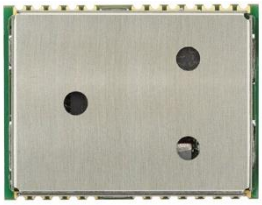
S

Bluetooth audio configuration guide

JODY-W2, JODY-W3 and MAYA-W1

Application note



Abstract

This application note describes how to configure the JODY-W2/3 and MAYA-W1 digital audio interface and audio codec on the evaluation kits for use with the Bluetooth Advanced Audio Distribution Profile (A2DP) and Hands-Free Profile (HFP) applications in Linux.

UBX-20029333 - R02
C1-Public

Document information

Title	**Bluetooth audio configuration guide**	
Subtitle	JODY-W2, JODY-W3 and MAYA-W1	
Document type	Application note	
Document number	UBX-20029333	
Revision and date	R02	20-May-2022
Disclosure restriction	C1-Public	

This document applies to the following products:

Product name
EVK-JODY-W263
EVK-JODY-W374
EVK-JODY-W377
EVK-MAYA-W161
EVK-MAYA-W166

Contents

1 Overview

This document describes how JODY-W2, JODY-W3, and MAYA-W1 series modules are configured for use with Bluetooth audio applications in Linux operating systems.

Explaining how to use the Advanced Audio Distribution Profile (A2DP) for streaming multimedia audio over a Bluetooth connection from a remote device, it also describes how the Bluetooth Hands-Free Profile (HFP) is used to communicate with mobile phones for voice calling applications – typically used inside cars. It also explains how to configure the digital audio data path between the modules and the audio codec in JODY-W2/JODY-W3/MAYA-W1 evaluation kits (EVK) for use with HFP.

Several examples, describing how to set up a system with the evaluation kits, configure the audio data path, and connect over Bluetooth to a mobile phone for A2DP audio streaming and HFP voice calling, are given. A Toradex Apalis TK1 [12] with Ixora carrier board [8] is used as the Linux host platform that connects to the EVKs. The examples can also be used with other host platforms, but some adaptions might be necessary.

2 Setting up the EVKs

2.1 Connecting to the host board

Figure 1 shows the basic setup with the JODY-W2 EVK [9] and the TK1 host board. The SDIO interface is used in this example to connect the Wi-Fi and Bluetooth functions of the module to the host system.

A headphone needs to be connected to the audio jack on the TK1 board located below the SD card slot (not in image) for the A2DP application. A USB headphone could also be used. A headset must be plugged in the audio jack on the EVB (bottom side) for use with HFP, as shown in Figure 1.

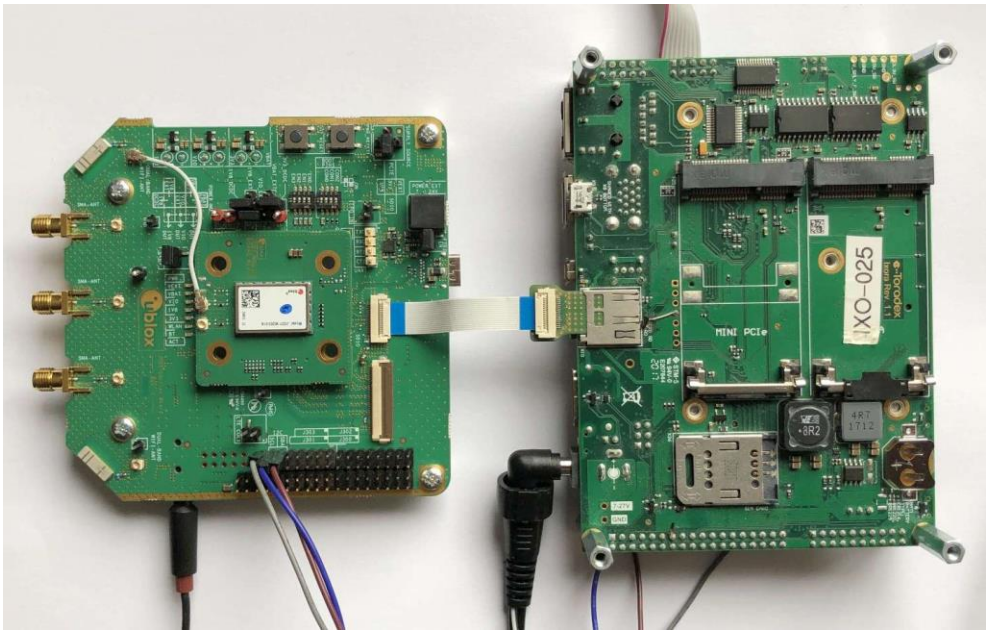


Figure 1: JODY-W2 EVK connected through SDIO (Wi-Fi and Bluetooth)

Figure 2 shows a similar arrangement using JODY-W3 EVK [10]. In this setup, the PCIe interface is used to connect the Wi-Fi function, and the USB-C interface is used to connect the Bluetooth UART of the module through a USB-to-UART bridge.

