. Commands and their output synopsis are presented as follows:

Synopsis:
COMMAND {*required parameter*} [*optional parameter*] STATIC TEXT [*2ND OPTIONAL PARAMETER*]

Command parameters, on the other hand, are described like this:

Description:	
parameter	Description

Responses to the command are described as in the table below:

Response:	
RESPONSE {*parameters*}	
parameter	Description

Events generated by commands or actions are described as follows:

Events:	
EVENT	Description

The list format is described as follows (only presented with SET commands):

Events:
COMMAND {*required parameter*} [*optional parameter*]

Finally, examples shown are described like this:

```
EXAMPLE COMMAND

RESPONSE TO COMMAND

(comments)
```

5.2 AT

"Attention", can be used to check that iWRAP is functional and in command mode.

5.2.1 Syntax

Synopsis:
AT

Response:
OK

5.2.2 Examples

Sending AT to iWRAP:

```
AT
OK
```

Tip:

Most iWRAP commands do not produce replies telling that command was successful or execution has finished. **AT** command can be used to provide this functionality, but appending AT into the end of other iWRAP commands.

Appending AT after "SET BT AUTH" command:

```
SET BT AUTH * 1234\r\nAT\r\n
OK
```

5.3 INQUIRY

Command **INQUIRY** is used to find other *Bluetooth* devices in the area i.e. to make a device discovery.

5.3.1 Syntax

Synopsis:
INQUIRY {*timeout*} [NAME] [LAP {*lap*}]

Description:	
timeout	The maximum amount of time (in units of 1.28 seconds) before the inquiry process is halted. Range: 1-48
NAME	Optional flag to automatically request the friendly name for found devices. See command **NAME** for more information about the remote name request.
LAP	Optional flag for specifying that inquiry access code will be used.
lap	Value for inquiry access code. The following values are possible: **0x9e8b33** General/Unlimited Inquiry Access Code (GIAC). This is the default value unless "**SET BT LAP**" is used. **0x9e8b00** Limited Dedicated Inquiry Access Code (LIAC). **0x9e8b01-0x9e8b32 and 0x9e8b34-0x9e8b3f** Reserved for future use.

Response:	

INQUIRY {*num_of_devices*}
 	
and
 	
INQUIRY {*addr*} {*class_of_device*}	

num_of_devices	The number of found devices
addr	*Bluetooth* address of a found device
class_of_device	*Bluetooth* Class of Device of a found device

Events:	
INQUIRY_PARTIAL	These events are delivered as devices are found.
NAME	These events are delivered after **INQUIRY** if the ***NAME*** flag is present.
INQUIRY_EXTENDED	These events are delivered when Bluetooth 2.1 + EDR devices are found that support Extended Inquiry Response (EIR)

Notes:

- It can take up to 10.24 seconds for a *Bluetooth* device to answer an inquiry scan and, thus, the timeout value should be at least 8 if it is necessary to find every device in the area.
- iWRAP 2.1.0 and later support RSSI in the inquiry, but this feature must be enabled with "**SET CONTROL CONFIG**" command.
- Inquiry uses by default the LAP value defined with "**SET BT LAP**".
- **INQUIRY_PARTIAL** events can be masked off by using the **"SET CONTROL ECHO"** command.
- "SET CONTROL CONFIG" bit 0x8000 enables low inquiry priority. Low inquiry priority decreases the inquiry priority.
- "SET CONTROL CONFIG" upper bit 0x2 enables Extended Inquiry Response.

5.3.2 Examples

Basic INQUIRY command:

```
INQUIRY 1
INQUIRY_PARTIAL 00:14:a4:8b:76:9e 72010c
INQUIRY_PARTIAL 00:10:c6:62:bb:9b 1e010c
INQUIRY_PARTIAL 00:10:c6:4d:62:5c 72010c
INQUIRY_PARTIAL 00:10:c6:3a:d8:b7 72010c
INQUIRY_PARTIAL 00:02:ee:d1:80:6d 520204
INQUIRY_PARTIAL 00:10:c6:62:bb:fa 1c010c
INQUIRY 6
INQUIRY 00:14:a4:8b:76:9e 72010c
INQUIRY 00:10:c6:62:bb:9b 1e010c
INQUIRY 00:10:c6:4d:62:5c 72010c
INQUIRY 00:10:c6:3a:d8:b7 72010c
INQUIRY 00:02:ee:d1:80:6d 520204
INQUIRY 00:10:c6:62:bb:fa 1c010c
```

An INQUIRY command with NAME resolution:

```
INQUIRY 1 NAME
INQUIRY_PARTIAL 00:10:c6:3a:d8:b7 72010c
INQUIRY_PARTIAL 00:10:c6:62:bb:9b 1e010c
INQUIRY_PARTIAL 00:14:a4:8b:76:9e 72010c
INQUIRY 3
INQUIRY 00:10:c6:3a:d8:b7 72010c
INQUIRY 00:10:c6:62:bb:9b 1e010c
INQUIRY 00:14:a4:8b:76:9e 72010c
NAME 00:10:c6:3a:d8:b7 "TOM"
NAME 00:10:c6:62:bb:9b "CSLTJANI"
NAME 00:14:a4:8b:76:9e "SWLTMIKKO_3"
```

An INQUIRY command with LAP in use:

```
INQUIRY 3 LAP 9e8b11
INQUIRY_PARTIAL 00:07:80:80:52:15 111111
INQUIRY_PARTIAL 00:07:80:80:52:27 111111
INQUIRY 2
INQUIRY 00:07:80:80:52:15 111111
INQUIRY 00:07:80:80:52:27 111111
```

An INQUIRY command with RSSI enabled:

```
INQUIRY 1
INQUIRY_PARTIAL 00:10:c6:62:bb:9b 1e010c "" -71
INQUIRY_PARTIAL 00:10:c6:4d:62:5c 72010c "" -73
INQUIRY_PARTIAL 00:10:c6:3a:d8:b7 72010c "" -73
INQUIRY 5
INQUIRY 00:10:c6:62:bb:9b 1e010c
INQUIRY 00:10:c6:4d:62:5c 72010c
INQUIRY 00:10:c6:3a:d8:b7 72010c
```

An INQUIRY command with EIR responses:

```
INQUIRY 2
INQUIRY_PARTIAL 00:18:42:f1:a5:be 5a020c "" -92
INQUIRY_PARTIAL 00:17:e4:ef:f9:01 50020c "" -92
INQUIRY_EXTENDED 00:07:80:87:68:ec RAW 0909575433322d53616d020a0800
INQUIRY_PARTIAL 00:07:80:87:68:ec 200428 "WT32-Sam" -73
```

```
INQUIRY 3
INQUIRY 00:18:42:f1:a5:be 5a020c
INQUIRY 00:17:e4:ef:f9:01 50020c
INQUIRY 00:07:80:87:68:ec 200428
```

5.4 IC

The **IC** (inquiry cancel) command can be used to stop an on-going **INQUIRY**.

5.4.1 Syntax

Synopsis:
IC

Description:
No Description

Response:	

INQUIRY {*num_of_devices*}
 	
INQUIRY {*addr*} {*class_of_device*}	

num_of_devices	The number of found devices
addr	*Bluetooth* address of a found device
class_of_device	*Bluetooth* Class of Device of a found device

Events:
None

5.4.2 Examples

Canceling an INQUIRY command:

```
INQUIRY 5
INQUIRY_PARTIAL 00:14:a4:8b:76:9e 72010c
INQUIRY_PARTIAL 00:10:c6:62:bb:9b 1e010c
IC
INQUIRY 2
INQUIRY 00:14:a4:8b:76:9e 72010c
INQUIRY 00:10:c6:62:bb:9b 1e010c
```

Note:

- IC command cancels the inquiry only if issued before the "**INQUIRY {num_of_devices}**" message.

5.5 NAME

Command **NAME** is used to retrieve the friendly name of the device.

5.5.1 Syntax

Synopsis:
NAME {*address*}

Description:	
address	Address of the *Bluetooth* device

Response:
No response

Events:	
NAME	These events are delivered after **INQUIRY** if the ***NAME*** flag is present.
NAME_ERROR	These events are delivered if name resolution fails.

5.5.2 Examples

Successful name resolution:

```
NAME 00:07:80:bf:bf:01
NAME 00:07:80:bf:bf:01 “iWRAP_2.1.0”
```

Unsuccessful name resolution:

```
NAME 00:07:80:bf:bf:01
NAME ERROR 0x104 00:07:80:bf:bf:01 HCI_ERROR_PAGE_TIMEOUT
```

5.6 PAIR

PAIR command can be used to pair with other *Bluetooth* devices.

5.6.1 Syntax

Synopsis:
PAIR {*bd_addr*}

Description:	
bd_addr	*Bluetooth* device address of the remote device

Response:
No response

Events:	
PAIR {*bd_addr*} {status}	This event occurs if PAIR event is enabled with "**SET CONTROL CONFIG**" and pairing is successful.
SYNTAX ERROR	If incorrect parameters are given.
AUTH	This event occurs if interactive pairing is enabled with "**SET CONTROL CONFIG**".

5.6.2 Examples

Successful pairing with a remote device when pin code is enabled with SET BT AUTH:

```
PAIR 00:07:80:80:12:34
PAIR 00:07:80:80:12:34 OK
```

Unsuccessful pairing with a remote device when pin code is enabled with SET BT AUTH:

```
PAIR 00:07:80:80:12:34
PAIR 00:07:80:80:12:34 FAIL
```

Successful pairing with a remote device and interactive pairing:

```
PAIR 00:07:80:80:12:34
AUTH 00:07:80:80:12:34?
AUTH 00:07:80:80:12:34 1234
PAIR 00:07:80:80:12:34 OK
```

5.7 AUTH

AUTH command can be used to reply to AUTH event.

5.7.1 Syntax

Synopsis:
AUTH {*bd_addr*} [*pin_code]*

Description:	
bd_addr	*Bluetooth* device address of the remote device
pin_code	*Bluetooth* pin code

Response:
No response

Events:	
PAIR {*bd_addr*} {*link_key*}	This event occurs if PAIR event is enabled with SET CONTROL CONFIG and pairing is successful.

5.7.2 Examples

Pairing with AUTH command, initiated from remote device:

```
AUTH 00:07:80:80:12:34?
AUTH 00:07:80:80:12:34 1234 (Remote device asks for a PIN code)
```

Declining pairing with AUTH command

```
AUTH 00:07:80:80:12:34?
AUTH 00:07:80:80:12:34 (Pairing fails)
```

Pairing with AUTH command and with **PAIR** event enabled.

```
AUTH 00:07:80:80:12:34?
AUTH 00:07:80:80:12:34 1234
PAIR 00:07:80:80:12:34 4000e000540007d007d006100db006b003100
```

Note:

- Even if pin code is enabled with "**SET BT AUTH**" command you still can use a different pin code with "**AUTH**" command. However if no pin code is set with "**SET BT AUTH**", the remote end can choose the pin code and "**AUTH {bd_addr} [pin_code]**" command must use the same.

5.8 CALL

The **CALL** command is used to initiate *Bluetooth* connections to the remote devices. Connections are closed by using command **CLOSE**. Currently open connections can be viewed by using command **LIST**.

5.8.1 Syntax

Synopsis:
CALL {*address*} {*target*} {*connect_mode*} [MTU {*packet size*}]

Description:	
address	*Bluetooth* address of the remote device
target	RFCOMM, HFP or HFP-AG, HID or A2DP target for the connection. The target can be one of the following: **channel** RFCOMM channel number HFP channel number HFP-AG channel number Format: xx (hex) **uuid16** 16-bit UUID for searching channel Format: xxxx (hex) **uuid32** 32-bit UUID for searching channel Format: xxxxxxxx (hex) **uuid128** 128-bit UUID for searching channel Format: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx (hex) **L2CAP psm** 16-bit L2CAP psm Format: xxxx (hex)

connect_mode	Defines the connection mode to be established. Possible modes are: **RFCOMM** Normal RFCOMM connection **HFP** Opens a connection in the Hands Free device mode. **HFP-AG** Opens a connection in the Hands Free Audio Gateway mode. **A2DP** Opens a connection in the Advanced Audio Distribution Profile (A2DP) mode or Audio Video Remote Control Profile (AVRCP) mode. L2CAP psm for A2DP is 19 and for AVRCP 17. **HID** Opens a connection in the HID keyboard mode or HID mouse mode. L2CAP psm for HID is 11. **L2CAP** Opens a generic L2CAP connection.
MTU	Optional static text to indicate that the packet size parameter is used.
packet size	Packet size to use (Values from 21 to 1008 can be used).

Response:	
CALL {*link_id*}	
link_id	Numeric connection identifier

Events:	
CONNECT	Delivered if the **CALL** command is successful.
NO CARRIER	Delivered if the **CALL** command fails.

PAIR	If the **PAIR** event is enabled by using "**SET CONTROL CONFIG**", it will be displayed during the call if paring has to be done.
CLOCK	If piconet clock event is enabled, **CLOCK** event will be displayed.
AUTH	IF interactive pairing mode is enabled and no paring exists, **AUTH** event will be displayed.

5.8.2 Examples

Creating a successful connection to 00:07:80:80:52:27 using Serial Port Profile.

(UUID16 SPP = 1101)

```
CALL 00:07:80:80:52:27 1101 RFCOMM
CALL 0
CONNECT 0 RFCOMM 1
```

Creating a successful connection to 00:07:80:80:52:27 using RFCOMM channel 1.

```
CALL 00:07:80:80:52:27 1 RFCOMM
CALL 0
CONNECT 0 RFCOMM 1
```

Unsuccessful SPP connection attempt to 00:07:80:80:52:26.

```
CALL 00:07:80:80:52:26 1101 RFCOMM
CALL 0
NO CARRIER 0 ERROR 406 RFC_CONNECTION_FAILED
```

Creating a successful connection to 00:07:80:80:52:27 with MTU 600.

```
CALL 00:07:80:80:52:27 1101 RFCOMM MTU 600
CALL 0
CONNECT 0 RFCOMM 1
```

Creating a successful A2DP connection

```
CALL 00:1f:00:bc:73:99 19 A2DP
CALL 0
CONNECT 0 A2DP 25
CONNECT 1 A2DP 25
```

Creating a successful AVRCP connection

```
CALL 00:1f:00:bc:73:99 17 A2DP
CALL 0
CONNECT 0 A2DP 23
```

Creating a successful HID connection

```
CALL 00:1f:00:bc:73:99 11 HID
CONNECT 0 HID 17
CONNECT 0 HID 19
```

Note:

- If **CALL** is used with **CHANNEL** instead of **UUID**, it will be on average around 300ms faster, since there is no need to do service discovery. However when calling directly with RFCOMM channel you need to be sure that the profile you want to connect to is always in that RFCOMM channel. RFCOMM channel assignments are manufacturer specific and vary between different *Bluetooth* devices.

5.9 CLOSE

Command **CLOSE** is used to terminate a *Bluetooth* connection.

5.9.1 Syntax

Synopsis:
CLOSE {*link_id*}

Description:	
link_id	Numeric connection identifier from a previously used command **CALL** or from event **RING**.

Response:
No response

Events:	
NO CARRIER	This event is delivered after the link is closed.

5.9.2 Examples

Closing an active connection:

```
CALL 00:60:57:a6:56:49 1103 RFC
CALL 0
CONNECT 0 RFCOMM 1
[+++] (mode transition)
READY.
CLOSE 0
NO CARRIER 0 ERROR 0
```

5.10 LIST

Command **LIST** shows information about active connections.

5.10.1 Syntax

Synopsis:
LIST

Description:
No Description

Response:	

LIST {*num_of_connections*}
 	
LIST {*link_id*} CONNECTED {*mode*} {*blocksize*} 0 0 {*elapsed_time*} {*local_msc*} {*remote_msc*} {*addr*} {*channel*} {*direction*} {*powermode*} {*role*} {*crypt*} {buffer}	

link_id	Numeric connection identifier
mode	**RFCOMM** Connection type is RDCOMM **L2CAP** Connection type is L2CAP **SCO** Connection type is SCO
blocksize	RFCOMM, L2CAP or SCO data packet size, that is, how many bytes of data can be sent in one packet
elapse_time	Link life time in seconds
local_msc	Local serial port status (MSC) bits. "8d" is a normal value.
remote_msc	Remote serial port status (MSC) bits. "8d" is a normal value.

addr	*Bluetooth* device address of the remote device
channel	RFCOMM channel number at remote device
direction	Direction of the link. The possible values are: **OUTGOING** The link is initiated by a local device (by using command **CALL**) **INCOMING** The link is initiated by the remote device
powermode	Power mode for the link. The possible values are: **ACTIVE** Link is in active mode **SNIFF** Link is in sniff mode **HOLD** Link is in hold mode **PARK** Link is in park mode
role	Role of the link. The possible values are: **MASTER** iWRAP is the master device of this link **SLAVE** iWRAP is the slave device of this link
crypt	Encryption state of the link. The possible values are: **PLAIN** Link is not encrypted **ENCRYPTED** Link is encrypted
buffer	Tells the amount of data (in bytes) that is stored in the incoming data buffer.

Response:	

LIST {*num_of_connections*}
 	
LIST {*link_id*} CONNECTED RFCOMM {*blocksize*} 0 0 {*elapsed_time*} {*local_msc*} {*remote_msc*} {*addr*} {*channel*} {*direction*} {*powermode*} {*role*} {*crypt*} {buffer}	

link_id	Numeric connection identifier
blocksize	RFCOMM data packet size, that is, how many bytes of data can be sent in one packet
elapse_time	Link life time in seconds
local_msc	Local serial port status bits, "8d" is a normal value
remote_msc	Remote serial port status bits, "8d" is a normal value
addr	*Bluetooth* device address of the remote device
channel	RFCOMM channel number at remote device
direction	Direction of the link. The possible values are: **OUTGOING** The link is initiated by a local device (by using command **CALL**) **INCOMING** The link is initiated by the remote device
powermode	Power mode for the link. The possible values are: **ACTIVE** Link is in active mode **SNIFF** Link is in sniff mode **HOLD** Link is in hold mode **PARK** Link is in park mode

role	Role of the link. The possible values are: **MASTER** iWRAP is the master device of this link **SLAVE** iWRAP is the slave device of this link
crypt	Encryption state of the link. The possible values are: **PLAIN** Link is not encrypted **ENCRYPTED** Link is encrypted
buffer	Tells the amount of data (in bytes) that is stored in the incoming data buffer.

Events:
No events raised

5.10.2 Examples

Listing active connections:

```
LIST
LIST 1
LIST 0 CONNECTED RFCOMM 320 0 0 3 8d 8d 00:60:57:a6:56:49 1 OUTGOING ACTIVE
MASTER PLAIN 0
```

5.11 SELECT

Command **SELECT** is used to switch from command mode to data mode.

5.11.1 Syntax

Synopsis:
SELECT {*link_id*}

Description:	
link_id	Numeric connection identifier

Response:
No response if a valid link is selected. iWRAP goes to data mode of the link ***link_id***.

Events:	
SYNTAX ERROR	This event occurs if an invalid ***link_id*** is given

5.11.2 Examples

Changing between links:

```
LIST
LIST 2
LIST 0 CONNECTED RFCOMM 668 0 0 243 8d 8d 00:07:80:80:38:77 1 OUTGOING ACTIVE
MASTER ENCRYPTED
LIST 1 CONNECTED RFCOMM 668 0 0 419 8d 8d 00:07:80:80:36:85 1 OUTGOING ACTIVE
MASTER ENCRYPTED
SELECT 1      (iWRAP goes to DATA mode – Device: 00:07:80:80:36:85)
```

5.12 SDP

The **SDP** command can be used to browse the available services on other *Bluetooth* devices.

5.12.1 Syntax

Synopsis:
SDP {*bd_addr*} {*uuid*}

Description:	
bd_addr	*Bluetooth* address of the remote device
uuid	Service to look for UUID "1002" stands for root and returns all the services the remote device supports.

Response:	
SDP {*bd_addr*} < I SERVICENAME S "service_*name*" > ***< I PROTOCOLDESCRIPTORLIST < < U L2CAP > < U RFCOMM I channel > > >*** ***SDP***	
bd_addr	*Bluetooth* address of the remote device
service name	Name of the service. For example "Serial Port Profile"
channel	RFCOMM channel for the service

Events:
None

5.12.2 Examples

How to look for the SPP service:

```
SDP 00:07:80:80:52:15 1101
SDP 00:07:80:80:52:15 < I SERVICENAME S "Bluetooth Serial Port" > < I PROTOCOLDE
```

```
SCRIPTORLIST < < U L2CAP > < U RFCOMM I 01 > > >
SDP
```

128-bit SDP response:

```
SDP 00:17:4b:67:a8:c3 1101
SDP 00:17:4b:67:a8:c3 < I SERVICENAME S "Bluetooth SPP" > < I
PROTOCOLDESCRIPTORLIST < < U 00000100-0000-1000-8000-00805f9b34fb > < U
00000003-0000-1000-8000-00805f9b34fb I 19 > > >
```

According to the *Bluetooth* specification:

00000100-0000-1000-8000-00805f9b34fb	= L2CAP
00000003-0000-1000-8000-00805f9b34fb	= RFCOMM

5.13 SDP ADD

The **SDP ADD** command can be used to modify a local service record to add new services.

5.13.1 Syntax

Synopsis:
SDP ADD {*uuid*} {*name*}

Description:	
uuid	Identifier of the service
name	Name of the service

Response:	
SDP {*channel*}	
channel	RFCOMM channel where the service is bound to

Events:
None

5.13.2 Examples

Adding a Dial-Up Networking profile

```
SDP ADD 1103 Dial-Up Networking
SDP 2
```

Note:

- The service record will be cleared when a reset is made, so SDP ADD command(s) must be given every time after a reset, unlike SET commands, which are stored on flash memory.
- “SET CONTROL INIT” can be used to automatically issue one “SDP ADD” command.

5.14 IDENT

IDENT command can be used to identify a remote *Bluetooth* device with the *Bluetooth* Device ID method.

5.14.1 Syntax

Synopsis:
IDENT {*bd_addr*}

Description:	
bd_addr	*Bluetooth* device address of the remote device

Response:
No response

Events:	
IDENT	**IDENT** event is raised if a successful response is received
IDENT ERROR	**IDENT ERROR** event is raised if identification fails

5.14.2 Examples

Succesfully using IDENT to identify a remote Bluetooth device.

```
IDENT 00:07:80:00:a5:a5
IDENT 00:07:80:00:a5:a5 BT:47 f000 3.0.0 “Bluegiga iWRAP”
IDENT 00:07:80:82:42:d8
IDENT 00:07:80:82:42:d8 BT:47 b00b 3.2.0 “Bluegiga Access Server”
```

Using IDENT to try to identify a remote Bluetooth device without success.

```
IDENT 00:07:80:00:48:84
IDENT ERROR 2 00:07:80:00:48:84 NOT_SUPPORTED_BY_REMOTE
```

5.15 L2CAP

Command **L2CAP** is used to create a L2CAP psm for L2CAP connections to the device.

5.15.1 Syntax

Synopsis:
L2CAP {*uuid16*}

Description:	
uuid16	16-bit UUID for searching channels; must be odd.

Response:
No response

Events:
SYNTAX ERROR if an invalid UUID is given; no event if successful.

5.15.2 Examples

Making an L2CAP call between two iWRAPs:

```
L2CAP 25 (device 1, address 00:07:80:12:34:56)

CALL 00:07:80:12:34:56 25 L2CAP (device 2)
CALL 0
CONNECT 0 L2CAP 37 (note: UUID is given in hexadecimal format for L2CAP and CALL, but is
displayed in decimal format in CONNECT events.)
(devices go to data mode)
+++
READY.
LIST
LIST 1
LIST 0 CONNECTED L2CAP 672 0 0 193 0 0 00:70:80:12:34:56 37 OUTGOING ACTIVE
MASTER PLAIN 0
```

5.16 RFCOMM

Command **RFCOMM** is used to create a RFCOMM channel for general RFCOMM connections.

5.16.1 Syntax

Synopsis:
RFCOMM {*action*}

Description:	
action	**CREATE** Creates a generic RFCOMM channel.

Response:	
RFCOMM {*channel*}	
channel	RFCOMM channel number

Events:
None

5.16.2 Examples

Creating a generic RFCOMM channel.

```
RFCOMM CREATE
RFCOMM 2
```

5.17 SCO ENABLE

The **SCO ENABLE** command is needed before any SCO (audio) connections can be used

5.17.1 Syntax

Synopsis:
SCO ENABLE

Description:
None

Response:
None

Events:
None

Note:

- The SCO ENABLE command must be given every time after reset; it is not stored on flash memory.
- “SET CONTROL INIT” can be used to automatically issue one “SCO ENABLE” command.
- IF HFP or HFP-AG mode is enabled SCO ENBLED command is not needed.