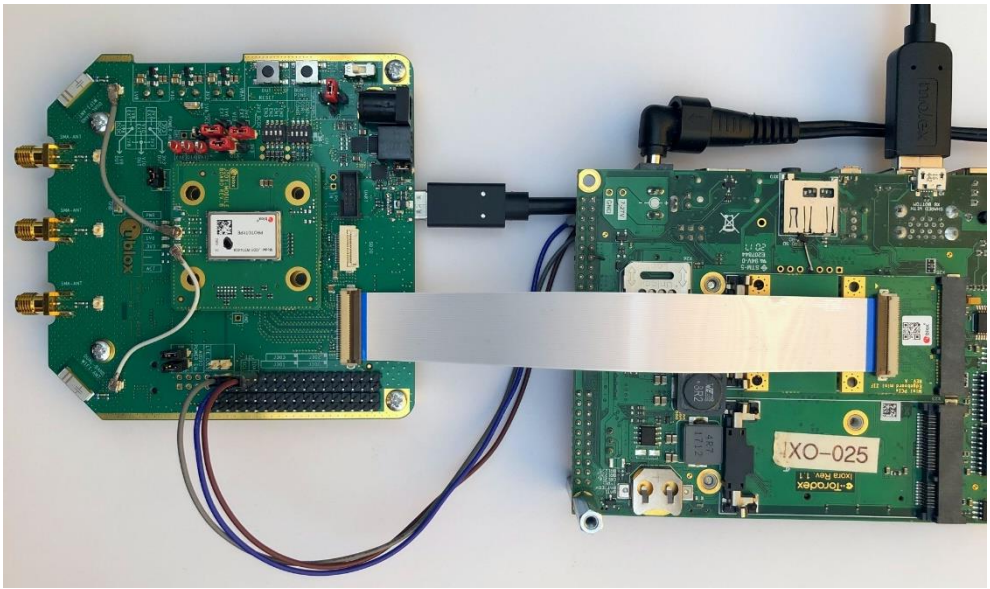


Figure 2: JODY-W3 EVK connected through PCIe (Wi-Fi) and USB-to-UART (Bluetooth)

Figure 3 shows the EVK-MAYA-W1 evaluation board [11] and its main connectors, including the USB-C interface and audio jack for Bluetooth. A micro-SD card adapter is used to connect the Wi-Fi SDIO interface to the host system and download the firmware.

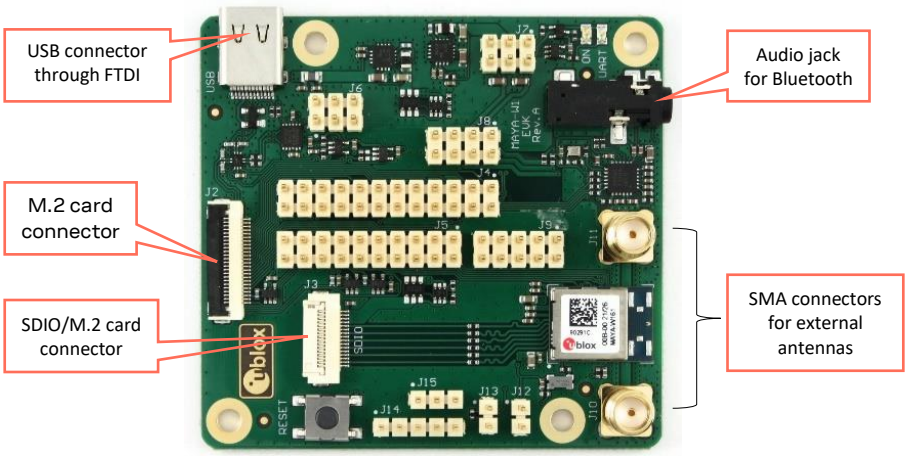


Figure 3: Evaluation board of EVK-MAYA-W161

2.2 Driver loading

The software running on the TK1 board is based on an embedded Linux image from Toradex (kernel version 4.14.90), with the *meta-ublox-modules* Yocto/OpenEmbedded meta layer added on top to provide the necessary drivers, firmware, and tools for the modules.

The following snippet shows how to load the drivers and firmware for the modules and enable the Bluetooth interface.

JODY-W2 (SDIO-SDIO):

```
root@tk1:~# modprobe jody-w2-sdio
root@tk1:~# modprobe jody-w2-sdio-bt
root@tk1:~# hciconfig hci0 up
```

In the above example, the first command loads the Wi-Fi driver and firmware for the JODY-W2 module and is optional. If this initial command is omitted, the firmware is loaded through the Bluetooth driver in the second command. The last command in this example brings up the Bluetooth HCI interface.

JODY-W3 (PCIe-UART):

```
root@tk1:~# modprobe jody-w3-pcieuart
root@tk1:~# modprobe jody-w3-pcieuart-bt
root@tk1:~# hciattach /dev/ttyUSB0 any 3000000 flow
root@tk1:~# hciconfig hci0 up
```

In the above example, the first command loads the Wi-Fi driver and firmware for the JODY-W3 module. The other commands load the Bluetooth driver, attach the serial interface to the stack, and bring up the HCI interface. If Wi-Fi is omitted, the Bluetooth firmware must be loaded separately through the serial interface first:

```
root@tk1:~# /opt/jody-w3/pcieuart/fw_loader /dev/ttyUSB0 115200 0 lib/firmware/nxp/jody-
   w3-pcieuart/uart9098_bt_v1.bin 3000000
root@tk1:~# modprobe jody-w3-pcieuart-bt
root@tk1:~# hciattach /dev/ttyUSB0 any 3000000 flow
root@tk1:~# hciconfig hci0 up
```

MAYA-W1 (SDIO-UART):

```
root@tk1:~# modprobe moal mod_para=nxp/wifi_mod_para.conf
root@tk1:~# modprobe hci_uart
root@tk1:~# hciattach /dev/ttyUSB0 any 3000000 flow
root@tk1:~# hciconfig hci0 up
```

In the above example, the first command loads the NXP open-source Wi-Fi driver and firmware for the MAYA-W1 module. The other commands load the standard Linux HCI UART Bluetooth driver, attach the serial interface to the stack, and bring up the HCI interface.

2.3 Audio codec for HFP

The MAX9860 audio codec for voice applications, provided with the JODY-W2/JODY-W3/MAYA-W1 EVK, is connected to the PCM/I2S interface of the module. A 3.5 mm audio jack for connecting a headset is available on the EVB. The audio codec is completely controlled through software using an I2C interface. This section explains how the audio codec is connected to the TK1 host board and how it is controlled.

2.3.1 Connecting to the host board

To power the audio codec on the JODY-W2/JODY-W3 EVB, make sure that jumpers are placed for **VDD** and **VDD_IO**, as shown in Figure 4.

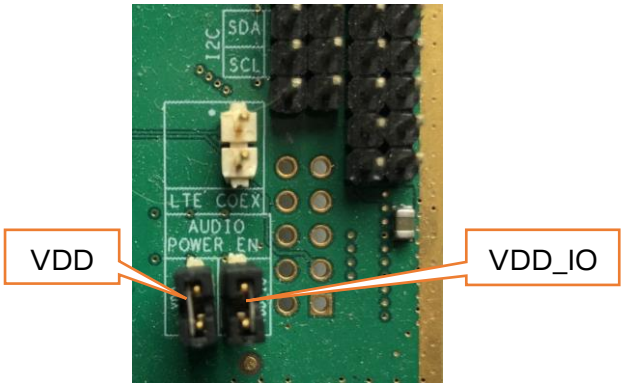


Figure 4: Audio codec power supply jumpers

The I2C interface of the audio codec is provided on connector J303 of the EVBs for JODY-W2/JODY-W3 and on connector J5 of the EVK-MAYA-W1. On the TK1 side, the I2C interface I2C1 is available on connector X27. Use wires to connect the I2C pins of the EVB and the TK1 as described in Table 1 and shown in Figure 5, Figure 6 and Figure 7.

Signal	Description	EVK-JODY-W2/3 (J303)	EVK-MAYA-W1 (J5)	TK1 (X27)
SDA	I2C serial data input/output	11	16	5
SCL	I2C serial data clock	13	18	6
GND	Ground	15	20	7