5.18 SCO OPEN

The **SCO OPEN** command is used to open the actual SCO connection. An existing RFCOMM connection is needed before SCO OPEN can be issued.

5.18.1 Syntax

Synopsis:
SCO OPEN {*link_id*}

Description:	
link_id	Numeric connection identifier

Response:
None

Response:
None

Events:	
CONNECT	If SCO connection was opened successfully
NO_CARRIER	If connection opening failed

Note:

- The SCO ENABLE command must be given before the SCO OPEN command can be used.

5.18.2 Examples

Creating an SCO connection to another iWRAP device:

```
SCO ENABLE
CALL 00:07:80:80:52:27 1 RFCOMM
CALL 0
CONNECT 0 RFCOMM 1
[+++]
SCO OPEN 0
CONNECT 1 SCO
```

5.19 CLOCK

CLOCK command can be used to read the *Bluetooth* piconet clock value.

5.19.1 Syntax

Synopsis:
CLOCK {*link_id*}

Description:	
link_id	Numeric connection identifier

Response:
No response

Events:	
CLOCK {bd_addr} {clock}	CLOCK event occurs, if valid ***link_id*** is used.
SYNTAX ERROR	If incorrect parameters are given.

5.19.2 Examples

Reading Piconet clock value:

```
CLOCK 0
CLOCK 00:07:80:12:34:56 3bb630
```

Note:

- Piconet clock is extremely useful when time needs to be synchronized between Piconet slaves. All the slaves in the Piconet are synchronized to master’s clock and they share the same clock value.
- Accuracy is 625us.

5.20 KILL

Command **KILL** is used to explicitly disconnect a connection.

5.20.1 Syntax

Synopsis:
KILL {*bd_addr*} [*reason*]

Description:	
bd_addr	*Bluetooth* address of the connected remote device.
reason	Reason for disconnecting; see Chapter 9 for a listing of possible error codes. The default value is 0x115: HCI_ERROR_OETC_POWERING_OFF, device is about to power off.

Response:
None

Events:	
NO CARRIER	This event is delivered after the link is closed.

5.21 BER

The **BER** command returns the Bit Error Rate of the given link ID.

5.21.1 Syntax

Synopsis:
BER {*link_id*}

Description:	
link_id	Numeric connection identifier

Response:	
BER {*bd_addr*} {*ber*}	
bd_addr	*Bluetooth* address of the remote device
ber	Average Bit Error Rate on the link. Possible values are from 0.0000 to 100.0000.

Events:
None

5.21.2 Examples

Checking the Bit Error Rate of an active connection

```
LIST
LIST 1
LIST 0 CONNECTED RFCOMM 320 0 0 3 8d 8d 00:60:57:a6:56:49 1 OUTGOING ACTIVE
MASTER PLAIN
BER 0
BER 00:60:57:a6:56:49 0.0103
```

Note:

- Works only for BDR links.

5.22 RSSI

The **RSSI** command returns the Receiver Signal Strength Indication of the link given as a parameter.

5.22.1 Syntax

Synopsis:
RSSI {*link_id*}

Description:	
link_id	Numeric connection identifier

Response:	
RSSI {*bd_addr*} {*rssi*}	
bd_addr	*Bluetooth* address of the remote device
rssi	Receiver Signal Strength Indication. Possible values are from +20 to -128. 20 = Good link -128 = Poor link

Events:
None

5.22.2 Examples

Checking the Bit Error Rate of an active connection:

```
LIST
LIST 1
LIST 0 CONNECTED RFCOMM 320 0 0 3 8d 8d 00:60:57:a6:56:49 1 OUTGOING ACTIVE
MASTER PLAIN
RSSI 0
RSSI 00:60:57:a6:56:49 -10
```

5.23 TXPOWER

The **TXPOWER** command can be used check the TX output power level of an active *Bluetooth* link.

5.23.1 Syntax

Synopsis:
TXPOWER {*link_id*}

Description:	
link_id	Numeric connection identifier

Response:	
TXPOWER {*bd_addr*} {*txpower*}	
bd_addr	*Bluetooth* address of the remote device
txpower	User TX power level in dBm

Events:
None

5.23.2 Examples

Checking the TX power level of an active connection:

```
LIST
LIST 1
LIST 0 CONNECTED RFCOMM 320 0 0 3 8d 8d 00:60:57:a6:56:49 1 OUTGOING ACTIVE
MASTER PLAIN
TXPOWER 0
TXPOWER 00:60:57:a6:56:49 3
```

5.24 PING

The **PING** command sends a *Bluetooth* test packet to the other device, which sends the packet back and the round trip time of the packet is shown.

5.24.1 Syntax

Synopsis:
PING {*link_id*}

Description:	
link_id	Numeric connection identifier

Response:	
RSSI {*bd_addr*} {*round trip time*}	
bd_addr	*Bluetooth* address of the remote device
round trip time	Round trip time of the packet

Events:
None

5.24.2 Examples

Checking the round trip time:

```
PING 0
PING 00:07:80:80:c3:4a 42

Round trip time is 42ms in this case.
```

5.25 ECHO

The **ECHO** command sends a specified string of characters to the active link specified by the 'link_id' parameter. This command can be used, for example, with command **SET CONTROL BIND** to send an indication of activity over a *Bluetooth* link.

5.25.1 Syntax

Synopsis:
ECHO {*link_id*} [*string*]

Description:	
link_id	Numeric connection identifier
string	User-determined string of characters

Response:
No response

Events:
None

5.25.2 Examples

```
ECHO 0 WT12_DATA

On the other device UART receive:
WT12_DATA
```

5.26 SLEEP

The **SLEEP** command will force deep sleep on. After issuing this command, the module will enter deep sleep until a *Bluetooth* connection is received or something is received from the UART interface in command mode. The SLEEP command will also work when there are one or more active connections and iWRAP is in command mode.

Deep sleep is an aggressive power saving mode for WRAP THOR modules.

5.26.1 Syntax

Synopsis:
SLEEP

Description:
None.

Response:
None

Events:
None

Note:

- Refer to power consumption documents for more information about power consumption in deep sleep mode.
- Deep sleep might sometimes be used even if there are active *Bluetooth* connections. However all the connections need to ne in aggressive sniff power saving mode.

5.27 PIO

Command **PIO** is used to get and set PIO states and directions. There are 6 usable IO pins (PIO2-PIO7) on the WT11/12 module, 11 (PIO0-PIO10) on the WT32; thus the range for the mask and state parameters for the WT11/12 is 4-FF (excluding values that have bits 0 or 1 set), for the WT32 it is 0-07FF.

5.27.1 Syntax

Synopsis:
PIO {*cmd*} [*mask*] [*states*]

Description:	
cmd	**GET** Read the contents of the PIO data register. Bits that are set denote pins that are pulled up. **GETDIR** Read the contents of the PIO direction register. Bits that are set denote output pins; others are input pins, i.e. controlled externally, such as the PIO buttons on the WT32 evaluation board. **GETBIAS** Read the contents of the PIO bias register. Bits that are set denote pins that are pulled up/down strongly, others are pulled up/down weakly. **SET {*mask*} {*states*}** Set the contents of the PIO data register; the first parameter is the bit mask for deciding which PIOs are affected, the second parameter is the bits to set/unset. **SETDIR {*mask*} {*states*}** Set the contents of the PIO direction register. By default, only bit 8 (PIO7) is set, thus only it can be controlled locally with PIO SET, and all others are input pins. **SETBIAS {*mask*} {*states*}** Set the contents of the PIO bias register. By default, all pins are pulled up/down weakly. **RESET** Set the registers to iWRAP defaults.

mask	Bit mask that defines the GPIO lines
states	Bit mask that defines the GPIO pin states

Response:	
No response if command execution is successful.	
SYNTAX ERROR	This event is raised if incorrect parameters are given

Events:	
PIO {*reg value*}	This event is raised

5.27.2 Examples

Setting PIO7 (which lights the PIO7 LED on the evaluation kit if the corresponding switch is in the LED position) and PIO5 (which is an input pin and thus cannot be set locally by default):

PIO SET 80 FF (80 in hex is 10000000 in binary, so only the 8th bit is affected; parameters 80 80 etc. would yield the same result)
PIO GET
PIO GET 180 (100 + bits; bit 9, e.g. 100 is always set)
PIO SET 20 FF (try to set 6th bit)
PIO GET
PIO GET 180 (only the 8th direction bit is set by default, so only PIO7 can be set)
PIO GETDIR
PIO GETDIR 80
PIO SETDIR 20 FF
PIO GETDIR
PIO GETDIR a0 (A0 is 10100000 in binary; now both PIO7 and PIO5 can be set)
PIO SET 20 FF
PIO GET
PIO GET 1a0 (now PIO5 is set too)
PIO RESET
PIO GET
PIO GET 100
PIO GETDIR
PIO GETDIR 80

5.28 VOLUME

Command **VOLUME** is used to modify and read the module's line out volume level.

5.28.1 Syntax

Synopsis:
VOLUME [{*vol*}]

Description:	
vol	New volume level value; leave blank to read current volume level. **0…9** Sets volume level: Range 0-9. **down** Decreases volume level by one. **up** Increases volume level by one.

Response:
None

Events:	
VOLUME {*vol*}	Current volume level.

5.29 TEMP

This command reads the value of internal temperature sensor. This value should not be considered very reliable. The value can be compensated by modifying PS-key PSKEY_TEMPERATURE_CALIBRATION.

5.29.1 Syntax

Synopsis:
TEMP

Description:
None.

Response:	
TEMP {*temp*}	
temp	Temperature in Celsius

Events:
None.

5.29.2 Examples

Reading the value of internal temperature sensor.

```
TEMP
TEMP 31
```

5.30 BATTERY

Command **BATTERY** is used to read the current voltage of the module battery.

5.30.1 Syntax

Synopsis:
BATTERY

Description:
None

Response:
None

Events:	
BATTERY {*mv*}	Current battery voltage in millivolts.

5.30.2 Examples

Reading battery voltage.

```
BATTERY
BATTERY 3673
```

5.31 BYPASSUART

BYPASSUART command enabled the UART bypass mode, in which the UART traffic is passed to GPIO pins instead of iWRAP. Please refer to the modules data sheet for more information. A physical reset is needed to return to normal operation mode.

5.31.1 Syntax

Synopsis:
BYPASSUART

Response:
No response

Events:
No event is raised

5.32 DEFRAG

This command defragments persistent store memory. The command resets iWRAP.

5.32.1 Syntax

Synopsis:
DEFRAG

Description:
None

Response:
No response

Events:
None

5.33 BCSP_ENABLE

Command **BCSP_ENABLE** is used to boot the device and enter BCSP mode; it is an alias for **BOOT 1.** See chapter 9.3 for a detailed explanation of iWRAP boot modes.

5.33.1 Syntax

Synopsis:
BCSP_ENABLE

Description:
None

Response:
No response

Events:
None

5.34 RESET

Command **RESET** is used to reset iWRAP.

5.34.1 Syntax

Synopsis:
RESET

Description:
No description

Response:
No response

5.35 BOOT

The **BOOT** command is used to change the iWRAP's boot mode. After issuing this command, the module will enter the selected boot mode. After resetting the module, it will boot in iWRAP mode again. The boot modes are explained in chapter.

5.35.1 Syntax

Synopsis:
BOOT [*boot_mode*]

Description:	
boot_mode	**0000** iWRAP **0001** HCI, BCSP, 115800,8n1 **0003** HCI, USB **0004** HCI, H4, 115200,8n1

Response:
No response

5.35.2 Examples

```
BOOT 1
      □Ò□¯WWUo`À
                □Ò□¯WWUo`À
                          □Ò□¯WWUo`À
                                    □Ò□¯WWUo`À
                                              □Ò□¯WWUo
`À
```

Example of changing the module to HCI BCSP 115200 with the BOOT command. After resetting the module, iWRAP becomes active.

5.36 TEST

The **TEST** command is used to give radio test commands to iWRAP. The commands are the same that can be given by using CSR BlueTest software. TEST commands must only be used for testing purposes, not for application functionality.

5.36.1 Syntax

Synopsis:
TEST {*mode*} [*mode_specific_parameters*]

Description:	
mode & ***mode_specific_parameters***	RF Test mode Supported test modes are: **PAUSE** Pause halts the current test and stops any radio activity. **TXSTART {lo_freq} {level} {mod_freq}** Enables the transmitter in continuous transmission at a designated frequency (**lo_freq**) with a designated output power (**level**) and designated tone modulation frequency (**mod_freq**). **lo_freq** range: 2402 – 2480 (MHz) **level** range: 0xff00 – 0xff3f **mod_freq range**: 0 – 32767 (recommended values 0 or 256) **TXDATA1 {lo_freq} {level}** Enables the transmitter with a designated frequency (**lo_freq**) and output power (**level**). Payload is PRBS9 data. In this mode, the receiver is not operating. **TXDATA2 {cc} {level}** Enables the transmitter with a simplified hop sequence designated by country code **{cc}** and output power **{level}**. Payload is PRBS9 data. In this mode, the receiver is not operating. Related test spec name: **TRM/CA/01/C** (output power), **TRM/CA/02/C** (power density).

	cc range: 0 – 3 (default = 0) **RXSTART {lo_freq} {highside} {attn}** Enables the receiver in continuous reception at a designated frequency (**lo_freq**) with a choice of low or high side modulation (**highside**) and with designated attenuation setting (**attn**). **highside** range: 0 or 1 (default = false = 0) **attn**: range: 0 – 15 **DEEPSLEEP** Puts the module into deep-sleep after a delay of half a second until woken by a reset or activity on UART. **PCMLB {pcm_mode}** Sets the PCM to loop back mode, where the data read from PCM input is output again on the PCM output. If **pcm_mode** = 0, module is slave in normal 4-wire configuration If **pcm_mode** = 1, module is master in normal 4-wire configuration If **pcm_mode** = 2, module is master in Manchester encoded 2-wire configuration **PCMEXTLB {pcm_mode}** Sets the PCM to external loop back mode, whereby the data written to PCM output is read again on the input. Check is made that the data read back is the same as that written. The external loop back may be a simple wire. Modes are save as above. **LOOPBACK {lo_freq} {level}** Receives data on set frequency **lo_freq** for data packets and then retransmits this data on the same channel at output power **level**. **CFGXTALFTRIM {xtal_ftrim}** This command can be used to set the crystal frequency trim value directly from iWRAP. This is not a permanent setting! **xtal_ftrim** range: 0 – 63 **PCMTONE {freq} {ampl} {dc}**

	Plays a constant tone on the PCM port. **freq** range: 0 – 5 **ampl** range : 0-8 **dc**: 0 – 60096 (set to 0) **SETPIO {mask} {bits}** Sets PIO high or low according to given parameters. NOTE: This command sets the PIO regardless of other usage! **mask**: Bit mask specifying the PIOs that are to be set **bits**: the bit values If you use hexadecimals, put 0x in front of the value, otherwise they are interpreted as decimals. **GETPIO** Gets the status of all the PIO lines.

Response:
OK for successful execution **ERROR** for unsuccessful execution

5.36.2 Examples

TEST TXSTART 2441 0xFF3F 0
OK

Example on how to set the module to transmit continuous carrier signal at 2441MHz and at full output power.

TEST PCMTONE 1 5 0
OK

Example on how to set the modules PCM output a constant signal for PCM testing.

Note:

- Always consult Bluegiga Technologies about the right parameters for RF testing. The parameters are unique for each module: WT11, WT12 and WT32.

5.37 TESTMODE

The **TESTMODE** command is used to put the iWRAP into a *Bluetooth* test mode, where a *Bluetooth* tester can control the hardware. Reset must be done to recover normal operation.

5.37.1 Syntax

Synopsis:
TESTMODE

Description:
No description.

Response:
TEST 0

Events:
None

5.38 HELP

Prints supported iWRAP commands.

5.38.1 Syntax

Synopsis:
HELP

Description:
No description

Response:
A list of supported iWRAP commands

Events:
None

5.38.2 Examples

```
HELP
HELP AUTH
HELP BATTery
HELP BER
HELP Call
HELP CLose
HELP CLOCK
HELP HELP
HELP INFO
HELP Inquiry
HELP IC
...
```

5.39 INFO

INFO displays information about iWRAP version and features.

5.39.1 Syntax

Synopsis:
INFO [*CONFIG* | *BOOTMODE*]

Description:	
CONFIG	Optional flag that that displays more detailed information about the firmware for example changed parameters.
BOOTMODE	Displays bootmode parameters

Response:
Information about iWRAP version and features.

Events:
None.

5.39.2 Examples

```
INFO
WRAP THOR AI (2.1.0 build 20)
Copyright (c) 2003-2006 Bluegiga Technologies Inc.
Compiled on Mar  1 2006 13:39:55, running on WT12 module, psr v5
        - BOCK3 version 15 (Mar  1 2006 13:38:28) (max acl/sco 7/1)
        - Bluetooth version 2.0, Power class 2
        - Firmware version 2626
        - up 0 days, 22:34, 0 connections (pool 1)
READY.
```

Detailed information display:

```
INFO CONFIG
WRAP THOR AI (2.3.0 build 77)
Copyright (c) 2003-2007 Bluegiga Technologies Inc.
Compiled on Oct 15 2007 18:19:00, running on WRAP THOR module, psr v0
        - BOCK3 version 27 (Mar 14 2007 15:19:59) (max acl/sco 7/1)
        - Bluetooth version 2.0, Power class 2
        - Loader 4156, firmware 4532 (56-bit encryption)
        - up 0 days, 00:39, 0 connections (pool 1)
        - User configuration:
&028b = 0000 0bb8
&028c = 0000 0020 0001 0008
&02a3 = 0031 0032 0033 0034
&02a4 = 1d80
&02a6 = 0007
&02a7 = 0020 0408
&02a8 = 8000 0000
&02aa = 0004 2000 0001 0033
&02ab = 0000 000f 3000
&02ac = 2181 0100 0000 0000 0000
&02ad = 5457 3233 662d 706f 0061
&02b6 = 0000
&02bb = 0000
READY.
```

Note:

- When requesting a custom firmware configuration from Bluegiga, it useful to attach output of "INFO CONFIG" to the request.

5.40 CONNECT

iWRAP3 can act as a repeater / range extender for RFCOMM connections by using the **CONNECT** command which will transparently link two ongoing connections together as a connection between the two remote devices.

5.40.1 Syntax

Synopsis:
CONNECT {*link_id_1*} {*link_id_2*}

Description:	
link_id_1	Numeric connection identifier as displayed by the LIST command.
link_id_2	Numeric connection identifier as displayed by the LIST command..

Response:
None

Events:
None

5.40.2 Examples

Changing PAGEMODE to 3 (to be able to accept the second call), receiving two calls, escaping to command mode with +++, checking for active connections with the LIST command and finally connecting the two remote devices through the local iWRAP:

```
SET BT PAGEMODE 3
RING 0 00:07:80:87:69:2f 1 RFCOMM
RING 1 00:07:80:87:68:ec 1 RFCOMM
+++
READY.
LIST
LIST 2
LIST 0 CONNECTED RFCOMM 320 0 0 33 8d 1 00:07:80:87:69:2f 1 INCOMING ACTIVE
SLAVE PLAIN 0
LIST 1 CONNECTED RFCOMM 320 0 0 31 8d 8d 00:07:80:87:68:ec 1 INCOMING ACTIVE
MASTER PLAIN 0
CONNECT 0 1
```

6. SET

With the **SET** command, you can display or configure different iWRAP configuration values.

6.1.1 Syntax of SET Commands

Synopsis:
SET [{*category*} {*option*} {*value*}]

Description:	
Without any parameters, **SET** displays the current configuration.	
category	Category of setting **BT** Changes different *Bluetooth* related settings. See **SET BT** for more information about options. **CONTROL** Changes different iWRAP settings. See **SET CONTROL** for more information about options. **PROFILE** Activates or deactivates Bluetooth profiles. **link_id** This command is used to control the various settings related to *Bluetooth* links in iWRAP. These are, for example, master, slave and power save modes (SNIFF and ACTIVE).
option	Option name, which depends on the category. See the following sections for more information.
value	Value for the option. See the following sections for more information.

Response:	
None if issued with parameters	

SET {*category*} {*option*} {*value*}
 	
SET
	If no parameters given displays current iWRAP settings.

Events:
None

6.1.2 Examples

Listing current settings:

```
SET
SET BT BDADDR 00:07:80:80:c2:37
SET BT NAME WT12
SET BT CLASS 50020c
SET BT AUTH * 9078
SET BT LAP 9e8b33
SET BT PAGEMODE 4 2000 1
SET BT PAIR 00:07:cf:51:f6:8d 9c4e70d929a83812a00badba7379d7c2
SET BT PAIR 00:14:a4:8b:76:9e 90357318b33817002c5c13b62ac6507f
SET BT PAIR 00:60:57:a6:56:49 3b41ca4f42401ca64ab3ca3303d8ccdc
SET BT ROLE 0 f 7d00
SET BT SNIFF 0 20 1 8
SET CONTROL BAUD 115200,8n1
SET CONTROL CD 80 0
SET CONTROL ECHO 7
SET CONTROL ESCAPE 43 00 1
SET
```

6.2 SET BT BDADDR

SET BT BDADDR shows the local device's *Bluetooth* address.

6.2.1 Syntax

Synopsis:
No description, since the value is read only.

Description:
No description

Response:
None

Events:
None

List format:	
SET BT BDADDR {*bd_addr*}	
bd_addr	*Bluetooth* device address of the local device

Note:

This value is read-only!

6.3 SET BT NAME

SET BT NAME shows or sets the local device's friendly name.

6.3.1 Syntax

Synopsis:
SET BT NAME {*friendly_name*}

Description:	
friendly_name	Friendly name of the local device

Response:
None

Events:	
SYNTAX ERROR	This event occurs if incorrect parameters are given

List format:
SET BT NAME {*friendly_name*}

Note:

- The maximum length of a friendly name is 16 characters in iWRAP 2.0.2 and older. In iWRAP 2.1.0 and newer versions, the maximum length is 256 characters.
- If ***friendly_name*** is left empty, some devices (like PCs or PDAs) may have problems showing the device in the inquiry.

6.4 SET BT CLASS

SET BT CLASS shows or sets the local device’s Class-of-Device (CoD).

Class of device is a parameter, which is received during the device discovery procedure, indicating the type of device and which services are supported.

6.4.1 Syntax

Synopsis:
SET BT CLASS {*class_of_device*}

Description:	
class_of_device	CoD of the local device

Response:
None

Events:	
SYNTAX ERROR	This event occurs if incorrect parameters are given

List format:
SET BT CLASS {*class_of_device*}

Note:

- The class-of-device parameter should reflect the features and supported profiles of a *Bluetooth* device. Refer to the *Bluetooth* specification for more information.
- A useful tool to work out Class of Device can be found from: http://bluetooth-pentest.narod.ru/software/bluetooth_class_of_device-service_generator.html

6.5 SET BT AUTH

SET BT AUTH shows or sets the local device's PIN code.

6.5.1 Syntax

Synopsis:
SET BT AUTH {*mode*} {*pin_code*}

Description:	
mode	Pin code usage mode: ***** Pin code will be displayed by "**SET**" command. **-** Pin code will NOT be displayed by "**SET**" command.
pin_code	PIN code for authorized connections. Authorization is required if this option is present. The PIN code can be from 0 to 16 characters.

Response:	
SYNTAX ERROR	This event occurs if incorrect parameters are given

Events:
None

List format:	
	If PIN code is not, **SET BT AUTH** * is not displayed
SET BT AUTH * {*pin_code*}	If PIN code is set
SET BT AUTH *	If pin code set with "**SET BT AUTH –**"

Note:

If command "***SET BT AUTH ****" is given, PIN code will be disabled and no encryption can be used during *Bluetooth* connections.

6.6 SET BT LAP

This command configures the Inquiry Access code (IAC) that iWRAP uses. IAC is used in inquiries and inquiry responses.

6.6.1 Syntax

Synopsis:
SET BT LAP {*iac*}

Description:	
iac	Value for the inquiry access code. The following values are possible: **9e8b33** General/Unlimited Inquiry Access Code (GIAC). This is the default value. **9e8b00** Limited Dedicated Inquiry Access Code (LIAC). **9e8b01 - 9e8b32** and **9e8b34-9e8b3f** Reserved for future use.

Response:	
SYNTAX ERROR	This event occurs if incorrect parameters are given

Events:
None.

List format:
SET BT LAP {*iac*}

Note:

- IAC is very useful in cases where the module needs to be visible in the inquiry but only for dedicated devices, such as other iWRAP modules, but not for standard devices like PCs or mobile phones. When the value of IAC is left to default value (**0x9e8b33**) iWRAP will be visible for all devices capable of making an inquiry. On the other, hand when IAC is set to 0x9e8b00 (LIAC), only devices capable of making limited inquiry will be able to discover iWRAP. Using LIAC will usually speed up the inquiry process since standard *Bluetooth* device like mobile phones and PC will normally not respond to inquiry.