Table 1: I2C pin assignment

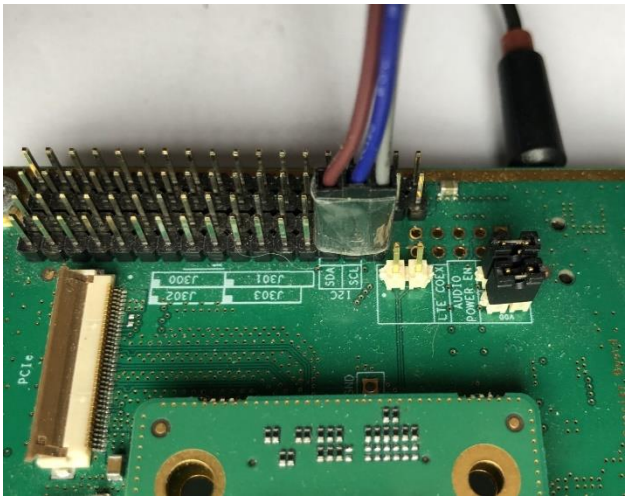


Figure 5: I2C interface on the JODY-W2/3 EVB

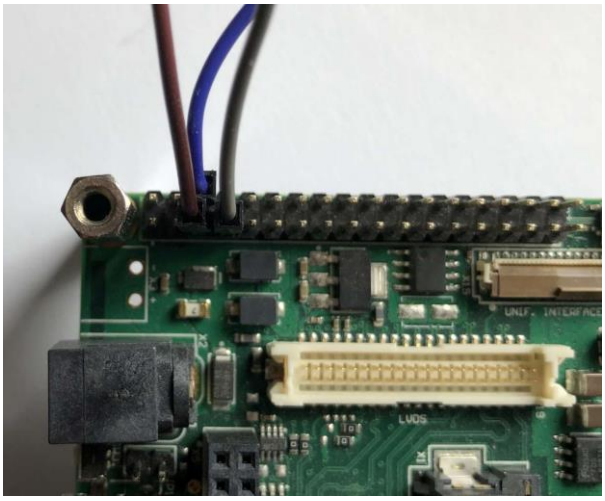


Figure 6: I2C interface on the TK1

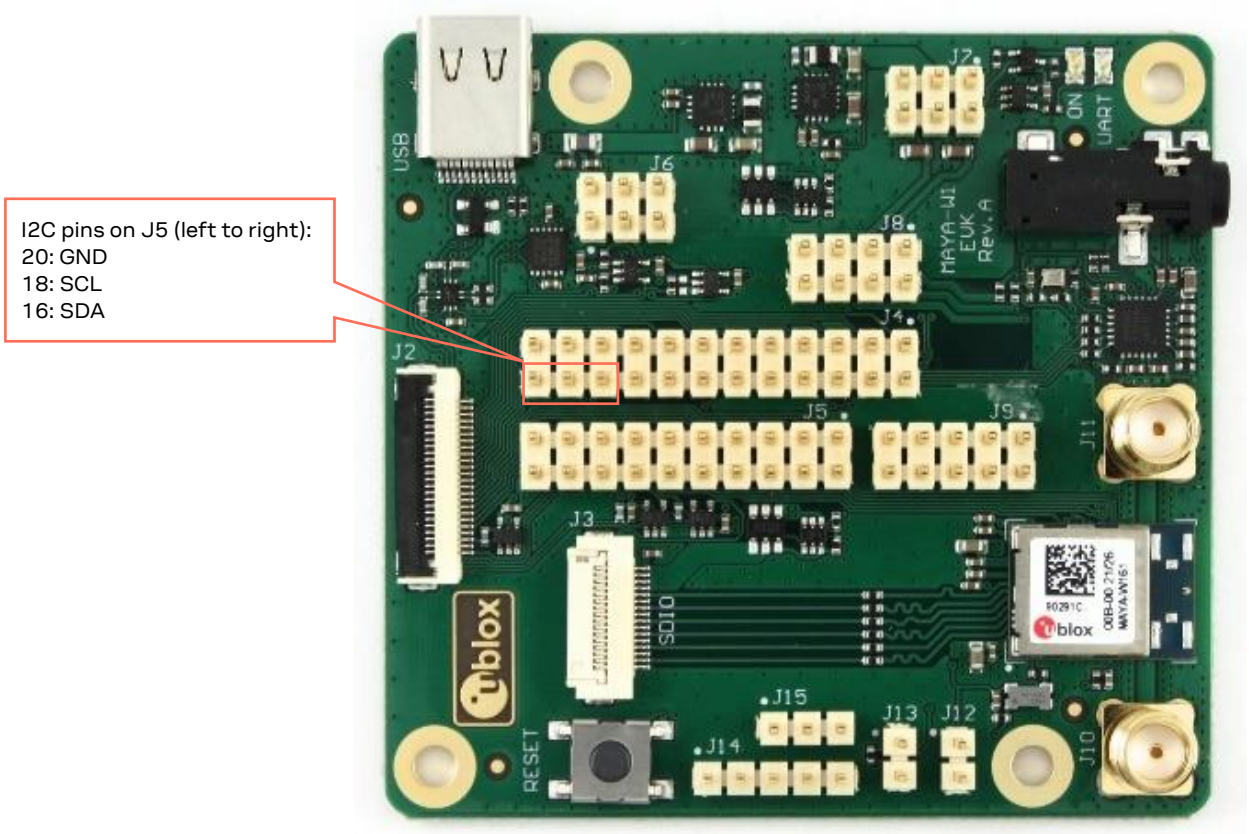


Figure 7: I2C interface on the MAYA-W1 EVB

2.3.2 Configuring the audio codec

The MAX9860 audio codec is connected to I2C bus 0 on the TK1. Its I2C slave address is 0x10. The I2C connection between the audio codec and the TK1 can be verified by reading the codec revision code register at address `0xff`, which should return `0x40`:

```
root@tk1:~# i2cget -y 0 0x10 0xff
0x40
```

Table 2 describes the required register settings of the MAX9860 audio codec for different audio configurations. The most relevant settings are in the system clock control (`0x03`) and digital audio interface (`0x6, 0x7`) registers. Use the following command to apply the specific configuration settings given in Table 2:

```
root@tk1:~# i2cset -y 0 0x10 0x02 <regs 0x02-0x10> i
```

For example, to configure the codec for NBS PCM master mode:

```
root@tk1:~# i2cset -y 0 0x10 0x02 0x00 0x16 0x00 0x00 0xac 0x21 0x00 0x06 0x33 0x00 0x34
   0x00 0x01 0x00 0x8a i
```

Audio configuration	MAX9860 register settings (0x02-0x10)
NBS PCM master	0x00 0x16 0x00 0x00 0xac 0x21 0x00 0x06 0x33 0x00 0x34 0x00 0x01 0x00 0x8a
NBS PCM slave	0x00 0x16 0x00 0x00 0x2c 0x21 0x00 0x06 0x33 0x00 0x34 0x00 0x01 0x00 0x8a
WBS PCM master	0x00 0x17 0x00 0x00 0xac 0x21 0x00 0x06 0x33 0x00 0x34 0x00 0x01 0x00 0x8a
WBS PCM slave	0x00 0x17 0x00 0x00 0x2c 0x21 0x00 0x06 0x33 0x00 0x34 0x00 0x01 0x00 0x8a
NBS I2S master	0x00 0x16 0x00 0x00 0xf0 0x31 0x00 0x06 0x33 0x00 0x34 0x00 0x01 0x00 0x8a
NBS I2S slave	0x00 0x16 0x00 0x00 0x70 0x31 0x00 0x06 0x33 0x00 0x34 0x00 0x01 0x00 0x8a
WBS I2S master	0x00 0x17 0x00 0x00 0xf0 0x31 0x00 0x06 0x33 0x00 0x34 0x00 0x01 0x00 0x8a
WBS I2S slave	0x00 0x17 0x00 0x00 0x70 0x31 0x00 0x06 0x33 0x00 0x34 0x00 0x01 0x00 0x8a

Table 2: MAX9860 register settings for different audio configurations

☞ The I2S modes use LRCLK invert, which can alternatively be enabled on the module side.

3 Bluetooth audio in Linux

The standard Bluetooth stack used in Linux is BlueZ [4]. Several A2DP and HFP implementation are available for Linux. For the purposes of this document, we chose to use PulseAudio [5] to provide the Bluetooth A2DP and HFP functionality. PulseAudio requires BlueZ for all Bluetooth functionality, and oFono [6] is additionally required for HFP support.

3.1 Installation

PulseAudio and oFono can be installed on the TK1 host board from the Toradex package repositories. To install the apps, connect the board to the Internet over Ethernet and run the following commands:

```
root@tk1:~# opkg update
root@tk1:~# opkg install ofono ofono-tests
root@tk1:~# opkg install pulseaudio-server pulseaudio-module-bluetooth-discover
   pulseaudio-module-bluetooth-policy pulseaudio-module-bluez5-discover pulseaudio-
   module-bluez5-device pulseaudio-module-loopback pulseaudio-misc pulseaudio-module-
   switch-on-connect
```

The software versions used in this document are

- BlueZ 5.46 (Linux 4.14.90)
- PulseAudio 10.0
- oFono 1.20

3.2 Configuration

Make necessary changes to the following PulseAudio configuration under /etc/pulse.

/etc/pulse/daemon.conf:

```
exit-idle-time = -1
```

/etc/pulse/client.conf:

```
allow-autospawn-for-root = yes
```

/etc/pulse/default.pa:

```
load-module module-bluetooth-discover headset=ofono
```

3.3 Enable wideband speech

PulseAudio 10.0 supports only narrowband speech (NBS) coding for voice audio with HFP when the module host interface is used as audio data path. Patches are required to support wideband speech (WBS) coding when the PCM/I2S interface is used for audio. The PulseAudio patches provided in Appendix B are used to enable WBS support for PulseAudio during codec negotiation. The patches do not add support for encoding and decoding wideband speech audio in PulseAudio and can only be used when the module PCM/I2S interface is used as the audio data path. In this case, all encoding and decoding of voice audio happens in the module.

☞ The host stack must keep track of the audio application mode (NBS/WBS) before the SCO/eSCO connection is set up. The parameters of the audio data path are then configured accordingly.

3.4 Connect to a remote Bluetooth device

1. Start oFono and PulseAudio services on the TK1 board.

```
root@tk1:~# systemctl start ofono
root@tk1:~# pulseaudio --start
```

Use the following steps to scan, pair, and connect the remote Bluetooth device using the interactive `bluetoothctl` tool provided by BlueZ. These steps are unnecessary if the device is already paired. In which case, it is only necessary to connect the device.

2. Start `bluetoothctl` and make sure that Audio Sink and Handsfree are properly registered in the local Bluetooth controller.

```
root@tk1:~# bluetoothctl
[NEW] Controller 60:09:C3:80:1F:98 BlueZ 5.46 [default]
Agent registered
[bluetooth]# show
Controller 60:09:C3:80:1F:98
        Name: BlueZ 5.46
        Alias: BlueZ 5.46
        Class: 0x2c0000
        Powered: yes
        Discoverable: no
        Pairable: yes
        UUID: Generic Attribute Profile (00001801-0000-1000-8000-00805f9b34fb)
        UUID: A/V Remote Control        (0000110e-0000-1000-8000-00805f9b34fb)
        UUID: Generic Access Profile    (00001800-0000-1000-8000-00805f9b34fb)
        UUID: PnP Information           (00001200-0000-1000-8000-00805f9b34fb)
        UUID: A/V Remote Control Target (0000110c-0000-1000-8000-00805f9b34fb)
        UUID: Audio Sink                (0000110b-0000-1000-8000-00805f9b34fb)
        UUID: Audio Source              (0000110a-0000-1000-8000-00805f9b34fb)
        UUID: Handsfree                 (0000111e-0000-1000-8000-00805f9b34fb)
        Modalias: usb:v1D6Bp0246d052E
        Discovering: no
```

3. Register an authentication agent to handle the passkey entry during the pairing process.

```
[bluetooth]# agent on
Agent is already registered
[bluetooth]# default-agent
Default agent request successful
```

4. To scan and start the pairing process from the remote Bluetooth device, enable discoverable and pairable on the local Bluetooth controller.

```
[bluetooth]# discoverable on
Changing discoverable on succeeded
[CHG] Controller 60:09:C3:80:1F:98 Discoverable: yes
[bluetooth]# pairable on
Changing pairable on succeeded
```

5. Run the following commands to scan for available Bluetooth devices and initiate pairing from the TK1 board.

```
[bluetooth]# scan on
Discovery started
[CHG] Controller 60:09:C3:80:1F:98 Discovering: yes
[NEW] Device 80:82:23:xx:xx:xx Phone
[bluetooth]# scan off
[CHG] Controller 60:09:C3:80:1F:98 Discovering: no
Discovery stopped
```

6. Start pairing with the remote Bluetooth device from either the remote device or TK1 board, as shown below.

```
[bluetooth]# pair 80:82:23:xx:xx:xx
Attempting to pair with 80:82:23:xx:xx:xx
[CHG] Device 80:82:23:xx:xx:xx Connected: yes
Request confirmation
[agent] Confirm passkey 244941 (yes/no):
```