6.7 SET BT PAGEMODE

SET BT PAGEMODE configures or displays the local device's page mode.

Page mode controls whether iWRAP can be seen in the inquiry and whether it can be connected. This command can also be used to change the page timeout.

6.7.1 Syntax

Synopsis:
SET BT PAGEMODE {*page_mode*} {*page_timeout*} {*page_scan_mode*}

Description:	
page_mode	This parameter defines the *Bluetooth* page mode. **0** iWRAP is NOT visible in the inquiry and does NOT answers calls **1** iWRAP is visible in the inquiry but does NOT answers calls **2** iWRAP is NOT visible in the inquiry but answers calls **3** iWRAP is visible in the inquiry and answers calls **4** Just like mode 3 if there are NO connections. If there are connections, it is like mode 0. (default value)
page_timeout	**0001 – FFFF** Page timeout defines how long the connection establishment can take before an error occurs. Page timeout is denoted as a hexadecimal number (HEX) and calculated as in the example below: 2000 (HEX) equals 8192 (DEC). Multiply it by 0.625 and you get the page timeout in milliseconds. In this case, it is 5120 ms (8192 * 0,625ms).
page_scan_mode	This parameter configures the *Bluetooth* page scan mode. The possible values are:

	0 Mode R0 means that iWRAP IS connectable all the time. High current consumption! **1** Mode R1 means that iWRAP is connectable every 1.28 sec (the default value) **2** Mode R2 means that iWRAP is connectable every 2.56 sec (lowest power consumption)

Response:	
SYNTAX ERROR	This event occurs if incorrect parameters are given

Events:
None

List format:
SET BT PAGEMODE {*page_mode*} {*page_timeout*} {*page_scan_mode*}

Note:

Command "**SET BT PAGEMODE**" returns default values.

6.8 SET BT PAIR

SET BT PAIR displays or configures the local device's pairing information.

6.8.1 Syntax

Synopsis:
SET BT PAIR {*bd_addr*} {*link_key*}

Description:	
bd_addr	*Bluetooth* address of the paired device
link_key	Link key shared between the local and the paired device. If this value is empty, pairing for the given *Bluetooth* address will be removed. Link key is 32hex values long.

Response:	
SYNTAX ERROR	This event occurs if incorrect parameters are given

Events:
None

List format:	
	SET BT PAIR is not displayed if there are no pairings
SET BT PAIR {*bd_addr*} {*link_key*}	One line per pairing is displayed

Note:

- iWRAP supports up to 16 simultaneous pairings. If 16 devices have been already paired, new pairings will not be stored.
- If command "**SET BT PAIR ***" is given, all pairings will be removed.

6.9 SET BT ROLE

This command configures or displays the local device's role configuration. With the "**SET BT ROLE**" command, iWRAP's master-slave behavior can be configured. This command can also be used to set the supervision timeout and link policy.

6.9.1 Syntax

Synopsis:
SET BT ROLE {*ms_policy*} {*link_policy*} {*supervision_timeout*}

Description:	
ms_policy	This parameter defines how the master-slave policy works. **0** This value allows master-slave switch when calling, but iWRAP does not request it when answering (default value). **1** This value allows master-slave switch when calling, and iWRAP requests it when answering. **2** If this value is set, master-slave switch is not allowed when calling, but it is requested for when answering.
link_policy	This bitmask controls the link policy modes. It is represented in a hexadecimal format. **Bit 1** If this bit is set, Role switch is enabled **Bit 2** If this bit is set, Hold mode is enabled **Bit 3** If this bit is set, Sniff mode is enabled **Bit 4** If this bit is set, Park state is enabled **F** This value enables all of the above modes (the default value)

	0 This value disables all of the above modes
supervision_timeout	**0001 – FFFF** Supervision timeout controls how long a *Bluetooth* link is kept open if the remote end does not answer. Supervision timeout is denoted as a hexadecimal number (HEX) and is calculated as in the example below: 12C0 (HEX) is 4800 (DEC). Multiply it by 0.625 and you get the supervision timeout in milliseconds. In this case, it is 3000 ms (4800 * 0,625ms). In other words, the remote end can be silent for three seconds until the connection is closed.

Response:
None

Events:	
SYNTAX ERROR	This event occurs if incorrect parameters are given

List format:
SET BT ROLE {*ms_policy*} {*link_policy*} {*supervision_timeout*}

Note:

Command **"SET BT ROLE"** returns default values.

6.10 SET BT SNIFF

This command enables automatic sniff mode for *Bluetooth* connections. Notice that remote devices may not support sniff.

6.10.1 Syntax

Synopsis:

SET BT SNIFF {*max*}{*min*} [{*attempt*} {*timeout*}]
 	
or
 	
SET BT SNIFF {*avg*}

Description:	
max	Maximum acceptable interval in milliseconds Range: **0002 b FFFE**; only even values are valid Mandatory Range: **0006** to **0540** Time = N * 0.625 msec Time Range: 1.25 msec to 40.9 sec
min	Minimum acceptable interval in milliseconds Range: **0002** to **FFFE**; only even values are valid Mandatory Range: **0006** to **0540** Time = N * 0.625 msec Time Range: 1.25 msec to 40.9 sec
avg	Average value in milliseconds. You can use this as a shortcut for easier sniff setting.
attempt	Number of base band receive slots for sniff attempt. Length = N* 1.25 msec Range for N: **0001 – 7FFF** Time Range: 0.625msec - 40.9 Seconds Mandatory Range for Controller: 1 to $T_{sniff}/2$

timeout	Number of Baseband receive slots for sniff timeout. Length = N * 1.25 msec Range for N: 0x0000 – 0x7FFF Time Range: 0 msec - 40.9 Seconds Mandatory Range for Controller: 0 to 0x0028

Response:
None

Events:	
SYNTAX ERROR	This event occurs if incorrect parameters are given

List format:
SET BT SNIFF {*max*}{*min*} {*attempt*} {*timeout*}

Note:

- "**SET BT SNIFF**" disables automatic sniff mode (default settings).
- You can not change sniff mode on the fly with "**SET BT SNIFF**", but you need to close all active *Bluetooth* connections, then change the sniff setting and reopen the connections. If you want to be able to control the sniff mode and keep the connections active, disable "**SET BT SNIFF**" use command "**SET {link_id} SNIFF**" instead.

6.11 SET BT POWER

This command changes the TX power parameters of the WRAP THOR module.

6.11.1 Syntax

Synopsis:
SET BT POWER [RESET] | [{*default*} {*maximum*} [*inquiry*]]

Description:	
	If no parameters are given, displays current TX power settings.
RESET	Returns default TX power values and resets iWRAP
default	Default TX power in dBm (used for **CALL** and **NAME** operations and when responding to inquiries)
maximum	Maximum TX power in dBm
inquiry	Transmit power in dBm used for **INQUIRY** operation. This is an optional parameter introduced in iWRAP version 3.0.0: if not given, inquiry power is unchanged; by default is equal to the default TX power.

Respone:	
SYNTAX ERROR	This event occurs if incorrect parameters are given

Events:
None

List format:
SET BT POWER {*default*} {*maximum*} {*inquiry*}

6.11.2 Examples

Change TX power to class 2 setting:

```
SET BT POWER 0 4 0
```

Note:

Please see the table below for details on setting the requirements for TX power:

Power class:	Max. TX power:	Nominal TX power:	Minimum TX power:
1	20 dBm	N / A	0dBm
2	4dBm	0dBm	-6 dBm
3	0dbm	N / A	N / A

Table 8: Power classes as defined in *Bluetooth* specification

- The values passed with "**SET BT POWER**" will always be rounded to the next available value in the radio power table.
- If possible, always use default values!

6.12 SET BT IDENT

This command changes the device identification information. Only the freeform description can be changed; the first four parameters exist for the sake of conformity.

6.12.1 Syntax

Synopsis:
SET BT IDENT {*src*}:{*vendor_id*} {*product_id*} {*version*} [*descr*]

Description:	
src	This attribute indicates which organization assigned the VendorID attribute. There are two possible values: BT for the Bluetooth Special Interest Group (SIG) or USB for the USB Implementer's Forum.
vendor_id	Intended to uniquely identify the vendor of the device. The Bluetooth SIG or the USB IF assigns VendorIDs. Bluegiga's VendorID is 47.
product_id	Intended to distinguish between different products made by the vendor in question. These IDs are managed by the vendors themselves, and should be changed when new features are added to the device.
version	Vendor-assigned version string indicating device version number. This is given in the form of *major.minor.revision*, for example "3.0.0".
descr	Optional freeform product description string.

Respone:	
SYNTAX ERROR	This event occurs if incorrect parameters are given

Events:
None

List format:
SET BT IDENT {*src*}:{*vendor_id*} {*product_id*} {*version*} [*descr*]

6.12.2 Examples

Changing the description string:

```
SET BT IDENT BT:47 f000 3.0.0 My Description String
```

6.13 SET CONTROL AUTOCALL

SET CONTROL AUTOCALL enables or disables the AUTOCALL functionality in iWRAP.

When the AUTOCALL feature is enabled, iWRAP tries to form a connection with a paired (see **"SET BT PAIR"**) device until the connection is established. If the connection is lost or closed, iWRAP tries to reopen it.

If there are several paired devices in iWRAP memory, an inquiry (transparent to the user) is made and the first paired device found is connected.

6.13.1 Syntax

Synopsis:
SET CONTROL AUTOCALL {*target*} {*timeout*} {*profile*}

Description:	
target	RFCOMM, HFP or HFP-AG, HID or A2DP target for the connection. The target can be one of the following: **channel** RFCOMM channel number HFP channel number HFP-AG channel number Format: xx (hex) **uuid16** 16-bit UUID for searching channel Format: xxxx (hex) **uuid32** 32-bit UUID for searching channel Format: xxxxxxxx (hex) **uuid128** 128-bit UUID for searching channel Format: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx (hex)

	L2CAP psm 16-bit L2CAP psm Format: xxxx (hex)
timeout	Timeout between calls (in milliseconds)
profile	Defines the connection mode to be established. Possible modes are: **RFCOMM** Normal RFCOMM connection **HFP** Opens a connection in the Hands Free device mode. **HFP-AG** Opens a connection in the Hands Free Audio Gateway mode. **A2DP** Opens a connection in the Advanced Audio Distribution Profile (A2DP) mode or Audio Video Remote Control Profile (AVRCP) mode. L2CAP psm for A2DP is 19 and for AVRCP 17. **HID** Opens a connection in the HID keyboard mode or HID mouse mode. L2CAP psm for HID is 11. **L2CAP** Opens a generic L2CAP connection.

Response:	
SYNTAX ERROR	This event occurs if incorrect parameters are given

Events:
None

List format:	
	If AUTOCALL is not enabled, **"SET CONTROL AUTOCALL"** will not be displayed
SET CONTROL AUTOCALL *{target}* *{timeout}* *{profile}*	When AUTOCALL is enabled

6.13.2 Examples

To enable AUTOCALL to Serial Port Profile (using UUID) with timeout of 5000 ms:

```
SET CONTROL AUTOCALL 1101 5000 RFCOMM
SET
SET BT BDADDR 00:07:80:80:c2:37
SET BT NAME WT12
SET BT CLASS 001f00
SET BT AUTH * 1
SET BT LAP 9e8b33
SET BT PAGEMODE 4 2000 1
SET BT PAIR 00:60:57:a6:56:49 d36c481fb6eb76a139f64c403d821712
SET BT ROLE 0 f 7d00
SET BT SNIFF 0 20 1 8
SET CONTROL AUTOCALL 1101 5000 RFCOMM
SET CONTROL BAUD 115200,8n1
SET CONTROL CD 00 0
SET CONTROL ECHO 7
SET CONTROL ESCAPE 43 00 1
SET
```

Disabling AUTOCALL:

```
SET CONTROL AUTOCALL
SET
SET BT BDADDR 00:07:80:80:c2:37
SET BT NAME WT12
SET BT CLASS 001f00
SET BT AUTH * 1
SET BT LAP 9e8b33
SET BT PAGEMODE 4 2000 1
SET BT PAIR 00:60:57:a6:56:49 d36c481fb6eb76a139f64c403d821712
SET BT ROLE 0 f 7d00
SET BT SNIFF 0 20 1 8
SET CONTROL BAUD 115200,8n1
SET CONTROL CD 00 0
SET CONTROL ECHO 7
SET CONTROL ESCAPE 43 00 1
SET
```

Note:

- If AUTOCALL is enabled no manual "CALL" commands should be given to iWRAP.
- INQUIRY commands may fail when AUTOCALL is enabled, because AUTOCALL makes inquiries (transparent to the user) if multiple devices are paired.

6.14 SET CONTROL BAUD

This command changes the local device's UART settings.

6.14.1 Syntax

Synopsis:
SET CONTROL BAUD {*baud_rate*},8{*parity*}{*stop_bits*}

Description:	
baud_rate	UART baud rate in bps. See modules data sheet for suitable values.
parity	UART parity setting **n** No parity **e** Even parity **o** Odd parity
stop_bits	Number of stop bits in UART communications **1** One stop bit **2** Two stop bits

Response:
None

Events:	
SYNTAX ERROR	This event occurs if incorrect parameters are given

List format:
SET CONTROL BAUD {*baud_rate*},8{*parity*}{*stop_bits*}

6.14.2 Examples

Configuring local UART to 9600bps, 8 data bits, no parity and 1 stop bit

```
SET CONTROL BAUD 9600,8N1
```

6.15 SET CONTROL CD

This command enables or disables the carrier detect signal (CD) in iWRAP.

Carrier detect signal can be used to indicate that iWRAP has an active *Bluetooth* connection. With "**SET CONTROL CD**" command, one PIO line can be configured to act as a CD signal.

6.15.1 Syntax

Synopsis:
SET CONTROL CD {*cd_mask*} {*datamode*}

Description:	
cd_mask	This is a bit mask, which defines the GPIO lines used for CD signaling For example, value 20 (HEX) must be used for PIO5. 20 (HEX) = 100000 (BIN) For PIO6, the value is 40 40 (HEX) = 1000000 (BIN)
datamode	This parameter defines how the carrier detect signal works. **0** CD signal is driven high if there are one or more connections. **1** CD signal is driven high only in data mode.

Events:	
SYNTAX ERROR	This event occurs if incorrect parameters are given

List format:
SET CONTROL CD {*cd_mask*} {*datamode*}

6.16 SET CONTROL CONFIG

6.16.1 Syntax

This command enables or disables various functional features in iWRAP. These features are described below.

Synopsis:
SET CONTROL CONFIG [*optional_configuration*] {*configuration_value*} [LIST]

Description:	
configuration_value	This value is a bit field (represented as a hexadecimal value), which is used to control various features in iWRAP. These features are described below: **Bit 0** If this bit is set, the RSSI value will be visible in the inquiry results **Bit 1** "*Bluetooth* clock caching". If this bit is set, iWRAP will store the clock states of devices discovered in inquiry. This may speed up connection establishment if the connected device has responded to inquiry. **Bit 2** "Interlaced inquiry scan". If this bit is set, interlaced inquiry will be used. As a rule, interlaced inquiry is a little bit faster than regular inquiry. **Bit 3** "Interlaced page scan". If this bit is set, interlaced page (call) will be used. As a rule, interlaced page is a little bit faster than regular page. **Bit 4** "Deep sleep enabled". If this bit is set, 'Deep sleep' power saving mode will be used. Deep sleep is an aggressive power saving mode used when there are no connections. **Bit 5** "*Bluetooth* address in CONNECT". If this bit is set, the *Bluetooth* address of the remote end will be displayed on the CONNECT event.

	Bit 6 Not used. Must be set to 0. **Bit 7** Displays the **<u>PAIR</u>** event after successful pairing. **Bit 8** Enables SCO links. This bit must be 1 if you use audio profiles. **Bit 9** Must be set to 0. **Bit 10** Must be set to 0. **Bit 11** Enables interactive pairing mode. Where pin code is prompted rather then pin code set with "SET BT AUTH" used. **Bit 12** If this bit is set iWRAP randomly replaces one of the existing pairings, when 17th pairing occurs (max number of pairings is 16). **Bit 13** If this bit is set CLOCK event will be displayed on CONNET and RING events. **Bit 14** If this bit is set UART will be optimized for low latency instead of throughput. **Bit 15** If this bit is set low inquiry priority is used. This feature reduces inquiry priority and number of inquiry responses but improves simultaneous data transfer performance.
optional_configuration	**Bit 0** If this bit is set. All changing iWRAP configuration with SET commands will be disabled. The only way to enable SET commands are by deleting PS-key: "user configuration data 30"

	Bit 1 *"Enhanced Inquiry Response"*. If this bit is set, iWRAP display **INQUIRY_EXTENDED** reports during inquiry.
LIST	Static text that prints the configuration in human readable format.

Response:	
SYNTAX ERROR	Occurs if incorrect parameters are given
SET CONTROL CONFIG [*optional_configuration*] {*configuration_value*}	If no parameters given
SET CONTROL CONFIG {*optional_configuration*} {*configuration_value*} {*human readable output*}	If "**SET CONTROL CONFIG LIST**" issued

Events:
None

List format:
None

6.16.2 Examples

RSSI, deep sleep, interlaced inquiry, and page scans enabled.

```
SET CONTROL CONFIG 1D
```

6.17 SET CONTROL ECHO

This command changes the echo mode of iWRAP.

6.17.1 Syntax

Synopsis:
SET CONTROL ECHO {*echo_mask*}

Description:	
echo_mask	Bit mask for controlling the display of echo and events **Bit 0** If this bit is set, the start-up banner is visible. **Bit 1** If this bit is set, characters are echoed back to client in command mode. **Bit 2** This bit indicates if set events are displayed in command mode.

Events:	
SYNTAX ERROR	This event occurs if incorrect parameters are given

List format:
SET CONTROL ECHO {*echo_mask*}

Warning!

If every bit is set off (value 0), it is quite impossible to know the iWRAP status.

If Bit 2 is set off, it is very hard to detect whether iWRAP is in command mode or in data mode. This can, however, be solved if one IO is used to indicate that iWRAP is in data mode (**“SET CONTROL CD”**).

6.18 SET CONTROL ESCAPE

6.18.1 Syntax

This command is used to change the escape character used to switch between command and data mode. This command also enables and disables DTR signaling.

Synopsis:
SET CONTROL ESCAPE *{esc_char} {dtr_mask} {dtr_mode}*

Description:	
esc_char	Decimal ASCII character to define the escape character used in the escape sequence. Use "-" to disable escape sequence (the default value is 43, which is "+").
dtr_mask	Bit mask for selecting I/O pins used for DTR. For example for IO5, the bit mask is **00100000** and **dtr_mask** is **20** (HEX).
dtr_mode	**0** DTR Disabled **1** Return to command mode when DTR is dropped. **2** Close the active connection when DTR is dropped. **3** Reset iWRAP when DTR is dropped.

Events:	
SYNTAX ERROR	This event occurs if incorrect parameters are given

List format:
SET CONTROL ESCAPE *{esc_char} {dtr_mask} {dtr_mode}*

6.18.2 Examples

How to disable default escape character “+” and configure DTR to PIO5.

```
SET CONTROL ESCAPE – 20 1
```

6.19 SET CONTROL INIT

SET CONTROL INIT lists or changes the initialization command in iWRAP. This command is run when iWRAP is started or reset.

6.19.1 Syntax

Synopsis:
SET CONTROL INIT [*command*]

Description:	
	If no command is given, will erase the initialization command.
command	Any of the available iWRAP commands. This command is automatically executed every time iWRAP starts (after power-on, RESET or watchdog event)

Events:
None

List format:
SET CONTROL INIT {*command*}

6.19.2 Examples

To remove all pairings after reset:

```
SET CONTROL INIT SET BT PAIR *
```

To change baud rate to 115200 bps after reset:

```
SET CONTROL INIT SET CONTROL BAUD 115200,8n1
```

Warning!

Issuing **SET CONTROL INIT RESET** will cause iWRAP to enter an infinite reset loop, rendering it unusable until the persistent store user key #27 is removed by hand.

6.20 SET CONTROL MUX

SET CONTROL MUX can be used to enable or disable the multiplexing mode. This chapter describes the usage of the command as well as the operation of multiplexing mode.

6.20.1 Syntax

Synopsis:
SET CONTROL MUX {*mode*}

Description:	
mode	Multiplexing mode **0** Multiplexing mode disabled. Normal (data-command) mode enabled **1** Multiplexing mode enabled. Multiplexing protocol must be used to talk to iWRAP.

Events:	
READY	READY event occurs after a successful mode change.

List format:	
	Nothing is displayed when multiplexing mode is disabled.
SET CONTROL MUX 1	This string is displayed when multiplexing mode is enabled.

6.20.2 Examples

To enable multiplexing mode:

```
SET CONTROL MUX 1
¿READY
```

To disable multiplexing mode:

```
BF FF 00 11 53 45 54 20 43 4f 4e 54 52 4f 4c 20 4d 55 58 20 30 00
READY
```

The command is "**SET CONTROL MUX** 0" in the frame format used by MUX mode. The command must be sent in hex format, not in ASCII format.

Note:

- When multiplexing mode is enabled, no ASCII commands can be given to iWRAP but the multiplexing protocol must be used. Multiplexing mode can be disabled by deleting PSKEY_USR3 with PSTool.

- ASCII commands do not need to end with"\r" when multiplexing mode is used.

6.20.3 Using Multiplexing Mode

The multiplexing protocol format is presented below:

Length:	**Name:**	**Description:**	**Value:**
8 bits	SOF	Start of frame	0xBF
8 bits	LINK	Link ID	0x00 - 0x08 or 0xFF (control)
6 bits	FLAGS	Frame flags	0x00
10 bits	LENGTH	Size of data field in bytes	-
0-800 bits	DATA	Data	-
8 bits	nLINK	{LINK} XOR OxFF	-

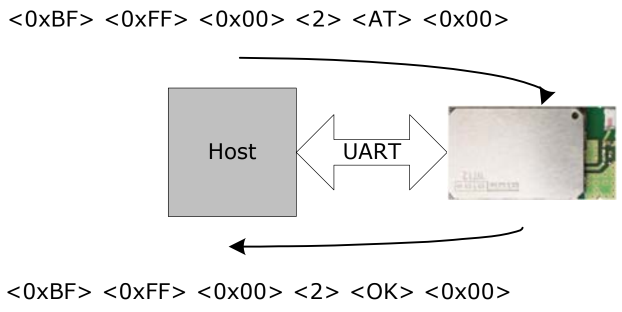
Table 9: Multiplexing frame format

When multiplexing mode is enabled, all the commands and data sent from host to iWRAP must be sent by using the frame format described above instead of plain ASCII commands. Also, the responses and data coming from iWRAP to the host are sent using the same format. iWRAP firmware autonomously processes the frames and decides whether they contain control commands or data which should be forwarded to its destination.

The advantage of multiplexing mode is that there is no need to do special command-data –command mode switching since data and commands are transmitted in the same mode. This saves a lot of time especially in multipoint scenarios, where - in the worst case - switching from data mode to command mode can take more than two seconds.

Also in scenarios where there are several connections, receiving data simultaneously from several devices is difficult if multiplexing mode is not used. In normal (data/command) mode, only one connection can be active (in data mode) at a time, and it can only be used to transmit or receive data. If there is any data received from the other connection during normal mode, the data is stored to small iWRAP buffers and received when the connections become active (data mode of the connection enabled).

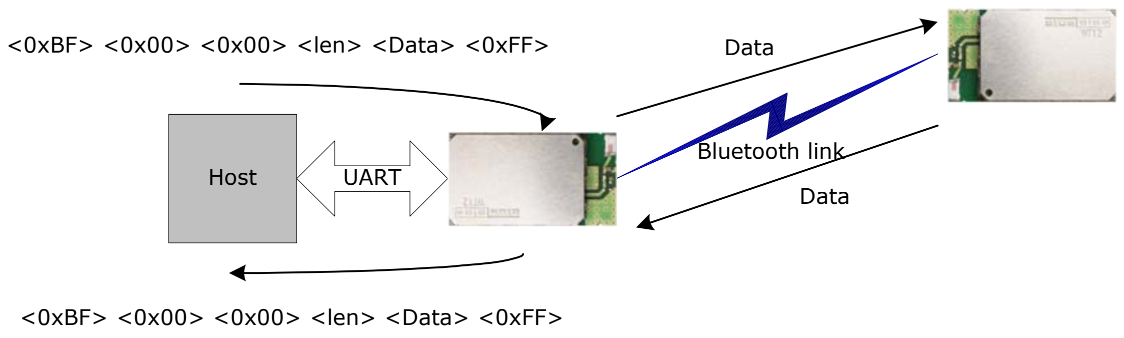
The next figure illustrates the host-iWRAP-host communications in multiplexing mode.



<0xBF> <0xFF> <0x00> <2> <AT> <0x00>
Host
UART
<0xBF> <0xFF> <0x00> <2> <OK> <0x00>

Figure 4: Host-iWRAP-Host communication

The figure below illustrates host-iWRAP-remote device communication when multiplexing mode is in use. The key thing is that the remote device does not need to know anything about the multiplexing communication and frame format, but it sees the connection as a standard *Bluetooth* connection.



<0xBF> <0x00> <0x00> <len> <Data> <0xFF>
Data
Host
UART
Bluetooth link
Data
<0xBF> <0x00> <0x00> <len> <Data> <0xFF>

Figure 5: Host-iWRAP-remote device communications

At the moment, four (4) simultaneous connections can be used in multiplexing mode.

Tips:

In MUX mode the processor of the module is highly utilized and on the edge of its performance. This may be seen as a instability of *Bluetooth* connections, especially if 3 or more connections are used or data rate is high. There are however a few tricks how the stability of the *Bluetooth* connections can be improved:

1. Use SNIFF mode: Using sniff mode reduces the rate the master device needs to poll the active connections are leaves more time for the processor to parse or generate the multiplexing protocol. Therefore as aggressive as possible sniff mode should be used.

2. Optimize *Bluetooth* packet size by using MTU option in CALL command: Using smaller *Bluetooth* packet size improves the multiplexing performance.

On the next page, there is a simple C-code example on how to create a simple multiplexing frame containing an iWRAP command.

```
//HOW TO CREATE A SIMPLE FRAME

char outbuf[128];         //Buffer for frame

char* cmd = "SET";        //ASCII command

int link = 0xff, pos=0;   //0xFF for control channel

int len = strlen(cmd);    //Calc. length of ASCII command

//Generate packet

outbuf[pos++]=0xbf;             //SOF

outbuf[pos++]=link;             //Link (0xFF=Control, 0x00 = connection 1, etc.)

outbuf[pos++]=0;                //Flags

outbuf[pos++]=len;              //Length

//Insert data into correct position in the frame

memmove(outbuf+pos cmd, len);

pos += len;                     //Move to correct position

outbuf[pos++]=link^0xff;        //nlink
```

6.21 SET CONTROL BIND

With **SET CONTROL BIND**, it is possible to bind iWRAP commands to GPIO pins.

6.21.1 Syntax

Synopsis:
SET CONTROL BIND {*pri*} [*io_mask*] [*direction*] [*command*]

Description:	
pri	Command priority. Determines the order in which the commands bound to PIO are executed. WT12 and WT11 value range: PIO2-PIO7 WT32 value range: PIO0-PIO10 If only ***pri*** parameter is given the current bind will be removed.
io_mask	Determines which PIO is to be bind. This is a hexadecimal value. Example: Set PIO5. 100000bin (5th bit is one) = 20hex
direction	Determines whether PIO is triggered on rising, falling, or on both edges of the signal. Possible values: **RISE** Command is executed on rising edge. **FALL** Command is executed on falling edge. **CHANGE** Command is executed on rising and falling edge.
command	Standard iWRAP command or string to be sent to the active *Bluetooth* link.

Response:
No response

6.21.2 Examples

Example usage of binding PIOs:

```
SET CONTROL  BIND 0 20 FALL CLOSE 0
SET CONTROL BIND 1 20 FALL SET BT PAIR *
SET

SET BT BDADDR 00:07:80:81:62:2a
SET BT NAME EKWT11_PR
SET BT CLASS 001f00
SET BT AUTH * 1234
SET BT LAP 9e8b33
SET BT PAGEMODE 4 2000 1
SET BT ROLE 0 f 7d00
SET BT SNIFF 0 20 1 8
SET CONTROL BAUD 115200,8n1
SET CONTROL BIND 0 20 F close 0
SET CONTROL BIND 1 20 F set bt pair *
SET CONTROL CD 80 0
SET CONTROL ECHO 7
SET CONTROL ESCAPE - 20 1
SET CONTROL MSC DTE 00 00 00 00 00 00
SET PROFILE HFP WT12 Hands Free
SET PROFILE SPP Bluetooth Serial Port
SET
```

Example of binding PIO5 to close the connection and delete all pairings after PIO5 has fallen. The SET command indicates that the binding of commands was successful.

6.22 SET CONTROL MSC

With iWRAP firmware, it is possible to transmit all the UART modem signals over the SPP (Serial Port Profile) *Bluetooth* link. The signals DSR, DTR, RTS, CTS, RI and DCD can be specified to GPIO pins on the WRAP THOR modules. The **SET CONTROL MSC** command is used to do this.

6.22.1 Syntax

Synopsis:
SET CONTROL MSC [[*mode*] [[*DSR*] [[*DTR*] [[*RTS*] [[*CTS*] [[*RI*] [*DCD*]]]]]]

Description:	
mode	Mode of the device iWRAP connects to. The mode can be: **DTE** or **nDTE** and **DCE** or **nDCE** **NOTE:** DTE means that remote *Bluetooth* device is DTE (so iWRAP is DCE and device connected to iWRAP is DTE). nDTE and nDCE means that the signals are active low, not active high.
DSR	Data Set Ready. Select PIO with a bitmask. See the note below on how to select the PIO.
DTR	Data Terminal Ready. See the note below on how to select the PIO.
RTS	Request To Send. See the note below on how to select the PIO.
CTS	Clear To Send. See the note below on how to select the PIO.
RI	Ring Indicator. See the note below on how to select the PIO.
DCD	Data Carrier Detect. See the note below on how to select the PIO.

Response:	
SYNTAX ERROR	This event occurs if incorrect parameters are given.

Events:
None

Note:

- The PIO pin is selected with a bit mask. For example, if you want to use PIO3, you will then have a bit mask where the third bit is 1, that is, 1000. This bit mask value is then given in the command in hexadecimal format. 1000(bin) = 8(hex).
- If MUX mode is in use physical PIO statuses do not change even if SET CONTROL MSC is used, since in MUX mode it would be hard tell which of the connections defines the MSC signal statuses.

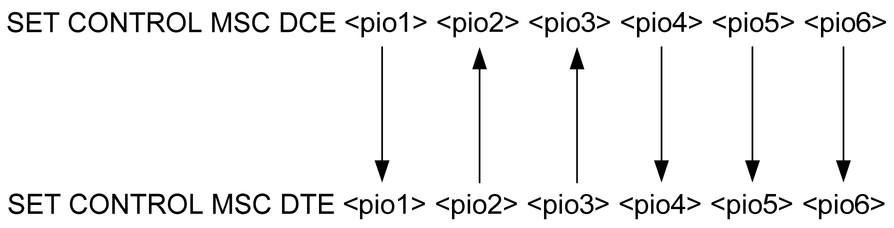

SET CONTROL MSC DCE <pio1> <pio2> <pio3> <pio4> <pio5> <pio6>
SET CONTROL MSC DTE <pio1> <pio2> <pio3> <pio4> <pio5> <pio6>

Figure 6: MSC signal directions

6.23 SET CONTROL GAIN

SET CONTROL GAIN is used to control the internal input and output gain.

6.23.1 Syntax

Synopsis:
SET CONTROL GAIN [{*input*} {*output*} [*DEFAULT*]]

Description:	
	If no parameters are given, returns the input and output gain ranges in decimal.
input	Input gain. Range: 0-17 (hex)
output	Output gain. Range: 0-17 (hex)
DEFAULT	If given, set given input and output gain as default values and save them in the persistent store.

Response:	
SYNTAX ERROR	This event occurs if incorrect parameters are given.

Events:
None:

List format:
SET CONTROL GAIN {*default input*} {*default output*}

Note:

- On A2DP sink the gain should be set to very low value or otherwise the A2DP audio quality will suffer radically.

6.24 SET CONTROL VREGEN

SET CONTROL VREGEN is used to set the behavior of the internal software-controlled regulator on the module. The PIO's specified by the ***PIO mask*** parameter will be pulled high by the regulator as specified by the ***mode*** parameter.

6.24.1 Syntax

Synopsis:
SET CONTROL VREGEN {*mode*} {*PIO mask*}

Description:	
mode	**0** Regulator is enabled on rising edge when its input voltage (VREG_ENA) rises past around 1V and will hold the PIO's high. **Warning:** using this mode may or may not, depending on your setup, keep the module powered on until its power source is disconnected or regulator mode is switched! **1** Regulator is enabled on rising edge (of VREG_ENA) and holds the PIO voltage up until a falling edge is encountered, at which point the regulator will pull the voltage down. **2** Regulator is enabled on rising edge and holds the voltage up until another rising edge followed by a falling edge is encountered, at which point the regulator will bring the voltage down.
PIO mask	Bit mask used to specify which PIO's are held up by the regulator. This parameter is given in hexadecimal format.

Response:	
SYNTAX ERROR	This event occurs if incorrect parameters are given.

Events:
None

List format:
SET CONTROL VREGEN {*mode*} {*PIO mask*}

6.24.2 Examples

Building a setting in which a switch is toggled to power on the module, and keep it powered on until the switch is first toggled back, then toggled up and down again (rising edge followed by falling edge); PIO2 is pulled high to hold up an external regulator

SET CONTROL VREGEN 2 4 (4 in hexadecimal is 100 in binary)

Note:

- Valid for WT32 only
- See the WT32 Design Guide located at http://techforum.bluegiga.com for further information on the design of the module and regulator**.**

6.25 SET CONTROL MICBIAS

SET CONTROL MICBIAS controls the linear regulator that drives current through the dedicated mic bias pin.

6.25.1 Syntax

Synopsis:
SET CONTROL MICBIAS [{*voltage*} {*current*}]

Description:	
	If no parameters are given, returns current mic bias settings.
voltage	Voltage driven through the mic bias pin. Range 0-F (hex).
current	Current driven through the mic bias pin. Range: 0-F (hex). The setting values and their corresponding typical voltage and current ranges are in the table below.

Value	**Voltage (V)**	**Current (mA)**
0	1.61 - 1.80	0.237 - 0.394
1	1.65 - 1.86	0.296 - 0.492
2	1.71 - 1.93	0.354 - 0.589
3	1.76 - 1.98	0.412 - 0.687
4	1.84 - 2.06	0.471 - 0.785
5	1.89 - 2.14	0.530 - 0.883
6	1.97 - 2.23	0.589 - 0.980
7	2.04 - 2.32	0.647 - 1.078
8	2.18 - 2.46	0.706 - 1.176
9	2.27 - 2.58	0.764 - 1.273
10	2.39 - 2.72	0.823 - 1.371
11	2.50 - 2.87	0.882 - 1.469
12	2.70 - 3.09	0.940 - 1.566
13	2.85 - 3.29	0.998 - 1.664
14	3.07 - 3.56	1.057 - 1.762
15	3.28 - 3.84	1.116 - 1.859

Response:	
SYNTAX ERROR	This event occurs if incorrect parameters are given.

Events:
None

List format:
SET CONTROL MICBIAS {*voltage*} {*current*}

6.26 SET CONTROL PCM

This command configures the physical PCM hardware interface settings and PCM data format.

6.26.1 Syntax

Synopsis:
SET CONTROL PCM {*config*} {*data*}

Description:	
	If no parameters are given lists the current settings.
config	Value for the PCM interface configuration. Corresponds to PS-key: PSKEY_PCM_CONFIG32
data	Value for PCM data format. Corresponds to PS-key: PSKEY_PCM_FORMAT

Response:	
SYNTAX ERROR	This event occurs if incorrect parameters are given.

Events:
None:

List format:
SET CONTROL PCM {*config*} {*data*}

Note:

- Use PCM configuration tool available at http://techforum.bluegiga.com to determine correct value for ***config***.
- See corresponding modules data sheet for description of ***config*** and ***data.***

6.27 SET PROFILE

The **SET PROFILE** command can be used to enable or disable the available *Bluetooth* profiles: SPP, OPP, HFP and HFP-AG, A2DP, AVRCP and HID.

6.27.1 Syntax

Synopsis:
SET PROFILE {*profile_name*} [*SDP_name*]

Description:	
profile_name	Specify the profile to be enabled or disabled. Possible profile acronyms are: **HFP** Hands Free Profile **HFP-AG** Hands Free Profile Audio Gateway **SPP** Serial Port Profile **OPP** Object Push Profile (server) **A2DP SINK** Advanced Audio Distribution Profile Sink mode. SDP name can not be changed with A2DP sink. **A2DP SOURCE** Advanced Audio Distribution Profile Source mode. SDP name can not be changed with A2DP sink. **HID** HID keyboard and mouse profiles.

SDP_name	**on** Enables the profile with default SDP name. With A2DP enabled A2DP in sink mode. ***<string>*** Enables the profile with ***string*** used as SDP name. If this parameter is not given, the profile will be disabled.

Response:
No response

6.27.2 Examples

Example of enabling HFP profile.

```
SET PROFILE HFP My Hands-Free
SET
SET BT BDADDR 00:07:80:80:c2:37
SET BT NAME WT12
SET BT CLASS 001f00
SET BT AUTH * 6666
SET BT LAP 9e8b33
SET BT PAGEMODE 4 2000 1
SET BT ROLE 0 f 7d00
SET BT SNIFF 0 20 1 8
SET CONTROL BAUD 115200,8n1
SET CONTROL CD 80 0
SET CONTROL ECHO 7
SET CONTROL ESCAPE 43 00 1
SET CONTROL MSC DTE 00 00 00 00 00 00
SET PROFILE HFP My Hands-Free
SET PROFILE SPP Bluetooth Serial Port
SET
RESET
```

Note:

- iWRAP must be reset for the profile to be activated or deactivated.
- If you want to use the HFP or HFP-AG audio profiles, enable also the support for SCO links, by setting "**SET CONTROL CONFIG**" bit 8 to 1. This is "**SET CONTROL CONFIG 100**" if no other configuration bits are enabled. This is only required with iWRAP 2.2.0.

6.28 SET {link_id} ACTIVE

This command disables all the power save modes for the defined, active *Bluetooth* link and sets it into active mode.

6.28.1 Syntax

Synopsis:
SET {*link_id*} ACTIVE

Description:	
link_id	Numeric connection identifier

Events:
None

6.28.2 Examples

Changing from SNIFF to active:

```
LIST
LIST 1
LIST 0 CONNECTED RFCOMM 320 0 0 3 8d 8d 00:60:57:a6:56:49 1 OUTGOING SNIFF
MASTER PLAIN
SET 0 ACTIVE
LIST
LIST 0 CONNECTED RFCOMM 320 0 0 3 8d 8d 00:60:57:a6:56:49 1 OUTGOING ACTIVE
MASTER PLAIN
```

6.29 SET {link_id} MASTER

This command attempts to switch the link to Piconet master. Notice that this may not be allowed by the remote end.

6.29.1 Syntax

Synopsis:
SET {*link_id*} MASTER

Description:	
link_id	Numeric connection identifier

Events:
None

6.29.2 Examples

Changing from slave to master:

```
LIST
LIST 1
LIST 0 CONNECTED RFCOMM 320 0 0 3 8d 8d 00:60:57:a6:56:49 1 OUTGOING ACTIVE
SLAVE PLAIN
SET 0 MASTER
LIST
LIST 0 CONNECTED RFCOMM 320 0 0 3 8d 8d 00:60:57:a6:56:49 1 OUTGOING ACTIVE
MASTER PLAIN
```

6.30 SET {link_id} SLAVE

This command attempts to switch the link to Piconet slave. Notice that this may not be allowed by the remote end.

6.30.1 Syntax

Synopsis:
SET {*link_id*} SLAVE

Description:	
link_id	Numeric connection identifier

Events:
None

6.31 SET {link_id} SNIFF

This command attempts to enable SNIFF mode for the defined *Bluetooth* link. Whether this command is successful or not, depends on if the remote end allows sniff to be used.

6.31.1 Syntax

Synopsis:

SET {*link_id*} SNIFF {max}{min} [{attempt} {timeout}]
 	
or
 	
SET {*link_id*} SNIFF {avg}

Description:	
link_id	Numeric connection identifier
max	Maximum acceptable interval in milliseconds
min	Minimum acceptable interval in milliseconds
avg	Average value is milliseconds. Shortcut for easier SNIFF setting.
attempt	Number of SNIFF attempts (default value 1)
timeout	SNIFF timeout (default value 8)

Events:
None

Note:

Refer to the *Bluetooth* specification for more information.

6.32 SET {link_id} MSC

With this command, it is possible to send 07.10 Modem Status Command to the remote device without having the signals actually connected to the module.

6.32.1 Syntax

Synopsis:
SET {*link_id*} MSC {*status*}

Description:	
link_id	Numeric connection identifier of the link where the modem status is to be sent.
status	Status of the signals according to 07.10 standards.

Response:
No response

6.32.2 Examples

Example usage of sending MSC:

```
SET  0 MSC 8D

Normal MSC status was sent.
```

6.33 SET {link_id} SELECT

With this command, you can define the active Bluetooth connection for the iWRAP command wrapped. This command is useful for example when two simultaneous Hands-Free connections or one Hands-Free connection and one A2DP connection is used

6.33.1 Syntax

Synopsis:
SET {*link_id*} SELECT

Description:	
link_id	Numeric connection identifier of the link where the modem status is to be sent.

Response:
No response

Note:

- iWRAP uses an internal command parser/wrapped with some Bluetooth profiles like Hands-Free profile. The internal parser handles commands like HANGUP, VOLUME etc. and transfers them into AT-commands defined in the Hands Free profile specification. To be able to send the commands you need to have correct Bluetooth link / parser selected and this can be done with **SET {link_id} SELECT** command.

7. IWRAP EVENTS

Events are a mechanism that iWRAP uses to notify the user for completed commands, incoming connections, and so on.

If iWRAP is in data mode (data is being transmitted and no multiplexing mode is used) the only possible event is **NO CARRIER** indicating that connection was closed or lost.

Note:

- iWRAP is designed so that unwanted events can be safely ignored. Events **CONNECT**, **NO CARRIER** and **RING** change the mode of operation and therefore they cannot be ignored.
- Events can be masked away by removing **Bit 2** from command **SET CONTROL ECHO**.

7.1 CONNECT

The **CONNECT** event is used to notify the user for a successful link establishment.

7.1.1 Syntax

Synopsis:
CONNECT {*link_id*} {SCO | RFCOMM | A2DP | HID | HFP | HFP-AG {*target*} [*address*]}

Description:	
link_id	Numeric connection identifier
target	Connected RFCOMM channel number or L2CAP psm
address	Address of the remote end. This is displayed only if bit 5 is set in "**SET CONTROL CONFIG**".

Note:

iWRAP automatically enters data mode after the **CONNECT** event if multiplexing mode is disabled.

7.2 INQUIRY_PARTIAL

The **INQUIRY_PARTIAL** event is used to notify the user for a found *Bluetooth* device. This event precedes response for the **INQUIRY** command.

7.2.1 Syntax

Synopsis:
INQUIRY_PARTIAL {*address*} {*class_of_device*} [{*cahced_name*} {*rssi*}]

Description:	
address	*Bluetooth* address of the found device
class_of_device	C *Bluetooth* Class of Device of the found device
cached_name	User friendly name of the found device if already known
rssi*	Received Signal Strength of the found device

*) RSSI is a value between -128 and 0. The lower the value, the lower the signal strength.

Note:

- ***cached_name*** and ***rssi*** are only visible if "Inquiry with RSSI" is enabled with "**SET CONTROL CONFIG**".

7.3 NO CARRIER

The **NO CARRIER** event is used to notify the user for a link loss or, alternatively, a failure in the link establishment.

7.3.1 Syntax

Synopsis:
NO CARRIER {*link_id*} ERROR {*error_code*} [*message*]

Description:	
link_id	Numeric connection identifier
error_code	Code describing the error
message	Optional verbose error message

7.4 READY

The **READY** event is used to notify the user for switching to command mode or to indicate that iWRAP is ready to be used after a reset or after a successful switch between normal or multiplexing mode has been done.

7.4.1 Syntax

Synopsis:
READY.

Description:
None

7.5 <u>NAME</u>

The **<u>NAME</u>** event is used to notify the user for a successful lookup for *Bluetooth* friendly name of the remote device.

7.5.1 Syntax

Synopsis:
NAME {*address*} {“*friendly_name*”}

Description:	
address	*Bluetooth* device address of the device
friendly_name	Friendly name of the device

7.6 NAME ERROR

The **NAME ERROR** event is used to notify the user for a *Bluetooth* friendly name lookup failure.

7.6.1 Syntax

Synopsis:
NAME ERROR {*error_code*} {*address*} [*message*]

Description:	
error_code	Code describing the error
address	*Bluetooth* address of the device
message	Optional verbose error message

7.7 PAIR

The **PAIR** event is used to notify the user for a successful pairing.

7.7.1 Syntax

Synopsis:
PAIR {*address*} {*key_type*} {*link_key*}

Description:	
address	*Bluetooth* device address of the paired device
key_type	Type of link key **0** Combination key **1** Local unit key **2** Remote unit key **ff** Unknown key
link_key	Link key shared between the local and the paired device

Note:

The **PAIR** event is enabled or disabled with the "**SET CONTROL CONFIG**" command.

If the **PAIR** event is enabled the event will also be shown during the **CALL** procedure and also before the **RING** event, if pairing occurs.

7.8 **RING**

The **RING** event is used to notify the user for an incoming connection. Incoming connections are only accepted if there is no existing links.

7.8.1 Syntax

Synopsis:
RING {*link_id*} {*address*} {SCO | {*channel*} RFCOMM}

Description:	
link_id	Numeric connection identifier
address	*Bluetooth* device address of the device
channel	Local RFCOMM or SCO channel

7.9 SYNTAX ERROR

SYNTAX ERROR is not an actual event, but an error message describing a faulty typed command or an error in command parameters.

7.9.1 Syntax

Synopsis:
SYNTAX ERROR

7.10 AUTH

AUTH event indicates that someone is trying to pair with iWRAP.

7.10.1 Syntax

Synopsis:
AUTH {*bd_addr*}?

Description:	
bd_addr	*Bluetooth* device address of the remote device

The **AUTH** event occurs only if interactive pairing is enabled with "**SET CONTROL CONFIG**" command.

7.11 CLOCK

CLOCK event indicates the Piconet clock value for a specific *Bluetooth* connection.

7.11.1 Syntax

Synopsis:
AUTH {*bd_addr*} {*clock*}

Description:	
bd_addr	*Bluetooth* device address of the remote device
clock	Piconet clock value

All the devices in a *Bluetooth* Piconet are synchronized to a same clock (master clock). The CLOCK event displays the clock value and it can for example be used for time synchronization of the Piconet slaves and master. The accuracy of the Piconet clock is 625us.

7.12 IDENT

IDENT event informs that a remote *Bluetooth* device has been identified by using the Device ID profile and reports the identification data sent by the remote device.

7.12.1 Syntax

Synopsis:
IDENT {*src*}:{*vendor_id*} {*product_id*} {*version*} "[*descr*]"

Description:	
src	This attribute indicates which organization assigned the VendorID attribute. There are two possible values: BT for the Bluetooth Special Interest Group (SIG) or USB for the USB Implementer's Forum.
vendor_id	Intended to uniquely identify the vendor of the device. The Bluetooth SIG or the USB IF assigns VendorIDs. Bluegiga's VendorID is 47.
product_id	Intended to distinguish between different products made by the vendor in question. These IDs are managed by the vendors themselves, and should be changed when new features are added to the device.
version	Vendor-assigned version string indicating device version number.
descr	Optional freeform product description string.

7.13 IDENT ERROR

IDENT ERROR event informs that a remote *Bluetooth* could not be identified by the Device ID profile.

7.13.1 Syntax

Synopsis:
IDENT ERROR {*error_code*} {*address*} [*message*]

Description:	
error_code	Code describing the error
address	*Bluetooth* address of the device
message	Optional verbose error message

7.14 BATTERY

The **BATTERY** event is used to report the current battery voltage to the user.

7.14.1 Syntax

Synopsis:
BATTERY {*mv*}

Description:	
mv	Current battery voltage in millivolts.

7.15 PIO

The **PIO** event is used to report the current PIO data, direction and bias register states.

7.15.1 Syntax

Synopsis:
PIO {*reg value*}

Description:	
reg	Which register is read: GET for data, GETDIR for direction, GETBIAS for bias register.
value	Register's value in hexadecimal.

7.16 VOLUME

The **VOLUME** event is used to report the current line out volume to the user.

7.16.1 Syntax

Synopsis:
VOLUME {*vol*}

Description:	
vol	Current volume, range 0-9.

8. IWRAP ERROR MESSAGES

This chapter briefly presents the iWRAP error messages.