7. Type “yes” to confirm that the passkey matches and complete the pairing process.

```
[agent] Confirm passkey 244941 (yes/no): yes
[CHG] Device 80:82:23:xx:xx:xx Modalias: bluetooth:v004Cp710Ad0E40
[CHG] Device 80:82:23:xx:xx:xx UUIDs: 00000000-deca-fade-deca-deafdecacafe
[CHG] Device 80:82:23:xx:xx:xx UUIDs: 00001000-0000-1000-8000-00805f9b34fb
[CHG] Device 80:82:23:xx:xx:xx UUIDs: 0000110a-0000-1000-8000-00805f9b34fb
[CHG] Device 80:82:23:xx:xx:xx UUIDs: 0000110c-0000-1000-8000-00805f9b34fb
[CHG] Device 80:82:23:xx:xx:xx UUIDs: 0000110e-0000-1000-8000-00805f9b34fb
[CHG] Device 80:82:23:xx:xx:xx UUIDs: 00001116-0000-1000-8000-00805f9b34fb
[CHG] Device 80:82:23:xx:xx:xx UUIDs: 0000111f-0000-1000-8000-00805f9b34fb
[CHG] Device 80:82:23:xx:xx:xx UUIDs: 0000112f-0000-1000-8000-00805f9b34fb
[CHG] Device 80:82:23:xx:xx:xx UUIDs: 00001132-0000-1000-8000-00805f9b34fb
[CHG] Device 80:82:23:xx:xx:xx UUIDs: 00001200-0000-1000-8000-00805f9b34fb
[CHG] Device 80:82:23:xx:xx:xx UUIDs: 00001801-0000-1000-8000-00805f9b34fb
[CHG] Device 80:82:23:xx:xx:xx UUIDs: 02030302-1d19-415f-86f2-22a2106a0a77
[CHG] Device 80:82:23:xx:xx:xx ServicesResolved: yes
[CHG] Device 80:82:23:xx:xx:xx Paired: yes
Pairing successful
[CHG] Device 80:82:23:xx:xx:xx ServicesResolved: no
[CHG] Device 80:82:23:xx:xx:xx Connected: no
```

8. Trust the remote device once to create a bonding and then connect to it. By trusting the device, the keys are stored and used the next time a connection is created.

```
[CHG] Device 80:82:23:xx:xx:xx Trusted: yes
Changing 80:82:23:xx:xx:xx trust succeeded
[bluetooth]# connect 80:82:23:xx:xx:xx
Attempting to connect to 80:82:23:xx:xx:xx
[CHG] Device 80:82:23:xx:xx:xx Connected: yes
Connection successful
[CHG] Device 80:82:23:xx:xx:xx ServicesResolved: yes
[Phone]#
```

9. Verify the Bluetooth profiles supported by the remote device. For the examples given in this application note, the device must support the Audio Source role for A2DP and Handsfree Audio Gateway role for HFP, as shown below.

```
[Phone]# info 80:82:23:xx:xx:xx
Device 80:82:23:xx:xx:xx
        Name: Phone
        Alias: Phone
        Class: 0x7a020c
        Icon: phone
        Paired: yes
        Trusted: yes
        Blocked: no
        Connected: no
        LegacyPairing: no
        UUID: Vendor specific           (00000000-deca-fade-deca-deafdecacafe)
        UUID: Service Discovery Serve.. (00001000-0000-1000-8000-00805f9b34fb)
        UUID: Audio Source              (0000110a-0000-1000-8000-00805f9b34fb)
        UUID: A/V Remote Control Target (0000110c-0000-1000-8000-00805f9b34fb)
        UUID: A/V Remote Control        (0000110e-0000-1000-8000-00805f9b34fb)
        UUID: NAP                       (00001116-0000-1000-8000-00805f9b34fb)
        UUID: Handsfree Audio Gateway   (0000111f-0000-1000-8000-00805f9b34fb)
        UUID: Phonebook Access Server   (0000112f-0000-1000-8000-00805f9b34fb)
        UUID: Message Access Server     (00001132-0000-1000-8000-00805f9b34fb)
        UUID: PnP Information           (00001200-0000-1000-8000-00805f9b34fb)
        UUID: Generic Attribute Profile (00001801-0000-1000-8000-00805f9b34fb)
        UUID: Vendor specific           (02030302-1d19-415f-86f2-22a2106a0a77)
        Modalias: bluetooth:v004Cp710Ad0E40
```

10. Type ‘quit’ to exit from `bluetoothctl` and return to the Linux command line.

```
[Phone]# quit
Agent unregistered
[DEL] Controller 60:09:C3:80:1F:98 BlueZ 5.46 [default]
root@tk1:~#
```

11. Use the following `bluetoothctl` commands to disconnect and unpair a remote device if needed later.

```
[Phone]# disconnect 80:82:23:xx:xx:xx
Attempting to disconnect from 80:82:23:xx:xx:xx
[CHG] Device 80:82:23:xx:xx:xx ServicesResolved: no
Successful disconnected
[CHG] Device 80:82:23:xx:xx:xx Connected: no
[bluetooth]# remove 80:82:23:xx:xx:xx
[DEL] Device 80:82:23:xx:xx:xx Phone
Device has been removed
```

4 Advanced audio distribution profile

The Advanced Audio Distribution Profile (A2DP) [3] defines the protocols and procedures for streaming high-quality audio content over Bluetooth ACL links. A2DP is typically used for streaming music content from a stereo music player to headphones or speakers, where the audio data is compressed to a proper format for efficient use of the limited bandwidth. Remote control functions are supported by A2DP in conjunction with the Audio/Video Remote Control Profile (AVRCP).

A2DP defines the following roles:

- Source – The device acting as the source of a digital audio stream. Typical source devices include portable music players, smartphones, or microphones.
- Sink – The device acting as the sink of a digital audio stream. Typical sink devices include headphones or portable music recorders.

4.1 A2DP sink

The following snippet describes how to configure the TK1 board as an A2DP sink to play audio from a connected source device over Bluetooth.