8.1 HCI Errors

HCI errors start with code: ***0x100***

ERROR MESSAGE	CODE	Explanation
HCI_SUCCESS	0x00	Success
HCI_ERROR_ILLEGAL_COMMAND	0x01	Unknown HCI command
HCI_ERROR_NO_CONNECTION	0x02	Unknown connection identifier
HCI_ERROR_HARDWARE_FAIL	0x03	Hardware Failure
HCI_ERROR_PAGE_TIMEOUT	0x04	Page timeout
HCI_ERROR_AUTH_FAIL	0x05	Authentication failure
HCI_ERROR_KEY_MISSING	0x06	PIN or key missing
HCI_ERROR_MEMORY_FULL	0x07	Memory capacity exceeded
HCI_ERROR_CONN_TIMEOUT	0x08	Connection timeout
HCI_ERROR_MAX_NR_OF_CONNS	0x09	Connection Limit Exceeded
HCI_ERROR_MAX_NR_OF_SCO	0x0a	Synchronous connection limit to a device exceeded
HCI_ERROR_MAX_NR_OF_ACL	0x0b	ACL Connection Already Exists
HCI_ERROR_COMMAND_DISALLOWED	0x0c	Command Disallowed
HCI_ERROR_REJ_BY_REMOTE_NO_RES	0x0d	Connection Rejected due to Limited Resources
HCI_ERROR_REJ_BY_REMOTE_SEC	0x0e	Connection Rejected Due To Security Reasons
HCI_ERROR_REJ_BY_REMOTE_PERS	0x0f	Connection Rejected due to Unacceptable BD_ADDR
HCI_ERROR_HOST_TIMEOUT	0x10	Connection Accept Timeout Exceeded

HCI_ERROR_UNSUPPORTED_FEATURE	0x11	Unsupported Feature or Parameter Value
HCI_ERROR_ILLEGAL_FORMAT	0x12	Invalid HCI Command Parameter
HCI_ERROR_OETC_USER	0x13	Remote User Terminated Connection
HCI_ERROR_OETC_LOW_RESOURCE	0x14	Remote Device Terminated Connection due to Low Resources
HCI_ERROR_OETC_POWERING_OFF	0x15	Remote Device Terminated Connection due to Power Off
HCI_ERROR_CONN_TERM_LOCAL_HOST	0x16	Connection Terminated By Local Host
HCI_ERROR_AUTH_REPEATED	0x17	Repeated Attempts
HCI_ERROR_PAIRING_NOT_ALLOWED	0x18	Pairing Not Allowed
HCI_ERROR_UNKNOWN_LMP_PDU	0x19	Unknown LMP PDU
HCI_ERROR_UNSUPPORTED_REM_FEATURE	0x1a	Unsupported Remote Feature / Unsupported LMP Feature
HCI_ERROR_SCO_OFFSET_REJECTED	0x1b	SCO Offset Rejected
HCI_ERROR_SCO_INTERVAL_REJECTED	0x1c	SCO Interval Rejected
HCI_ERROR_SCO_AIR_MODE_REJECTED	0x1d	SCO Air Mode Rejected
HCI_ERROR_INVALID_LMP_PARAMETERS	0x1e	Invalid LMP Parameters
HCI_ERROR_UNSPECIFIED	0x1f	Unspecified Error
HCI_ERROR_UNSUPP_LMP_PARAM	0x20	Unsupported LMP Parameter Value
HCI_ERROR_ROLE_CHANGE_NOT_ALLOWED	0x21	Role Change Not Allowed
HCI_ERROR_LMP_RESPONSE_TIMEOUT	0x22	LMP Response Timeout
HCI_ERROR_LMP_TRANSACTION_COLLISION	0x23	LMP Error Transaction Collision

HCI_ERROR_LMP_PDU_NOT_ALLOWED	0x24	LMP PDU Not Allowed
HCI_ERROR_ENC_MODE_NOT_ACCEPTABLE	0x25	Encryption Mode Not Acceptable
HCI_ERROR_UNIT_KEY_USED	0x26	Link Key Can Not be Changed
HCI_ERROR_QOS_NOT_SUPPORTED	0x27	Requested QoS Not Supported
HCI_ERROR_INSTANT_PASSED	0x28	Instant Passed
HCI_ERROR_PAIR_UNIT_KEY_NO_SUPPORT	0x29	Pairing With Unit Key Not Supported

Table 10: HCI errors

Please see *Bluetooth* 2.0+EDR core specification page 493 for more information about error codes.

8.2 SDP Errors

SDP errors start with code: ***0x300***

ERROR MESSAGE	CODE	Explanation
SDC_OK	0x00	
SDC_OPEN_SEARCH_BUSY	0x01	
SDC_OPEN_SEARCH_FAILED	0x02	SDP search failed
SDC_OPEN_SEARCH_OPEN	0x03	
SDC_OPEN_DISCONNECTED	0x04	
SDC_OPEN_SEARCH_FAILED_PAGE_TIMEOUT	0x05	
SDC_OPEN_SEARCH_FAILED_REJ_PS	0x06	
SDC_OPEN_SEARCH_FAILED_REJ_SECURITY	0x07	SDP search failed because of security
SDC_OPEN_SEARCH_FAILED_REJ_RESOURCES	0x08	
SDC_OPEN_SEARCH_FAILED_SIGNAL_TIMEOUT	0x09	
SDC_ERROR_RESPONSE_PDU	0x10	
SDC_NO_RESPONSE_DATA	0x11	
SDC_CON_DISCONNECTED	0x12	
SDC_CONNECTION_ERROR	0x13	
SDC_CONFIGURE_ERROR	0x14	
SDC_SEARCH_DATA_ERROR	0x15	
SDC_DATA_CFM_ERROR	0x16	
SDC_SEARCH_BUSY	0x17	
SDC_RESPONSE_PDU_HEADER_ERROR	0x18	

SDC_RESPONSE_PDU_SIZE_ERROR	0x19	
SDC_RESPONSE_TIMEOUT_ERROR	0x1a	
SDC_SEARCH_SIZE_TOO_BIG	0x1b	
SDC_RESPONSE_OUT_OF_MEMORY	0x1c	
SDC_RESPONSE_TERMINATED	0x1d	
SDC_OPEN_SEARCH_FAILED_PAGE_TIMEOUT	305	SDP search failed because of page timeout
SDC_OPEN_SEARCH_FAILED_REJ_TIMEOUT	305	SDP search failed because of page timeout

Table 11: SDP errors

8.3 RFCOMM Errors

RFCOMM errors start with code: ***0x400***

ERROR MESSAGE	CODE
RFC_OK	0x00
RFC_CONNECTION_PENDING	0x01
RFC_CONNECTION_REJ_PSM	0x02
RFC_CONNECTION_REJ_SECURITY	0x03
RFC_CONNECTION_REJ_RESOURCES	0x04
RFC_CONNECTION_REJ_NOT_READY	0x05
RFC_CONNECTION_FAILED	0x06
RFC_CONNECTION_TIMEOUT	0x07
RFC_NORMAL_DISCONNECT	0x08
RFC_ABNORMAL_DISCONNECT	0x09
RFC_CONFIG_UNACCEPTABLE	0x0a
RFC_CONFIG_REJECTED	0x0b
RFC_CONFIG_INVALID_CID	0x0c
RFC_CONFIG_UNKNOWN	0x0d
RFC_CONFIG_REJECTED_LOCALLY	0x0e
RFC_CONFIG_TIMEOUT	0x0f
RFC_REMOTE_REFUSAL	0x11
RFC_RACE_CONDITION_DETECTED	0x12
RFC_INSUFFICIENT_RESOURCES	0x13
RFC_CANNOT_CHANGE_FLOW_CONTROL_MECHANISM	0x14

RFC_DLC_ALREADY_EXISTS	0x15
RFC_DLC_REJ_SECURITY	0x16
RFC_GENERIC_REFUSAL	0x1f
RFC_UNEXPECTED_PRIMITIVE	0x20
RFC_INVALID_SERVER_CHANNEL	0x21
RFC_UNKNOWN_MUX_ID	0x22
RFC_LOCAL_ENTITY_TERMINATED_CONNECTION	0x23
RFC_UNKNOWN_PRIMITIVE	0x24
RFC_MAX_PAYLOAD_EXCEEDED	0x25
RFC_INCONSISTENT_PARAMETERS	0x26
RFC_INSUFFICIENT_CREDITS	0x27
RFC_CREDIT_FLOW_CONTROL_PROTOCOL_VIOLATION	0x28
RFC_RES_ACK_TIMEOUT	0x30

Table 12: RFCOMM errors

9. GENERAL INFORMATION

This chapter contains a lot of useful information about iWRAP and its usage.

9.1 Changing Parameters over RS232 with PSTool

PSTool software allows the user to change the internal parameters (PS keys) of the module. Most of the parameters should not be touched, since they can affect the performance of the module. On the other hand, there are some useful parameters, which can not be accessed from iWRAP, such as hardware flow control, host interface parameters and so on.

Notice that although the parameters can be easily changed over the UART interface, incorrect configuration may prevent iWRAP from working and block any other than SPI communications with the module.

iWRAP has a useful feature called AutoBCSP. With AutoBCSP, iWRAP can automatically recognize BCSP (BlueCore Serial Protocol) traffic and is able to interpret it. BCSP can be used to change the internal parameters and is also supported by the PSTool software.

To change the internal parameters, proceed as follows:

1. Connect an RS2323 cable between your WTxx Bluetooth module and your PC
2. Power up the Bluetooth module
3. Open PSTool application
4. Use the default connection settings: **BCSP**, **COMn** and **115200**
5. Change the needed parameters (remember to press 'SET' after changing the parameter value)
6. Close PSTool and reset WRAP THOR

iWRAP is automatically activated after a reset, unless parameters affecting iWRAP operation are changed.

NOTE:

- When using BCSP, the UART baud rate does NOT depend on the configuration set with "**SET CONTROL BAUD**" command, but is defined by using PS key "PSKEY_UART_BAUD RATE". By default, the parameter value is 115200 bps.
- The AutoBCSP feature only works if PSKEY_UART_BAUD_RATE and "SET CONTROL BAUD" have same UART baud rate values values.
- PSTool can be also used through the SPI interface. A cable called *Onboard Installation Kit* is needed.

9.2 Using BlueTest over RS232

BlueTest is a piece of software, which can be used to perform several built-in radio tests, such as Bit Error Rate (BER) measurements, TX power measurements and RX measurements. BlueTest also uses the BCSP protocol to talk to the module and can be used in a similar way as PSTool. i

To use BlueTest:

1. Connect an RS2323 cable between the WTxx module and your PC
2. Power up the Bluetooth module
3. Open BlueTest
4. Use the default connection settings: **BCSP**, **COMn** and **115200**
5. Perform the necessary tests
6. Close BlueTest and reset WTxx module.

9.3 Switching to HCI Firmware

New iWRAP firmware builds use a so called *unified firmware (*iWRAP firmware 2.1.0 and newer*).* This means that the firmware contains all iWRAP firmware, RFCOMM and HCI stacks. Selecting the active part is done by using PS keys and there is no need to reflash the actual firmware as with older versions of iWRAP.

Switching can be done by using PSTool software.

1. Connect the WTxx Bluetooth module as instructed in chapter 9.1.

2. Change the following parameters to switch to HCI mode

   a. PSKEY_INITIAL_BOOTMODE

      i. 0000 = iWRAP

      ii. 0001 = HCI, BCSP, 115800,8n1

      iii. 0003 = HCI, USB

      iv. 0004 = HCI, H4, 115200,8n1

   b. PSKEY_UART_BAUDRATE (Suitable value if H4 or BCSP used)

   c. PSKEY_UART_CONFIG_H4
      PSKEY_UART_CONFIG_BCSP (Suitable key/value)

   d. PSKEY_USB_XXXX (If USB is used, configure the necessary keys)

Note:

- PSTool 1.21 or later is needed to change the parameters mentioned above.

9.4 Firmware Updates over SPI

The SPI interface is dedicated to firmware updates. The Onboard Installation Kit (SPI cable) and a Windows™ software called iWRAP update client (or BlueTest) can be used to update / restore the firmware.

iWRAP update client is an easier and the suggested way to do the firmware upgrade. instead of BlueFlash. iWRAP update client can recognize the hardware and software version of the module and reflash correct firmware and parameters into the module, and the user just needs to select the firmware version.

9.5 Firmware Updates over UART

The firmware can also be updated over the UART or RS232 interface. A method called Device Firmware Upgrade (DFU) is needed. Bluegiga has a DFU Wizard tool, which allows the updates to be made from a Windows™ based PC in a similar way as with iWRAP update client.

There is also a possibility the write the DFU support into a host processor connected to the WTxx Bluetooth module. In this way, the firmware can be updated even if the module cannot be accessed from a PC.

The DFU protocol is open and the description can be requested from support@bluegiga.com.

DFU file sizes:

• iWRAP update:	~20-30kB
• *Bluetooth* stack update:	~700kB
• Full update (max DFU size):	~1MB

Note:

- Please refer to *Firmware & Parameters User Guide* for more information about iWRAP update client and DFU Wizard.

9.6 UART Hardware Flow Control

Hardware flow control is enabled by default. It can be disabled by changing the value of PSKEY_UART_CONFIG_XXX (XXX = USR, H4, H5 or BCSP). With iWRAP, the PS key is PSKEY_UART_CONFIG_USR.

- If PSKEY_UART_CONFIG_USR is **08a8**, HW flow control is enabled
- If PSKEY_UART_CONFIG_USR is **08a0**, HW flow control is disabled

Hardware flow control can be disabled also with a proper hardware design. If the flow control is enabled from PS-keys, but no flow control is used, the following steps should be implemented in the hardware design:

- CTS pin must be grounded
- RTS pin must be left floating

WARNING:

- If hardware flow control is disabled and iWRAP buffers are filled (in command or data mode), the firmware will hang and needs a physical reset to survive. Therefore, hardware flow control should be used whenever possible to avoid this situation.
- However, if hardware flow control must be disabled, the host system should be designed in a way that it can recognize that the firmware has hung and is able to survive it.

9.7 RS232 Connections Diagram

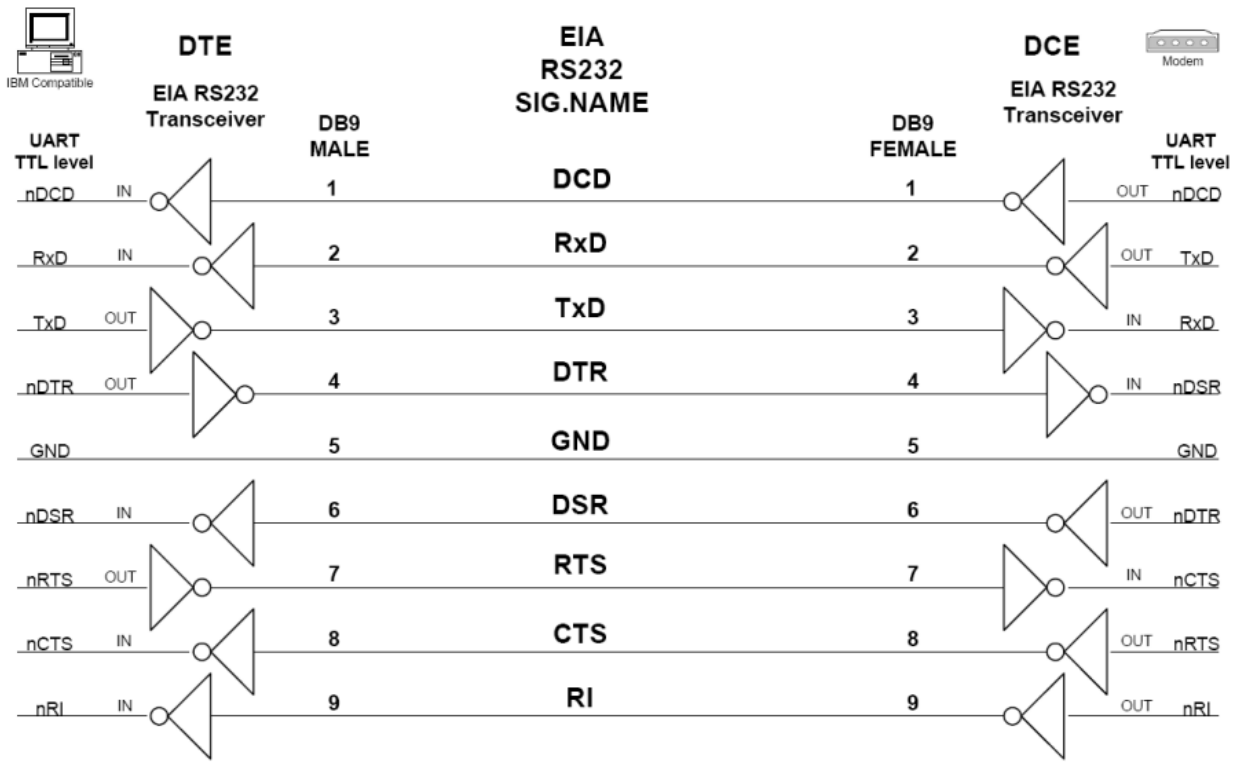


Figure 7: RS232 connections

9.8 Supported *Bluetooth* Profiles Overview

9.8.1 RFCOMM with TS07.10

The RFCOMM protocol emulates the serial cable line settings and status of an RS-232 serial port and is used for providing serial data transfer. RFCOMM connects to the lower layers of the *Bluetooth* protocol stack through the L2CAP layer.

By providing serial-port emulation, RFCOMM supports legacy serial-port applications while also supporting the OBEX protocol among others. RFCOMM is a subset of the ETSI TS 07.10 standard, along with some *Bluetooth* specific adaptations.

The RFCOMM protocol supports up to 60 simultaneous connections between two *Bluetooth* devices. The number of connections that can be used simultaneously in a *Bluetooth* device is implementation-specific.

For the purposes of RFCOMM, a complete communication path involves two applications running on different devices (the communication endpoints) with a communication segment between them. The figure above shows the complete communication path. (In this context, the term application may mean other things than end-user application; e.g. higher layer protocols or other services acting on behalf of end-user applications.)

RFCOMM is intended to cover applications that make use of the serial ports of the devices in which they reside. In the simple configuration, the communication segment is a Bluetooth link from one device to another (direct connect), see the figure to the left. Where the communication segment is another network, Bluetooth wireless technology is used for the path between the device and a network connection device like a modem. RFCOMM is only concerned with the connection between the devices in the direct connect case, or between the device and a modem in the network case.

RFCOMM can support other configurations, such as modules that communicate via Bluetooth wireless technology on one side and provide a wired interface on the other side, as shown in the figure below. These devices are not really modems but offer a similar service. They are therefore not explicitly discussed here.

Basically two device types exist that RFCOMM must accommodate. Type 1 devices are communication end points such as computers and printers. Type 2 devices are those that are part of the communication segment; e.g. modems. Though RFCOMM does not make a distinction between these two device types in the protocol, accommodating both types of devices impacts the RFCOMM protocol.

Source: Bluetooth SIG, URL:

http://www.bluetooth.com/Bluetooth/Technology/Works/RFCOMM.htm

9.8.2 Service Discovery Protocol (SDP)

SDAP describes how an application should use SDP to discover services on a remote device. It illustrates several approaches to managing the device discovery via Inquiry and Inquiry Scan and service discovery via SDP. The ideas contained in the SDAP specification augment the basic specifications provided in GAP, SDP, and the basic processes of device discovery. The use cases for SDAP are intended to encompass the majority of service discovery scenarios associated with all profiles and devices.

Source: Bluetooth SIG, URL:

http://www.bluetooth.com/Bluetooth/Technology/Works/SDAP.htm

9.8.3 Serial Port Profile (SPP)

A scenario would be using two devices, such as PCs or laptops, as virtual serial ports and then connecting the two devices via Bluetooth technology.

The SPP defines two roles, Device A and Device B.

- Device A – This is the device that takes initiative to form a connection to another device (initiator).

- Device B – This is the device that waits for another device to take initiative to connect (acceptor).

The applications on both sides are typically legacy applications, able and wanting to communicate over a serial cable (which in this case is emulated). But legacy applications cannot know about Bluetooth procedures for setting up emulated serial cables, which is why they need help from some sort of Bluetooth aware helper application on both sides. (These issues are not explicitly addressed in this profile; the major concern here is for Bluetooth interoperability.)

Source: Bluetooth SIG, URL:

http://www.bluetooth.com/Bluetooth/Technology/Works/SPP.htm

9.8.4 Hands-Free Profile (HFP)

HFP describes how a gateway device can be used to place and receive calls for a hand-free device.

The HFP defines two roles, that of an Audio Gateway (AG) and a Hands-Free unit (HF):

- Audio Gateway (AG) – This is the device that is the gateway of the audio, both for input and output, typically a mobile phone.

- Hands-Free Unit (HF) – This is the device acting as the Audio Gateway's remote audio input and output mechanism. It also provides some remote control means.

Hands-Free control is the entity responsible for Hands-Free unit specific control signaling; this signaling is AT command based.

Although not shown in the model to the left, it is assumed by this profile that Hands-Free Control has access to some lower layer procedures (for example, Synchronous Connection establishment).

The audio port emulation layer shown in the figure to the left is the entity emulating the audio port on the Audio Gateway, and the audio driver is the driver software in the Hands-Free unit.

For the shaded protocols/entities in the figure to the left, the Serial Port Profile is used as the base standard. For these protocols, all mandatory requirements stated in the Serial Port Profile apply except in those cases where this specification explicitly states deviations.

Source: Bluetooth SIG, URL:

http://www.bluetooth.com/Bluetooth/Technology/Works/HFP.htm

9.8.5 Dial-up Networking Profile (DUN)

DUN provides a standard to access the Internet and other dial-up services over *Bluetooth* technology. The most common scenario is accessing the Internet from a laptop by dialing up on a mobile phone wirelessly. It is based on SPP and provides for relatively easy conversion of existing products, through the many features that it has in common with the existing wired serial protocols for the same task. These include the AT command set specified in ETSI 07.07 and PPP.

Like other profiles built on top of SPP, the virtual serial link created by the lower layers of the *Bluetooth* protocol stack is transparent to applications using the DUN profile. Thus, the modem driver on the data-terminal device is unaware that it is communicating over *Bluetooth* technology. The application on the data-terminal device is similarly unaware that it is not connected to the gateway device by a cable.

DUN describes two roles, the gateway and terminal devices. The gateway device provides network access for the terminal device. A typical configuration consists of a mobile phone acting as the gateway device for a personal computer acting as the terminal role.

Source: Bluetooth SIG, URL:

http://www.bluetooth.com/Bluetooth/Technology/Works/DUN.htm

9.8.6 Object Push Profile (OPP)

OPP defines the roles of push server and push client. These roles are analogous to and must interoperate with the server and client device roles that GOEP defines. It is called push because the transfers are always instigated by the sender (client), not the receiver (server). OPP focuses on a narrow range of object formats to maximize interoperability. The most common acceptable format is the vCard. OPP may also be used for sending objects such as pictures or appointment details.

Source: Bluetooth SIG, URL:

http://www.bluetooth.com/Bluetooth/Technology/Works/OPP.htm

9.8.7 Advanced Audio Distribution Profile

A2DP describes how stereo-quality audio can be streamed from a media source to a sink.

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

- Source (SRC) – A device is the SRC when it acts as a source of a digital audio stream that is delivered to the SNK of the Piconet.
- Sink (SNK) – A device is the SNK when it acts as a sink of a digital audio stream delivered from the SRC on the same Piconet.

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

This profile relies on GAVDP. It includes mandatory support for low complexity subband codec (SBC) and supports optionally MPEG-1,2 Audio, MPEG-2,4 AAC and ATRAC.

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

Source: Bluetooth SIG, URL:

http://www.bluetooth.com/Bluetooth/Technology/Works/A2DP.htm

9.8.8 Audio Video Remote Control Profile

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

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

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

This protocol specifies the scope of the AV/C Digital Interface Command Set (AV/C command set, defined by the 1394 trade association) to be applied, realizing simple implementation and easy operability. This protocol adopts the AV/C device model and command format for control messages and those messages are transported by the Audio/Video Control Transport Protocol (AVCTP).

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

Source: Bluetooth SIG, URL:

http://www.bluetooth.com/Bluetooth/Technology/Works/AVRCP.htm

9.8.9 Human Interface Device Profile

The HID profile defines the protocols, procedures and features to be used by Bluetooth HID such as keyboards, pointing devices, gaming devices and remote monitoring devices.

The HID defines two roles, that of a Human Interface Device (HID) and a Host:

- Human Interface Device (HID) – The device providing the service of human data input and output to and from the host.
- Host – The device using or requesting the services of a Human Interface Device.

The HID profile uses the universal serial bus (USB) definition of a HID device in order to leverage the existing class drivers for USB HID devices. The HID profile describes how to use the USB HID protocol to discover a HID class device's feature set and how a Bluetooth enabled device can support HID services using the L2CAP layer. The HID profile is designed to enable initialization and control self-describing devices as well as provide a low latency link with low power requirements.

The Bluetooth HID profile is built upon the Generic Access Profile (GAP), specified in the Bluetooth Profiles Document; see Referenced Documents. In order to provide the simplest possible implementation, the HID protocol runs natively on L2CAP and does not reuse Bluetooth protocols other than the Service Discovery Protocol.

Source: Bluetooth SIG, URL:

http://www.bluetooth.com/Bluetooth/Technology/Works/HID.htm

9.9 Power Saving

SNIFF mode:

Once a *Bluetooth* device is connected to a Piconet, it can enter one of three power saving modes. In SNIFF mode, the activity of a *Bluetooth* device is lowered, enabling it to listen at a reduced rate to the Piconet. The interval or period between SNIFF is configurable.

The SNIFF mode is the least power efficient of all three power saving modes.

Deep sleep:

iWRAP supports a also a power saving mode called deep sleep. In deep sleep state the main processor is put into reduced operation mode and this radically affects the current consumption of WTxx modules. Deep sleep can be used in generally in two operational modes:

- iWRAP is in "idle" state i.e. no Bluetooth connections are active.

- iWRAP has active Bluetooth connections, which are in high sniff power saving modes.

Deep sleep can reduce the current consumption to as low as 0.01mA.

9.10 *Bluetooth* profile UUIDs

UUID:	***Bluetooth* Profile:**
0001	SDP
0003	RFCOMM
0008	OBEX
000C	HTTP
000F	BNEP
0100	L2CAP
1000	Service Discovery Server Service ClassID
1001	Browse Group Descriptor Service ClassID
1002	Public Browse Group
1101	Serial Port Profile
1102	LAN Access Using PPP
1103	Dial up Networking
1104	IrMC Sync
1105	OBEX Object Push Profile
1106	OBEX File Transfer Profile
1107	IrMC Sync Command
1108	Headset
1109	Cordless Telephony
110A	Audio Source
110B	Audio Sink
110C	A/V_Remote Control Target

110D	Advanced Audio Distribution Profile (A2DP)
110E	A/V_Remote Control
110F	Video Conferencing
1110	Intercom
1111	Fax
1112	Headset Audio Gateway
1113	WAP
1114	WAP_CLIENT
1115	Personal Area Networking User
1115	PANU
1116	Network Access Point
1116	NAP
1117	Group Network
1117	GN
1118	Direct Printing
1119	Reference Printing
111A	Imaging
111B	Imaging Responder
111C	Imaging Automatic Archive
111D	Imaging Referenced Objects
111E	Hands-Free
111F	Hands-Free Audio Gateway
1120	Direct Printing Reference Objects Service

1121	ReflectedUI
1122	Basic Printing
1123	Printing Status
1124	Human Interface Device Service
1125	Hardcopy Cable Replacement
1126	HCR_Print
1127	HCR_Scan
1128	Common_ISDN_Access
1129	Video Conferencing GW
112A	UDI_MT
112B	UDI_TA
112C	Audio/Video
112D	SIM_Access
112E	Phonebook Access - PCE
112F	Phonebook Access - PSE
1130	Phonebook Access
1200	PnP Information
1201	Generic Networking
1202	Generic File Transfer
1203	Generic Audio
1204	Generic Telephony
1205	UPNP_Service
1206	UPNP_IP_Service

1300	ESDP_UPNP_IP_PAN
1301	ESDP_UPNP_IP_LAP
1302	ESDP_UPNP_L2CAP
1303	Video Source
1304	Video Sink
1305	Video Distribution

Table 13: UUIDs and Profiles

For more information, please go to:

https://programs.*Bluetooth*.org/apps/content/?doc_id=49709

10. KNOWN ISSUES

Issue		Explanation
	Using multiple DLCs can crash iWRAP	Opening several connections to iWRAP using the same channel may crash the firmware. UUID should be used instead of a channel. This is a bug in the CSR firmware.
	Listing remote SDP record may run out of memory	When a service discovery is made by using the SDP command and if *root* mode is used and remote device supports many services, iWRAP may run out of memory and reset. To overcome this, only a specific service should be searched for instead of using root mode.
	Do not force sniff	If sniff is enabled by using the 'SET BT SNIFF' command, iWRAP cannot unsniff if remote end requests for it.
	Frame mode flow control hangs	In multiplexing mode, the firmware will hang if data length is longer than 100 bytes. A physical reset is needed. This is a bug in the CSR firmware.
	Inquiry RSSI and clock caching	If RSSI in the inquiry and clock offset caching are enabled, connections can not be opened. This is a bug in the CSR firmware.
	HW flow control	If HW flow control is not used and iWRAP buffers are filled either in data or command mode, the firmware will hang and needs a physical reset. This is a bug in the CSR firmware.
	Simultaneous connection between two iWRAPs	Two simultaneous ACL connections can not be opened between two iWRAPs.
	SET CONTROL INIT RESET	Issuing SET CONTROL INIT RESET will result in an infinite reset loop. PSKEY_USR_27 must be deleted to survive this condition.
238	SELECT [link_id] with MUX mode	SELECT [link_id] can be issued in MUX mode and it puts iWRAP into normal data mode.
387	Interactive pairing	Pin code should be enabled even if interactive pairing mode is used. If pin code is disabled with "SET BT PAIR *", *Bluetooth* security mode 0 is used and interactive pairing does not produce "AUTH" vent.
337	Link IDs get mixed	If an existing *Bluetooth* connection is lost while opening a new connection and exactly between "CALL [link_id]" and "CONNECT [link_id]" events, the Link IDs the mixed. Example: CALL 00:00:00:00:00:00 1101 rfcomm CALL 1 NO CARRIER 0 ERROR 0 CONNECT 0 RFCOMM 1

459	Outgoing connections	Outgoing HFP, HID, A2DP etc. connections can be made even if the profile is not locally enabled.
379	Data received before CD signal goes high	There is a 30ms delay between RING event and CD signal. In some rare cases data can be received from the *Bluetooth* connection is this 30ms slot.

Table 14: Known issues in iWRAP

11. TROUBLESHOOTING

Q: I get no response from iWRAP?

Make sure your terminal settings are correct. Use ***PSTool*** to check the UART settings from the WRAP THOR *Bluetooth* module and make similar settings to your terminal software.

Check also your ECHO MODE settings. If you have set ECHO MODE to 0, you should not be able to see any responses.

You can also use iWRAP update to restore the firmware and default settings.

Q: I changed 'UART Baud rate' key, but it didn't seem to work?

UART baud rate is stored now into user keys instead of '***UART baud rate'*** key. Delete '***User configuration data 26***' in order to return back to default settings ***115200,8n1***.

Notice also that if you change the baud rate with "SET CONTROL BAUD", it does not affect the baud rate you need to use with PSTool, if you want to access parameters. This baud rate is defined by the ***'UART baud rate'*** key.

AutoBCSP requires that iWRAP baud rate is same as '***UART baud rate***' key.

Q: Data coming from the UART is corrupted

If you are using 'Deep sleep' the minimum baud rate that can be used is 19200. Lower baud rates will corrupt the data.

Q: I'm missing characters when I type ASCII commands.

If deep sleep is used, the first character written to UART wakes the module from the 'Deep sleep' and that's why the character is lost. There are two ways to overcome this problem:

1. If you command iWRAP with a micro controller or processor, add 'space' or 'line break' characters in front of every command.

2. In PSTool, set parameter 'EXIT deep sleep on CTS line activity' to TRUE. Now 'Deep sleep' does not lose characters any more, but current consumption will increase.

12. IWRAP USAGE EXAMPLES

This section contains various iWRAP configuration and usage examples.

12.1 Serial Port Profile – Slave mode

This example shows how to set up iWRAP into a simple Serial Port Profile slave mode. In this mode iWRAP accepts Bluetooth connections and transmits data. The basic configuration steps are displayed in the figure below:

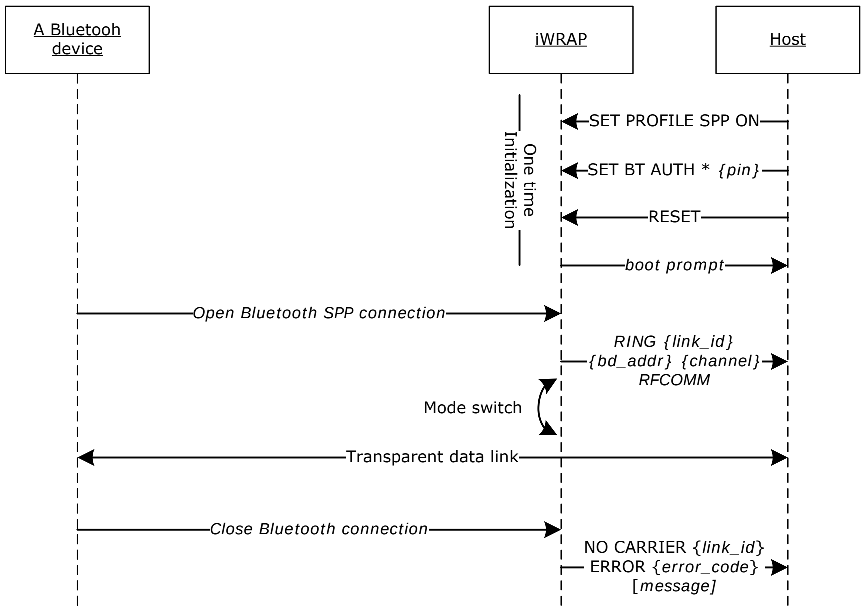

A Bluetooh device
iWRAP
Host
One time Initialization
SET PROFILE SPP ON
SET BT AUTH * {pin}
RESET
boot prompt
Open Bluetooth SPP connection
RING {link_id} {bd_addr} {channel} RFCOMM
Mode switch
Transparent data link
Close Bluetooth connection
NO CARRIER {link_id} ERROR {error_code} [message]

Figure 8: SPP slave example

In the initialization section, Bluetooth Serial Port Profile is enabled, Bluetooth pin code set and finally reset is needed to active the profile change.

The example above illustrates the simplest setup, which could be tweaked with a lot of different iWRAP settings. Some of most common settings are discussed below:

- **SET BT PAGEMODE 3 2000 1**

  With this setting, iWRAP is configured to be visible in the inquiry and to be connectable as a slave device should be.

  On the other hand, in some cases the slave device does not need to be visible in the inquiry i.e. In this case, our setting would be: **SET BT PAGEMODE 2 2000 1**. If iWRAP is not visible in the inquiry, the current consumption will be 1-2mA lower.

- **SET BT ROLE 0 f 7d00**

  This setting configures the master / slave mode in iWRAP. With the default values iWRAP will stay as a slave. In some cases it is needed or is at least useful to perform a master-salve switch when connection is received. The two main reasons for this are:

1. If sniff mode is not used master uses around three times lass current then slave

2. If iWRAP needs to host more then 2 connections (3 or 4) it needs to be master for all the connections i.e. master slave switch is needed when connections are received.

- **SET CONTROL ECHO {*mode*}**

  This setting allows controlling the amount of feedback iWRAP sends to the host. In the default setting all possible information like events is sent to the host over UART interface. If iWRAP needs to be controlled totally transparent i.e. only the actual data should be sent to the host ***mode*** should be set to 0.

- **SET CONTROL CD**

  Carrier detect signal can be used to indicate active Bluetooth connection with one of the GPIO pins. This can be used instead of **RING** event to monitor if Bluetooth connections exist.

- **SET CONTROL ESCAPE**

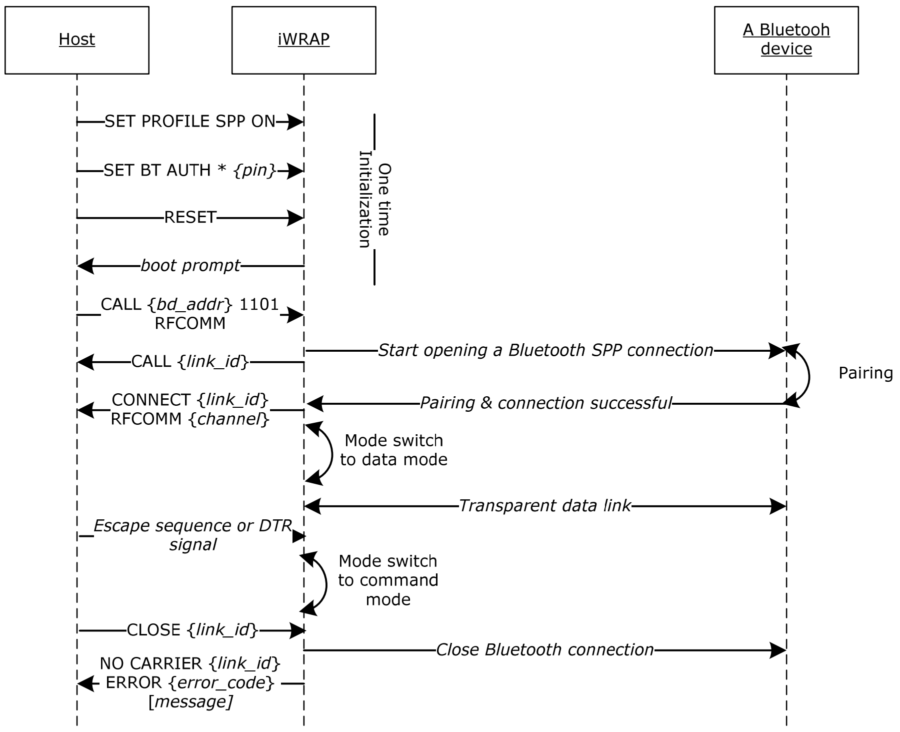
  Escape setting can be used to define the escape character or GPIO pin to make command-data mode switches or alternatively to close active Bluetooth connections.

- **SET BT SNIFF**

  This setting enables automatic sniff mode i.e. all active connections will be out to power saving mode automatically.

12.2 Serial Port Profile – Master mode

This example shows how to set up iWRAP into a simple Serial Port Profile master mode. In this mode iWRAP opens Bluetooth connection(s) and transmits data. The basic configuration steps are displayed in the figure below:



Host
iWRAP
A Bluetooh device
SET PROFILE SPP ON
SET BT AUTH * {pin}
RESET
boot prompt
One time Initialization
CALL {bd_addr} 1101 RFCOMM
Start opening a Bluetooth SPP connection
CALL {link_id}
Pairing
CONNECT {link_id} RFCOMM {channel}
Pairing & connection successful
Mode switch to data mode
Transparent data link
Escape sequence or DTR signal
Mode switch to command mode
CLOSE {link_id}
Close Bluetooth connection
NO CARRIER {link_id} ERROR {error_code} [message]

Figure 9: SPP master example

In the initialization section, Bluetooth Serial Port Profile is enabled, Bluetooth pin code set and finally reset is needed to active the profile change.

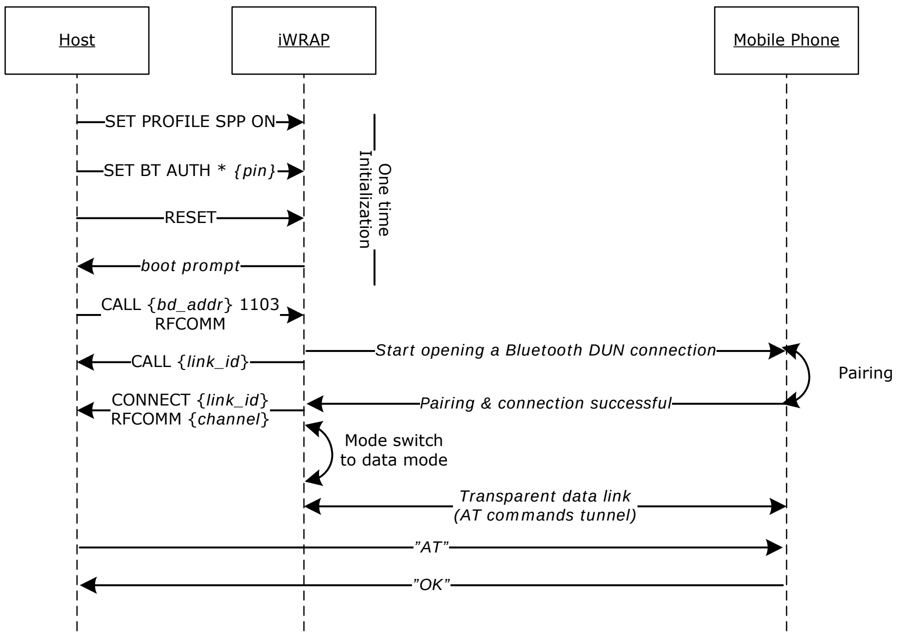
Again lot of iWRAP settings exists to modify the basic master functionality. The same settings apply for master mode as for salve mode and they are not discussed again. However some extra configurations might be useful:

- **SET CONTROL AUTOCALL**

Autocall enables so called "cable replacer" functionality in iWRAP. In this mode iWRAP monitors the status of a Bluetooth connection and if it is lost iWRAP tries to reestablish it automatically. Autocall configuration allows one to configure the type of connection one tries to open (SPP, DUN, A2DP, etc.) and the timeout between connection attempts.

12.3 Dial-up Networking

The Dial-Up Networking (DUN) profile allows you for example to connect to phone phones and control their GSM modem with AT commands. The most common use cases for DUN are sending SMS messages or connecting to Internet via GPRS or 3G. The simple below shows how to open a Dial-Up Networking connection to a phone and how to send an AT command to the phone.



Host
iWRAP
Mobile Phone
SET PROFILE SPP ON
SET BT AUTH * {pin}
RESET
boot prompt
One time Initialization
CALL {bd_addr} 1103 RFCOMM
Start opening a Bluetooth DUN connection
CALL {link_id}
Pairing
CONNECT {link_id} RFCOMM {channel}
Pairing & connection successful
Mode switch to data mode
Transparent data link (AT commands tunnel)
"AT"
"OK"

Figure 10: How to open a DUN connection to a mobile phone

In iWRAP the *Bluetooth* code must be set, since most of the mobile phones always require the PIN code authentication, before allowing the Dial-Up Networking connection.

It may be wise to do the pairing from the mobile phone and make the iWRAP module 'trusted'. Once this is done, the phone does not ask for the PIN code every time the connection is opened.

Notice that not all the mobile phones support the same AT commands, since some of the commands are optional and some mandatory.

Refer to the following AT command specifications for more information and examples: 3GPP TS 27.005 and 3GPP TS 07.07.

12.4 OBEX Object Push Profile Server

This example shows how to set up a simple Object Push Profile (OPP) server for receiving files over *Bluetooth*.

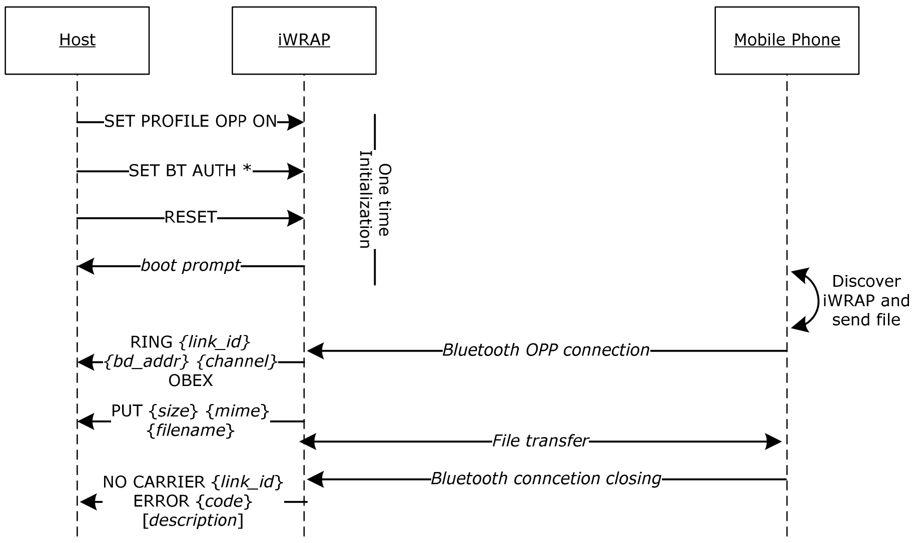

Host
iWRAP
Mobile Phone
SET PROFILE OPP ON
SET BT AUTH *
RESET
boot prompt
One time Initialization
Discover iWRAP and send file
RING {link_id} {bd_addr} {channel} OBEX
Bluetooth OPP connection
PUT {size} {mime} {filename}
File transfer
NO CARRIER {link_id} ERROR {code} [description]
Bluetooth conncetion closing

Figure 11: Receiving files through OPP

- **SET PROFILE OPP on**

  This command enables OBEX OPP profile in iWRAP needed for receiving files. In the example, the PIN code is disabled so that the phone does not prompt for the PIN when sending the file.

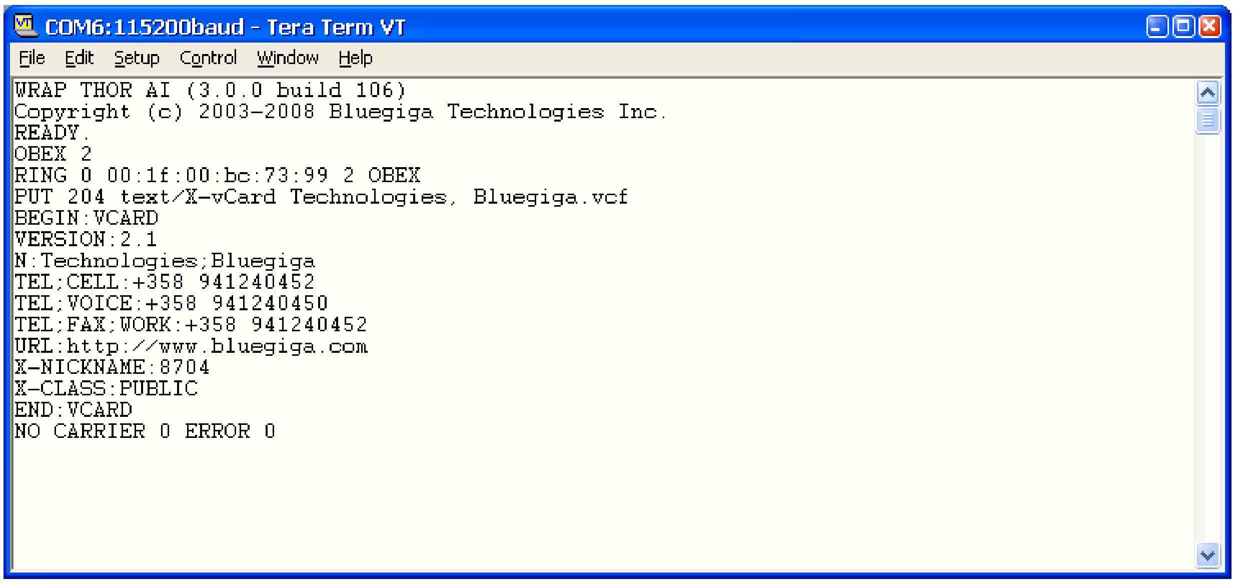
When file reception starts iWRAP will produce an event:

Synopsis:
PUT {*size*} {*mime*} {*filename*}

Description:	
size	Size of file in bytes
mime	Type of file
filename	Name of file

Some devices also require that that Class of Device (CoD) is configured correctly before they are able to send files via OBEX. A correct class of device setting can be found from the *Bluetooth* specification.