1. Connect with a remote Bluetooth device, like a smartphone, that supports the A2DP Audio Source role. For information about starting PulseAudio and connecting to the Bluetooth device, see also Connect to a remote Bluetooth device.

```
root@tk1:~# bluetoothctl
[bluetooth]# connect 80:82:23:XX:XX:XX
Attempting to connect to 80:82:23:XX:XX:XX
[CHG] Device 80:82:23:XX:XX:XX Connected: yes
Connection successful
[Phone]# info
Device 80:82:23:XX:XX:XX
[…]
        UUID: Audio Source              (0000110a-0000-1000-8000-00805f9b34fb)
[…]
[Phone]# quit
```

2. Check the available sink cards for audio output on the TK1 board.

```
root@tk1:~# pactl list short sinks
0       alsa_output.platform-sound.analog-stereo        module-alsa-card.c      s16le
   2ch 44100Hz        SUSPENDED
1       bluez_sink.80_82_23_XX_XX_XX.headset_audio_gateway      module-bluez5-
   device.c  s16le 1ch 8000Hz        SUSPENDED
```

3. Select the analog audio output of the Apalis TK1 as the default sink and set the volume to 100% to listen to music through the board audio jack.

```
root@tk1:~# pactl set-default-sink 0
root@tk1:~# pactl set-sink-volume 0 100%
```

4. Connect a headphone or speaker to the TK1 audio jack on the Ixora carrier board and start playing music on the remote Bluetooth device. The music should now play from the speaker connected to the TK1 board. You can use the *bluetooth-player* included in the *bluez5-noinst-tools* package to control music playback and show player information through AVRCP.

5 Hands-Free Profile

This section describes the Hands-Free Profile (HFP) audio data path configurations for Bluetooth in JODY-W2, JODY-W3, and MAYA-W1 series modules evaluation kits (EVK). It also explains how to use the HFP implementation in Linux operating systems.

The Bluetooth Hands-Free Profile [3] enables a headset or hands-free unit to connect wirelessly to a cellular phone. In this use case, the hands-free unit provides the audio input and output mechanisms of the cellular phone and allows typical telephony functions to be performed without access to the actual phone. The following roles are defined for HFP:

- Audio Gateway (AG) – This is the device that is the gateway of the audio, both for input and output. Typical devices acting as Audio Gateways are cellular phones.
- 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.

The examples given in this document show how to set up the TK1 board with the u-blox module EVK as the HF unit connecting over Bluetooth to a cellular phone acting as the AG. The audio input/output mechanism is provided by the MAX9860 digital audio codec [7] on the evaluation boards. Figure 8 shows the setup for an HFP use case.

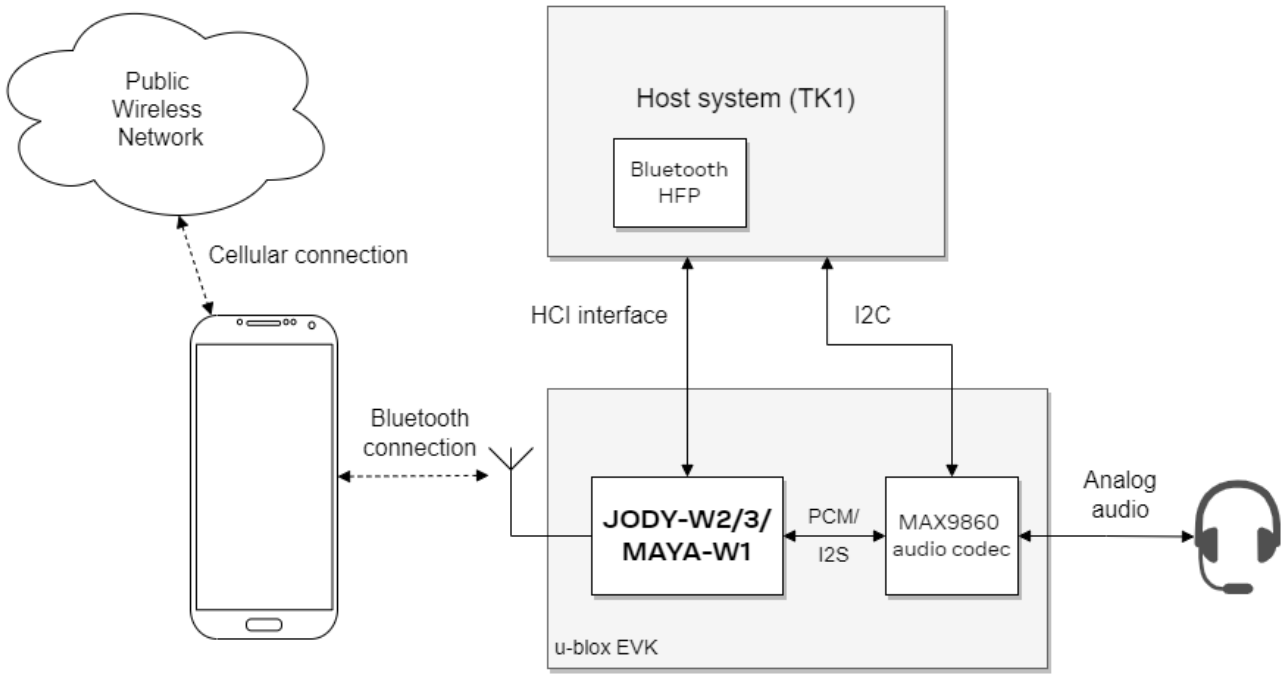


Figure 8: HFP use case and setup

5.1 Digital audio interface for Bluetooth

HFP uses Synchronous Connection Oriented (SCO)/Extended SCO (eSCO) logical links for the audio connection.