Example trace of file reception via OPP:


COM6:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
WRAP THOR AI (3.0.0 build 106)
Copyright (c) 2003-2008 Bluegiga Technologies Inc.
READY.
OBEX 2
RING 0 00:1f:00:bc:73:99 2 OBEX
PUT 204 text/X-vCard Technologies, Bluegiga.vcf
BEGIN:VCARD
VERSION:2.1
N:Technologies;Bluegiga
TEL;CELL:+358 941240452
TEL;VOICE:+358 941240450
TEL;FAX;WORK:+358 941240452
URL:http://www.bluegiga.com
X-NICKNAME:8704
X-CLASS:PUBLIC
END:VCARD
NO CARRIER 0 ERROR 0
```

Figure 12: Receiving a vCard over OPP

12.5 Hands-Free Audio Gateway Connection to a Headset Device

This example shows how the Hands-Free audio gateway mode in iWRAP can be used to set up a Hands-Free connection to a headset device.

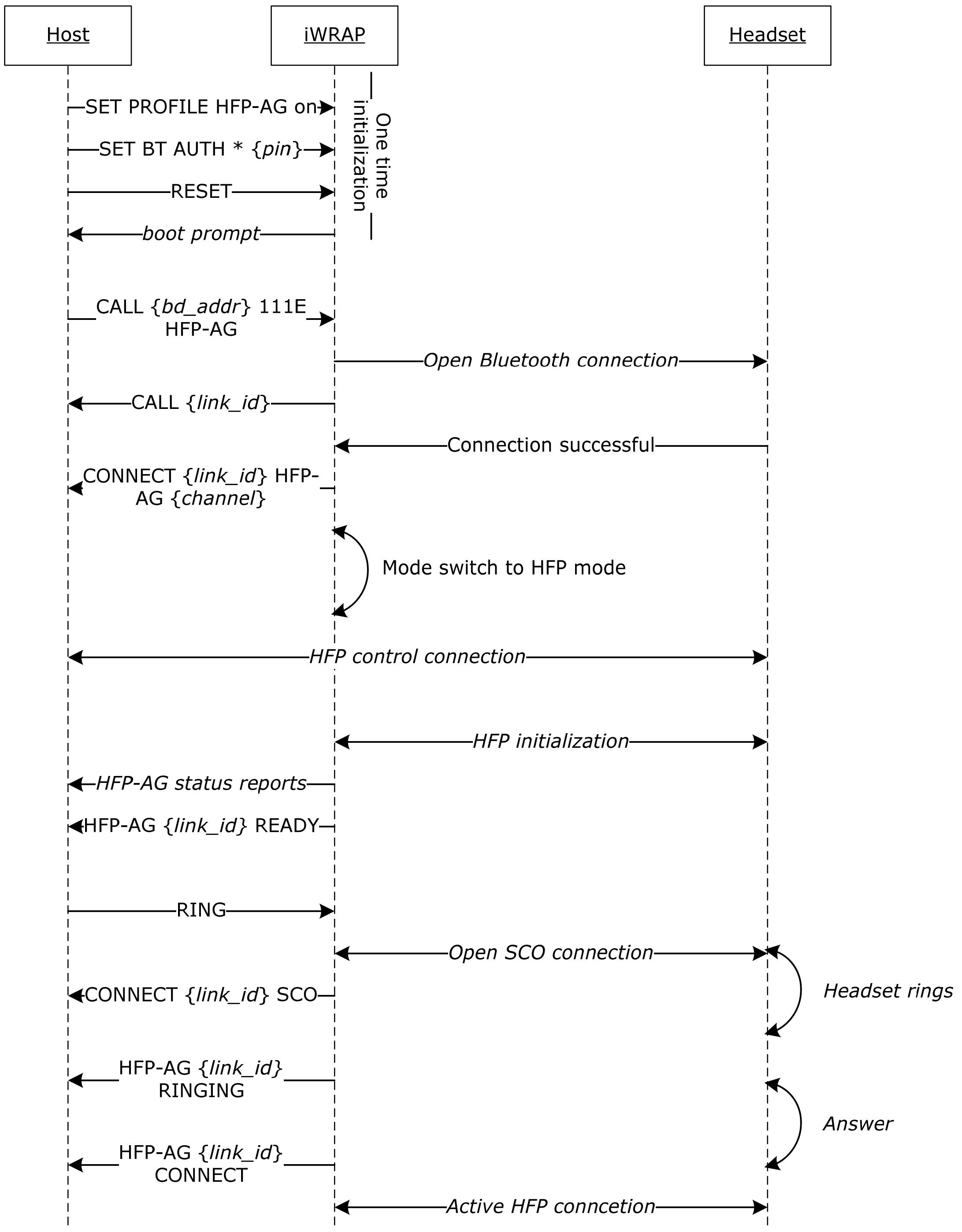


Figure 13: iWRAP to headset Hands-Free connection

Hands-Free profile consists of two connections: a control connection (RFCOMM) an audio connection (SCO). **CALL {*bd_addr*} 111E HFP-AG** opens a HFP control connection, which allows HFP messaging between iWRAP and the headset. The possible HFP messages are:

Command	**Function**	**AT command mapping**
ANSWER	Answer to call	callsetup 0, call 1
DISCONNECT	Hang-up call	callsetup 0, call 0
ERROR	Send ERROR result to Hands Free	error
HANGUP	Hang-up call	callsetup 0, call 0
REJECT	Reject call	callsetup 0, call 0
OK	Send OK result to Hands Free	ok
RING [*count*] [*number*]	Notify Hands Free for incoming call. Optional parameter [***count***] indicates the amount of ring indications. Optional parameter [***number***] displays the number from where the call is coming from	callsetup 1, ring, AT+CLIP ring is sent [***count***] times, if [***number***] is given, AT+CLIP is sent
STATUS {*status*}	Set status and send it to Hands Free. ***status*** 0 means no GSM connectivity and ***status*** 1 means active GSM connection.	AT+CIND
{raw AT command}	Sends the raw AT command to the headset	-

Table 15: Supported HFP-AG commands

Once the **RING** command is sent and answer is pressed on the headset also the SCO connection will be opened to transmit audio.

The possible HFP-AG status reports mentioned in the example are described below:

Event	Explanation
HFP-AG *{link_id}* READY	Service Level Connection open, HFP ready ***link_id*** Numeric connection identifier
HFP-AG *{link_id}* VOLUME *{level}*	Volume level information ***link_id*** Numeric connection identifier ***level*** Volume level information
HFP-AG *{link_id}* MIC *{level}*	Headset microphone gain information ***link_id*** Numeric connection identifier ***level*** Volume level information
HFP-AG *{link_id}* RINGING	Incoming call / headset ringing ***link_id*** Numeric connection identifier
HFP-AG *{link_id}* CONNECT	HFP connection / call active ***link_id*** Numeric connection identifier
HFP-AG *{link_id}* NO CARRIER	Call ended ***link_id*** Numeric connection identifier
HFP-AG *{link_id}* UNKNOWN *{data}*	Unrecognized AT command ***link_id*** Numeric connection identifier ***data*** Raw input data

Table 16: HFP-AG status messages

- ***"%s %d CALLING\r\n"***

  Outgoing call

- ***"%s %d BUSY\r\n"***

The supported AT-commands for hands Free profile can be found from the specification document: https://www.*Bluetooth*.org/foundry/adopters/document/HFP_1.5_SPEC_V10 or requested from support@bluegiga.com.

12.6 Hands-Free connection to a Mobile Phone

iWRAP can act as a hands-free device and send audio to a mobile phone. The example below reveals how that is done.

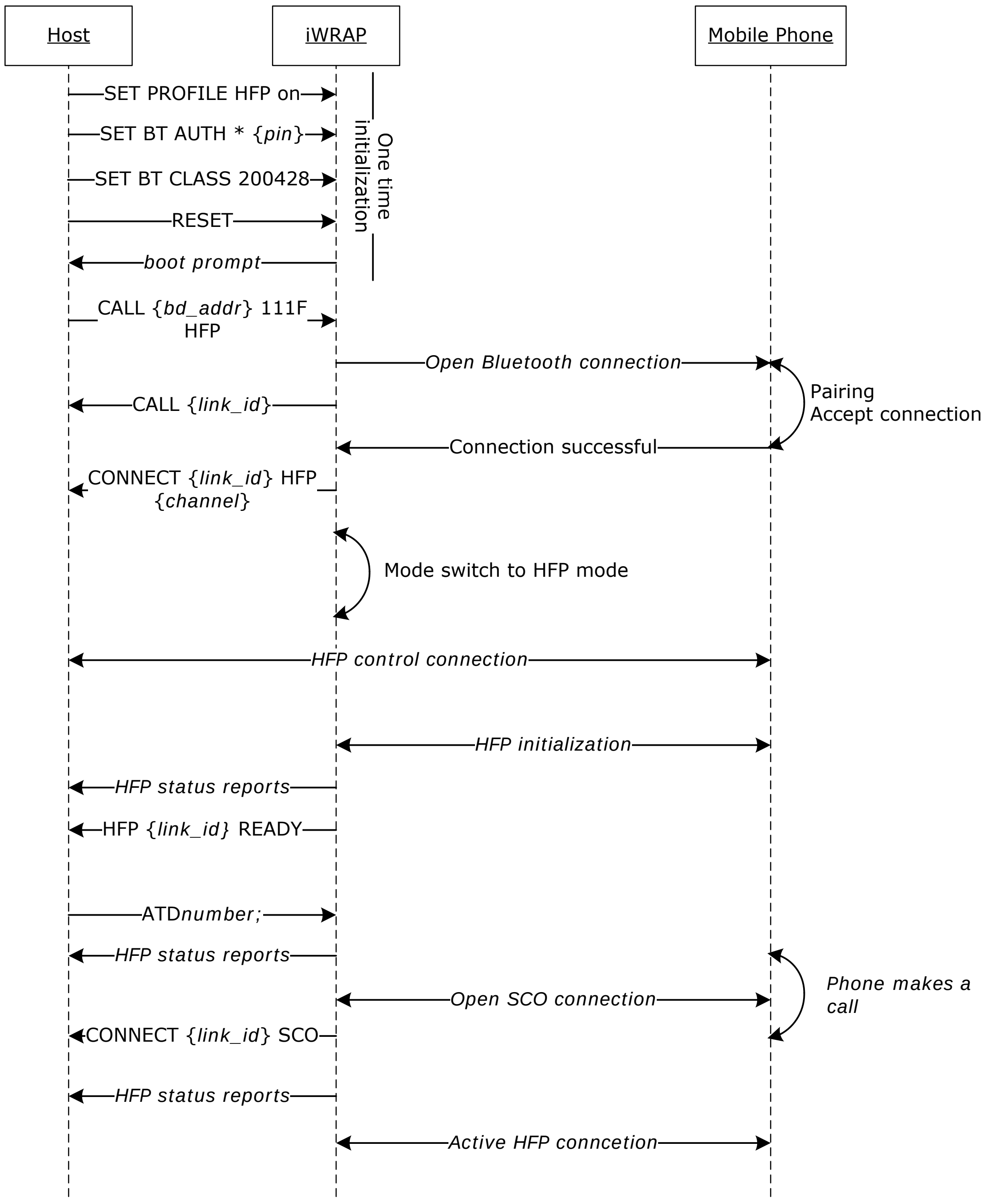


Figure 14: HFP connection to a mobile phone

HFP connection works in a similar fashion as HFP-AG connection. First the control connection (RFCOMM) is set up and when the GSM call is made then also the audio link (SCO) is activated. As in HFP-AG mode also if HFP mode several commands are supported and also several HFP events are displayed. Both commands and events are described in the two tables below.

Command	**Function**	**AT command mapping**
ANSWER	Answer to call	callsetup 0, call 1
DISCONNECT	Hang-up call	callsetup 0, call 0
HANGUP	Hang-up call	callsetup 0, call 0
REJECT	Reject call	callsetup 0, call 0
DMTF {*code*}	Set status and send it to Hands Free. ***status*** 0 means no GSM connectivity and ***status*** 1 means active GSM connection.	AT+CIND
{raw AT command}	Sends the raw AT command to the headset	-

Table 17: Supported HFP commands

The possible HFP status reports mentioned in the example are described below:

Event	**Explanation**
HFP {*link_id*} STATUS "{*feature*}" {*status*}	Status of ***feature*** ***feature*** HFP-AG feature: **service** = Network status **call** = call status **call_setup** = call setup **call_held** = call hold status **signal** = signal status **roam** = roaming status **batt_chg** = battery status ***status*** Status identifier
HFP {*link_id*} READY	Service Level Connection open, HFP ready ***link_id*** Numeric connection identifier

HFP *{link_id}* NETWORK "*{name}*"	Network operator name ***link_id*** Numeric connection identifier ***name*** Network operator name string
HFP *{link_id}* OK	OK response from HFP-AG ***link_id*** Numeric connection identifier
HFP *{link_id}* RING	Incoming call ***link_id*** Numeric connection identifier
HFP *{link_id}* CALLERID "*{number}*" "" *{entry}*	Caller ID ***link_id*** Numeric connection identifier ***number*** Phone number ***entry*** Phone book entry
HFP *{link_id}* UNKNOWN *{data}*	Unrecognized AT command ***link_id*** Numeric connection identifier ***data*** Raw input data

Table 18: HFP-AG status messages

The supported AT-commands for hands Free profile can be found from the specification document: https://www.*Bluetooth*.org/foundry/adopters/document/HFP_1.5_SPEC_V10 or requested from support@bluegiga.com.

12.7 iWRAP to iWRAP Serial Port Profile + SCO connection

iWRAP also supports SCO (audio) connections. This example shows how to open a simultaneous Serial Port Profile and SCO audio connection between two iWRAP modules.

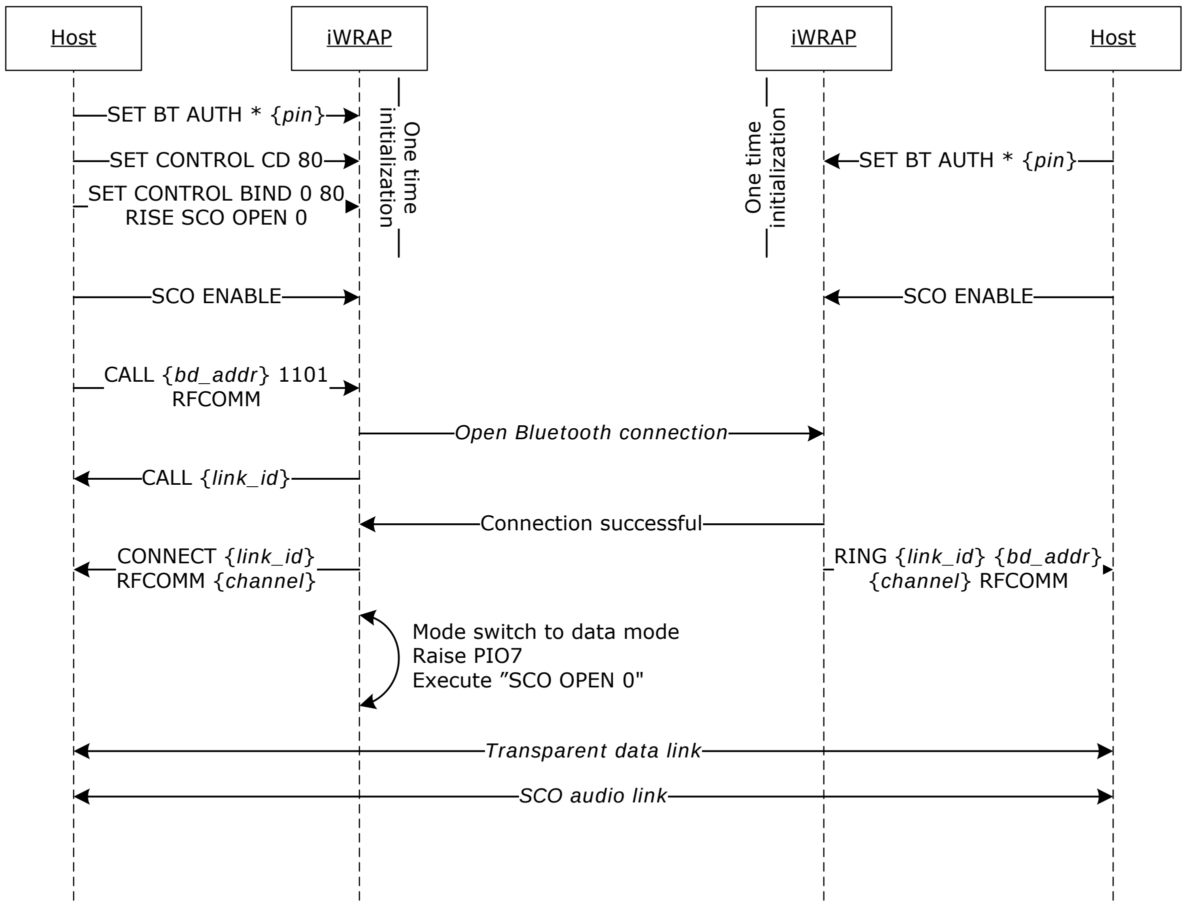


Figure 15: Simultaneous SPP and SCO connection between two iWRAP modules

- **SET CONTROL CD 80**

This configuration enables Carrier Detect (CD) signal to PIO7. The purpose is to raise PIO7 when the Serial Port Profile connection becomes active.

- **SCO ENABLE**

This command is needed to generally enable SCO connections. Alternatively one time configuration **SET CONTROL INIT SCO ENABLE** could be use, which causes iWRAP to execute **SCO ENABLE** in every restart. If HFP profile is enabled **SCO ENABLE** is not needed

- **SET CONTROL BIND 0 80 RISE SCO OPEN 0**

This setting binds **SCO OPEN 0** command to PIO7. The command gets priority 0 and is executed on the rising edge. So when PIO7 is raised by **SET CONTROL CD** command iWRAP executed **SCO OPEN 0** and opens and activates SCO connection between the iWRAP modules. This setting assumes (0 used as link_id) that the Serial Port Profile connection is the first and only connection iWRAP has. Alternatively the command could be given manually to iWRAP using syntax **SCO OPEN {link_id}**.

12.8 Wireless IO Replacement

iWRAPs can be used to do wireless IO replacement, that is, to transmit the status of GPIO PINs over the SPP link. This means that if the status of the local IO changes, so does the status of the remote IO. This functionality can be accomplished by using the MSC (Modem Status Control) feature in iWRAP.

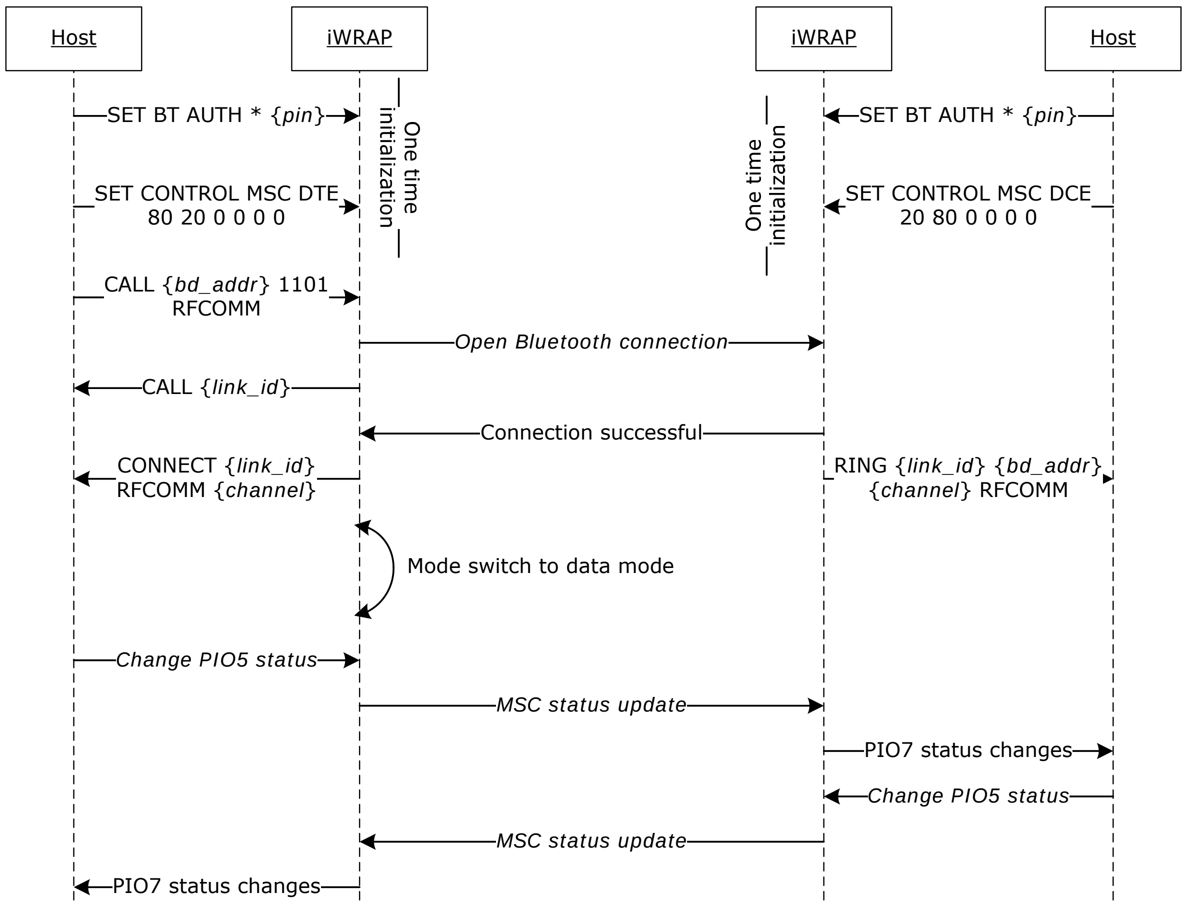


Figure 16: Wireless IO replacement connection

The example above was done with WT12 evaluation kits. In the evaluation kit, there is a DSR button in PIO5 and a LED in PIO7. Parameter 80 matches with PIO7 and parameter 20 with PIO5. So whenever DSR button is pressed in the local device, the LED status changes in the remote end.

NOTE:

- Switching the IO status very rapidly may reset iWRAP as the GPIO interrupts are handled with low priority. Therefore MSC feature is not feasible for radio GPIO sampling application.
- There is also a delay when transmitting the MSC status over the *Bluetooth* link. Without power saving in use, this delay is roughly 20ms and if power saving is in use, the delay depends on SNIFF mode parameters.

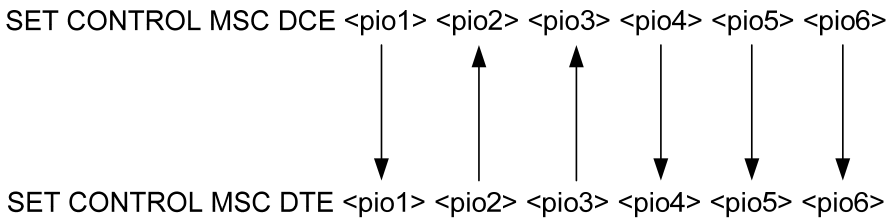

SET CONTROL MSC DCE <pio1> <pio2> <pio3> <pio4> <pio5> <pio6>
SET CONTROL MSC DTE <pio1> <pio2> <pio3> <pio4> <pio5> <pio6>

Figure 17: MSC signal directions

12.9 A2DP Sink

This example shows how to set up iWRAP into A2DP source mode i.e. to audio transmitter. In this mode iWRAP can open Bluetooth A2DP connections to A2DP sink(s) or receive connections from them. The basic configuration steps are displayed in the figure below:

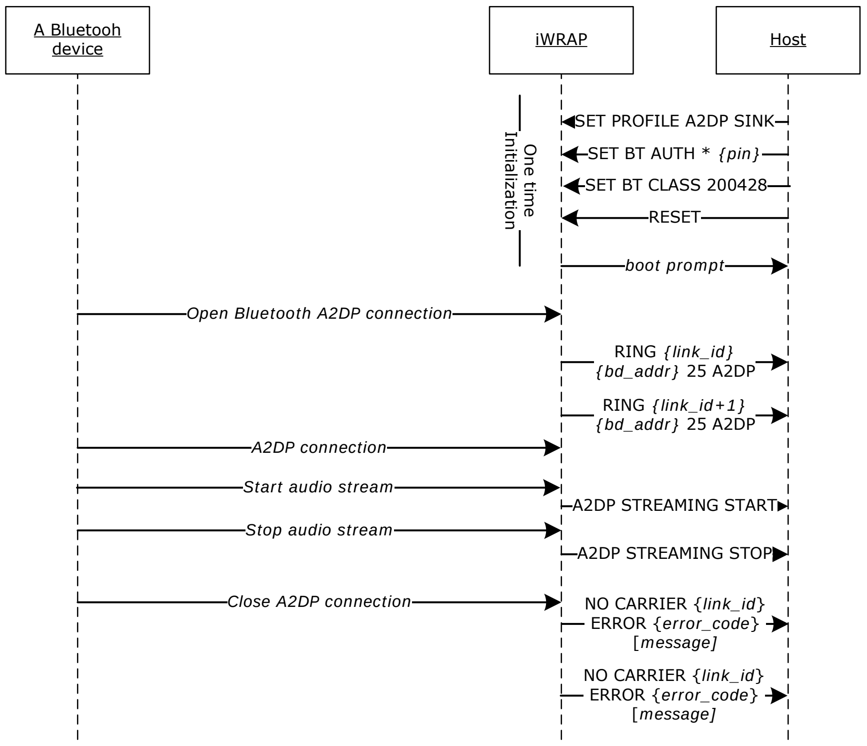


Figure 18: A2DP sink example

A2DP initialization is done in a similar fashion as with other profiles. However for mobile phones and PCs the Class of Device needs to be set to 200428 or 240428 otherwise they might not be able to open A2DP connection.

When audio streaming is enabled **A2SP STRAMING START** event will be displayed and when streaming is stopped **A2SP STRAMING STOP** is displayed.

12.10 A2DP Source

This example shows how to set up iWRAP into A2DP sink mode i.e. to receive audio. In this mode iWRAP can accept Bluetooth A2DP connections or connect an A2DP source. The basic configuration steps are displayed in the figure below:

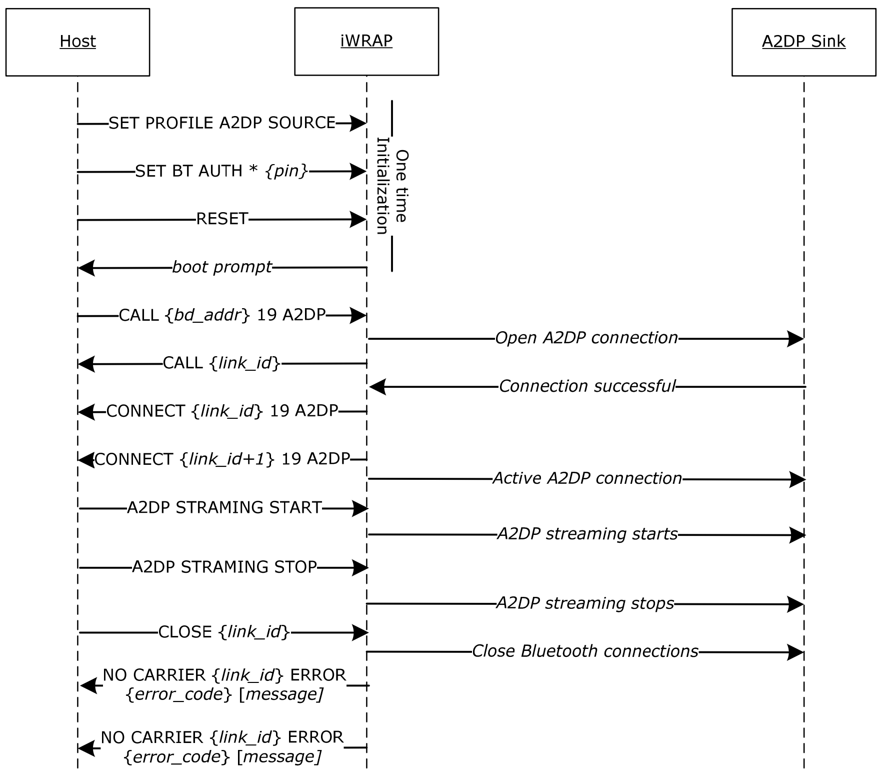


Figure 19: A2DP source example

When audio streaming needs to be enabled **A2SP STRAMING START** command needs to be given to iWRAP and on the other hand when streaming needs to be stopped **A2DP STRAMING STOP** command should be issued.

12.11 AVRCP Connection

This example shows how to set up an AVRCP connection from iWRAP to AVRCP target like a mobile phone. When AVRCP connection is set up remote control commands like play, pause etc. can be send from iWRAP to the remote device. The basic configuration steps are displayed in the figure below:

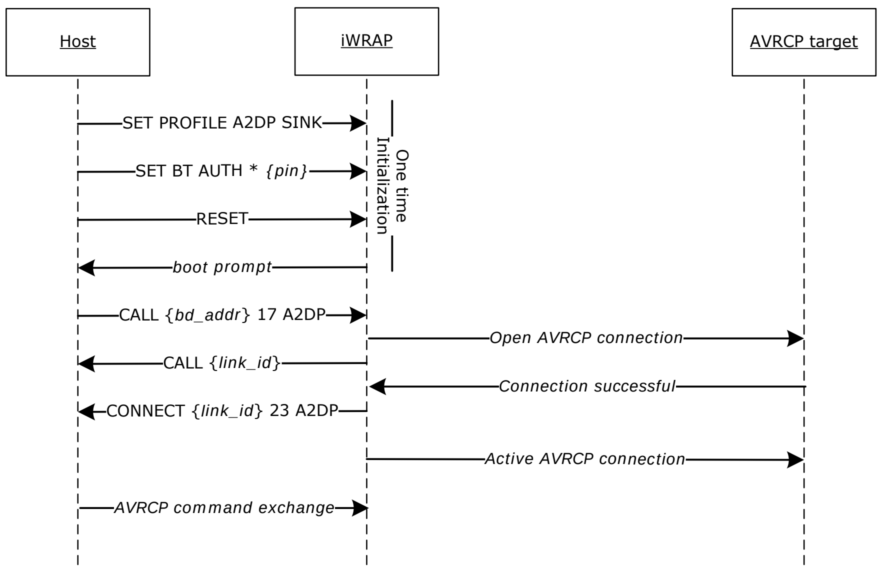


Figure 20: AVRCP example

The available AVRCP commands are described in the table below.

Command	Description
AVRCP UP	Volume up
AVRCP DN	Volume down
AVRCP MUTE	Mute
AVRCP STOP	Stop
AVRCP PLAY	Play
AVRCP PAUSE	Pause
AVRCP REWIND	Rewind
AVRCP FAST_FORWARD	Fast Forward
AVRCP FORWARD	Forward (next song)
AVRCP BACKWARD	Backward (previous song)
AVRCP {*raw*}	AVRCP command in raw hex mode. **Available codes:** 00-13, 20-2c, 30-37, 40-4B, 50-51, 71-75 and 7e

Table 19: Available AVRCP commands

12.12 Human Interface Device (HID) Example

This HID keyboard and mouse modes allows iWRAP to be used as a wireless *Bluetooth* keyboard or *Bluetooth* mousre. iWRAP can be connected to a PC, PDA or a mobile phone that supports HID host profile and then iWRAP can be used to simulate a keyboard or a mouse. An example is provided in the figure below.

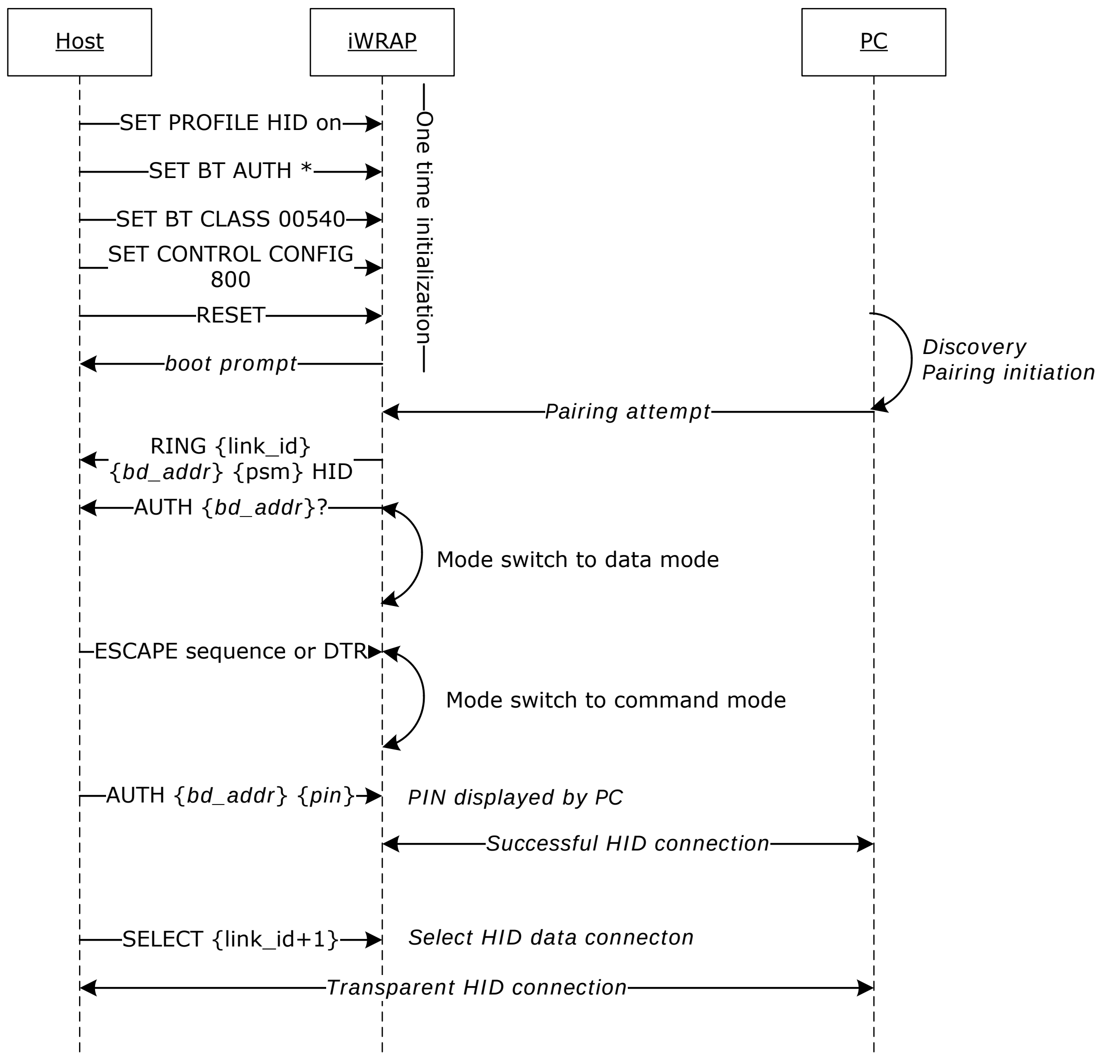


Figure 21: HID keyboard example

In the configuration part pin code needs to be disabled and interactive pairing mode enabled with **SET CONTROL CONFIG 800** as with HID keyboard the HID host will provide a random pin code, which needs to be typed with the keyboard.

The connection needs to be established from the PC and iWRAP will show **CONNECT** and **AUTH** events. Host must respond to the **AUTH** event with **AUTH** command and the pin code shown by the PC. After pairing has been completed the HID control connection needs to be activated.

The above example was made using Broadcom (Widcomm) Bluetooth stack and the operation may vary a little bit with different stacks.

The HID keyboard layout is US and the following key codes are supported.

Code	Description
0	Left control + space
1	Left control + a
2	Left control + b
3	Left control + c
4	Left control + d
5	Left control + e
6	Left control + f
7	Left control + g
8	Backspace
9	Tab
10	Enter
11	Left control + k
12	Left control + l
13	Enter
14	Left control + n
15	Left control + o
16	Left control + p
17	Left control + q
18	Left control + r
19	Left control + s
20	Left control + t
21	Left control + u
22	Left control + v

23	Left control + w
24	Left control + x
25	Left control + y
26	Left control + z
0	Left control + space
28	Esc
28-31	N/A
32-126	Corresponding ASCII character
127	backspace
128	Cursor up
129	Cursor right
130	Cursor down
131	Cursor left
132	Insert
133	Delete
134	Home
135	End
136	Page up
137	Page down
138	Mouse buttons up
139	Mouse up (10px)
140	Mouse right (10px)
141	Mouse down (10px)
142	Mouse left (10px)
143	Mouse button 1 (first clear with 138)

144	Mouse button 2 (first clear with 138)
145	Mouse button 3 (first clear with 138)
146-158	-
159	raw mode*
160-255	-

Table 20: Available HID key codes and mouse events

Raw mode enables sending of raw HID reports, one report at a time. After the rawmode byte (159) you need to give the length of the report which is in keyboard report's case 10 and in mouse report's case 5. The reports must use the following format:

Keyboard report:

0xa1	**0x01**	**modifier**	**0x00**	**key code 1**	**key code 2**	**key code 3**	**key code 4**	**key code 5**	**key code 6**

Figure 22: Raw HID keyboard report

Mouse report:

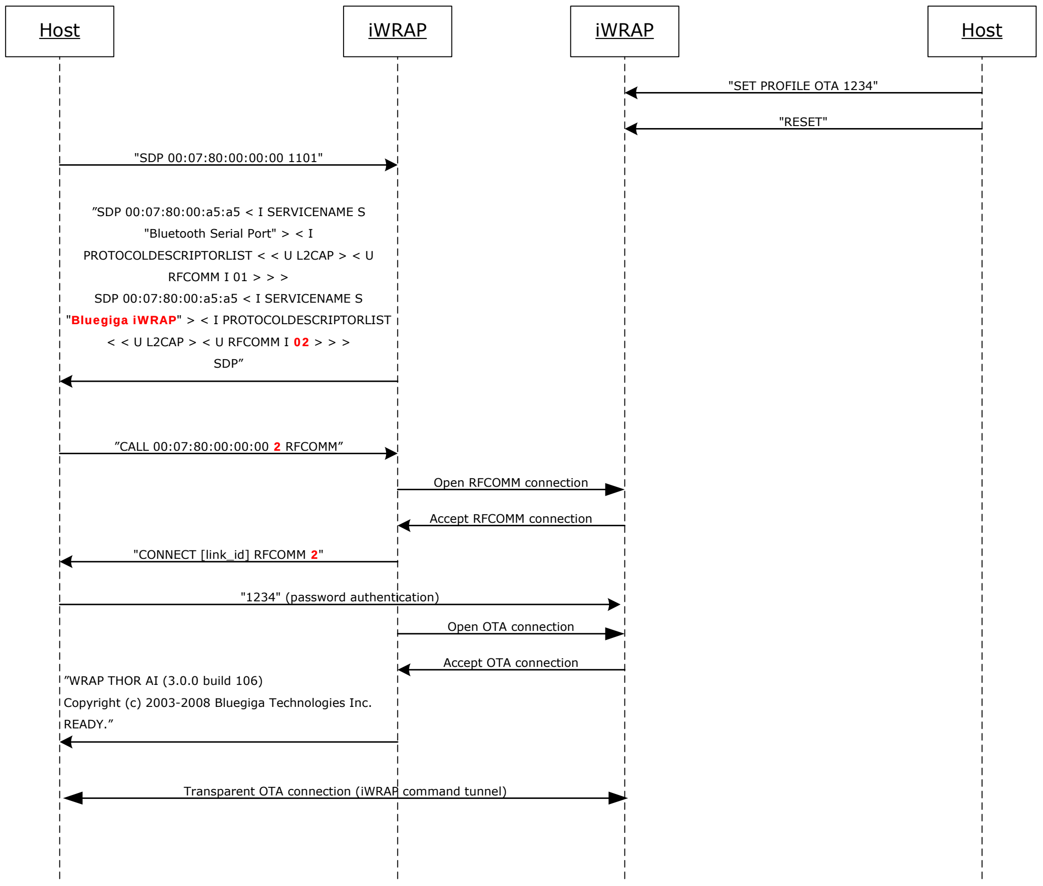
0xa2	**0x02**	**buttons**	**x-step**	**y-step**

Figure 23: Raw HID mouse report

Key codes can be found from document *USB HID Usage Tables.*

12.13 Over-the-Air Configuration

iWRAP3 has Over-the-Air (OTA) configuration interface, which allows one to configure iWRAP settings over a *Bluetooth* SPP connection. OTA gives one access to standard iWRAP commands which also available over UART interface. This example shows how OTA interface can be accessed from another iWRAP device.


Host
iWRAP
iWRAP
Host
"SET PROFILE OTA 1234"
"RESET"
"SDP 00:07:80:00:00:00 1101"
"CALL 00:07:80:00:00:00 2 RFCOMM"
Open RFCOMM connection
Accept RFCOMM connection
"CONNECT [link_id] RFCOMM 2"
"1234" (password authentication)
Open OTA connection
Accept OTA connection
Transparent OTA connection (iWRAP command tunnel)

```
"SDP 00:07:80:00:a5:a5 < I SERVICENAME S
"Bluetooth Serial Port" > < I
PROTOCOLDESCRIPTORLIST < < U L2CAP > < U
RFCOMM I 01 > > >
SDP 00:07:80:00:a5:a5 < I SERVICENAME S
"Bluegiga iWRAP" > < I PROTOCOLDESCRIPTORLIST
< < U L2CAP > < U RFCOMM I 02 > > >
SDP"
```

```
"WRAP THOR AI (3.0.0 build 106)
Copyright (c) 2003-2008 Bluegiga Technologies Inc.
READY."
```

Figure 24: Over-the-Air connection example

On a remote iWRAP OTA is simply activated by issuing iWRAP command: **SET PROFILE OTA {*password*}** and by performing a reset.

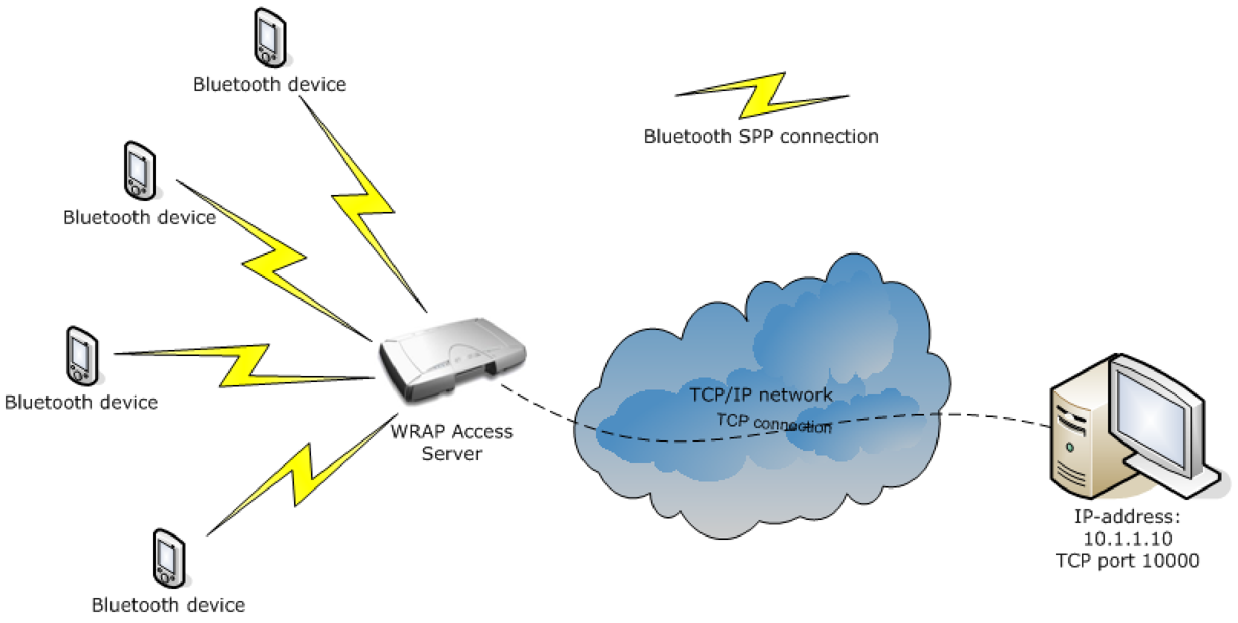
In the *Bluetooth* interface OTA is seen as a standard Bluetooth Serial Port Profile service with a fixed service name "***Bluegiga iWRAP***".

When OTA connection is opened the first thing that needs to be done is to send the password from the controlling device to the controlled iWRAP. If the password is correct iWRAP boot prompt will be displayed, otherwise the connection will be closed.

There is a special use case for OTA to remotely read/write the GPIO pins of the iWRAP under control.

12.14 ***Bluetooth*** **Networking with iWRAP and WRAP Access Server**

In this example, a *Bluetooth* network with WRAP Access Servers and iWRAP master modules is built. The network consists of several access servers and several iWRAP modules. The purpose of this network is to provide a transparent 'always on' connectivity from iWRAP modules to a PC over *Bluetooth* and Local Area Network. The figure below illustrates this kind of a network set up:

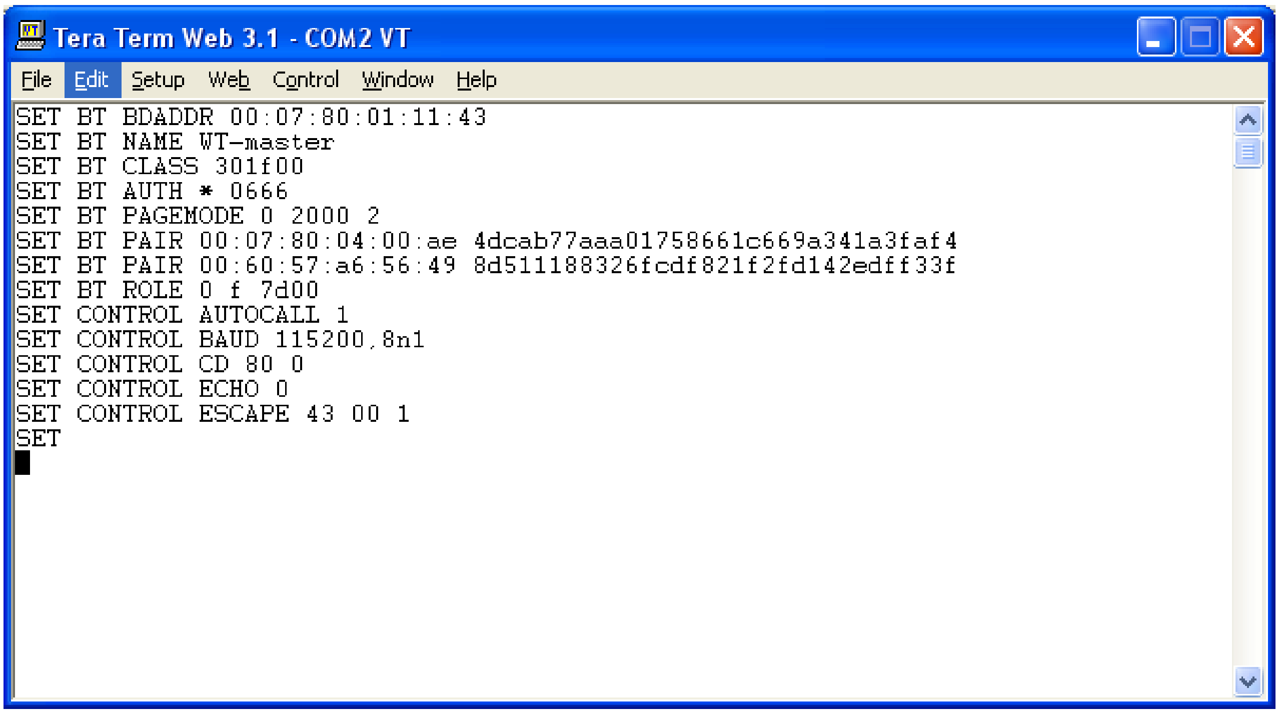


The configuration in iWRAP is similar to the one in our second example.

Also the WRAP Access Servers must be configured correctly. The application providing the connectivity between the PC and iWRAPs is know as SPP-over-IP and it is a standard feature in the WRAP Access Server with software version 2.0.4 and later.

Please refer to SPP-over-IP documentation to see how access servers are configured.

Now, if there are several WRAP Access Servers in our network and iWRAP devices are mobile, a little bit more configuration in iWRAP modules is needed. In a mobile situation, we want the iWRAP to connect to the WRAP Access Server which is in its range.


Tera Term Web 3.1 - COM2 VT
File Edit Setup Web Control Window Help

```
SET BT BDADDR 00:07:80:01:11:43
SET BT NAME WT-master
SET BT CLASS 301f00
SET BT AUTH * 0666
SET BT PAGEMODE 0 2000 2
SET BT PAIR 00:07:80:04:00:ae 4dcab77aaa01758661c669a341a3faf4
SET BT PAIR 00:60:57:a6:56:49 8d511188326fcdf821f2fd142edff33f
SET BT ROLE 0 f 7d00
SET CONTROL AUTOCALL 1
SET CONTROL BAUD 115200,8n1
SET CONTROL CD 80 0
SET CONTROL ECHO 0
SET CONTROL ESCAPE 43 00 1
SET
```

Figure 25: Configuration for multiple slaves

As you see, the configuration is similar to the configuration in the second example. The only difference is that now there are two pairings in iWRAP, that is, iWRAP is paired with two access servers.

If there are several pairings in iWRAP and the autocall feature is enabled, a transparent inquiry is made. When the first paired device is found in the inquiry, iWRAP ends the inquiry and tries to connect to this device. If the connection is successful, iWRAP stays connected until the connection is closed or lost.

If all the access servers in this network are paired with all the iWRAPs, it is possible to achieve a transparent 'data only, always on' network with this configuration. Of course, there will be a short break in the connection if the connection is closed or lost and iWRAP is searching for a connection to a new access server.

13. SUPPORT

- For technical questions and problems, please contact: support@bluegiga.com
- Firmware, parameters, tools and documentation can be downloaded from: http://techforum.bluegiga.com
- iWRAP FAQ can be found from: http://techforum.bluegiga.com/faq

14. RELATED DOCUMENTATION

Please also take a look at the following documentation:

- **iWRAP Update Client User Guide**
- **Firmware & PS-key Guide**
- ***Bluetooth* specification: http://www.bluetooth.org**

Visit also Tech Forum for additional information and design references: http://techforum.bluegiga.com.

15. CONTACT INFORMATION

Sales:	sales@bluegiga.com
Technical support:	support@bluegiga.com http://techforum.bluegiga.com
Orders:	orders@bluegiga.com

Head Office / Finland

Phone:	+358-9-4355 060
Fax:	+358-9-4355 0660
Street Address:	Sinikalliontie 5 A 02630 ESPOO FINLAND
Postal address:	P.O. BOX 120 02631 ESPOO, FINLAND

Sales Office / USA

Phone:	(781) 556-1039
Street address:	Bluegiga Technologies, Inc. 99 Derby Street, Suite 200 Hingham, MA 02043