The data paths supported for SCO/eSCO audio on JODY-W2/JODY-W3/MAYA-W1 series modules include:

- SCO/eSCO over UART or SDIO host interface (Narrowband speech (NBS))
- SCO/eSCO over PCM interface (NBS/Wideband speech (WBS))
- SCO/eSCO over I2S interface (NBS/WBS)

5.1.1 PCM/I2S interface

JODY-W2/JODY-W3 and MAYA-W1 series modules support a PCM interface with:

- PCM Master or Slave mode
- PCM bit width size of 8 bits or16 bits
- 8 kHz (NBS) or 16 kHz (WBS) PCM_SYNC signal
- Up to four PCM slots with configurable bit width and start positions
- PCM short frame and long frame[1] synchronization

JODY-W3 and MAYA-W1 series modules also support an I2S interface with:

- I2S-justified master or slave mode

☞ PCM pins are shared with the I2S interface and can be configured to PCM or I2S mode using HCI commands.

In PCM master mode, the interface generates a 2 MHz or 2.048 MHz **PCM_CLK**, and an 8 kHz or 16 kHz **PCM_SYNC** signal. In PCM slave mode, the signals are used as inputs and are generated by another master device on the PCM bus. The **PCM_SYNC** short frame synchronization is a one clock-wide pulse and one clock ahead of the first data bit. The PCM interface consists of up to four, time-divided, PCM slots that are preceded by a PCM synchronization signal. Each PCM slot is 16 bits wide. The slots can be separated in time but are not required to follow immediately after one another. The timing is relative to **PCM_CLK**.

In I2S mode, the interface supports I2S-justified master or slave mode. Serial data is transmitted on the falling edge of **BCLK** and sampled at the rising edge of **BLCK**. The Most Significant Bit (MSB) is transmitted on the falling edge of the first **BLCK** after the **LRCLK** transition.

5.1.2 Configuration commands

Table 3 defines the vendor specific HCI commands used for configuring the audio data path and PCM/I2S interface on JODY-W2, JODY-W3, and MAYA-W1 series modules[2].

HCI command	OCF	Description
HCI_CMD_SET_SCO_DATA_PATH	0x001D	Selects between PCM/I2S and host data path
HCI_CMD_WRITE_PCM_I2S_SETTINGS	0x0007	Configures PCM/I2S settings
HCI_CMD_WRITE_PCM_I2S_SYNC_SETTINGS	0x0028	Configures PCM/I2S synchronization settings
HCI_CMD_WRITE_PCM_LINK_SETTINGS	0x0029	Configures PCM link settings
HCI_CMD_SET_I2S_SETTINGS	0x0079	Configures I2S settings (EMMY-W1 series only)
HCI_CMD_SET_I2S_LINK_SETTINGS	0x007A	Configures I2S link settings (EMMY-W1 series only)
HCI_CMD_SET_WBS_CONNECTION	0x0073	Enables or disables WBS for the next SCO connection
HCI_CMD_HOST_PCM_I2S_AUDIO_CONFIG	0x006F	Start or stop audio on PCM/I2S
HCI_CMD_HOST_PCM_I2S_CONTROL_ENABLE	0x0070	Enables or disables host software control over PCM/I2S

Table 3: PCM/I2S configuration commands

For more information about the HCI commands, see Enhanced SCO commands – vendor specific parameters for NBS/WBS. For information about the commands needed for configuring the digital audio interface on the audio codec side, see also Configuring the audio codec.

[1] In PCM Master mode, PCM long frame synchronization is 1 clock wide. In PCM Slave mode, the long frame synchronization pattern of the PCM Master is supported.
[2] The HCI commands are partially applicable to other NXP-based modules, like EMMY-W1 series. Some exceptions are noted.

5.2 Hands-Free call procedure

Use the following procedure to configure JODY-W2/JODY-W3/MAYA-W1 series modules for making an HFP audio call to a cellular phone connected over Bluetooth, using oFono software and the hands-free profile (HFP). The digital audio interface of the module is configured to route the audio data to the codec provided on the EVB. A headset also needs to be plugged in the audio jack on the EVB.

1. During Bluetooth host stack initialization, apply voice settings using the standard HCI Write Voice Setting command.

```
root@tk1:~# hcitool cmd 0x03 0x0026 0x60 0x00
```

2. Enable control of the PCM/I2S audio interface through the host software, using the HCI_CMD_HOST_PCM_I2S_CONTROL_ENABLE command.

☞ Enabling host control of the PCM/I2S interface and use of enhanced SCO commands are mandatory for supporting Wide Band Speech (WBS) on JODY-W3/MAYA-W1 series modules and is optional for the JODY-W2 series.

```
root@tk1:~# hcitool cmd 0x3F 0x0070 0x01
```

3. Set up the PCM/I2S interface as the audio path for the module, using the HCI_CMD_SET_SCO_DATA_PATH command.

```
root@tk1:~# hcitool cmd 0x3F 0x001D 0x01
```

4. Configure the corresponding PCM/I2S settings for the module and the audio codec on the EVB. For information about the vendor specific HCI commands used to configure the PCM/I2S interface of the JODY-W2/JODY-W3/MAYA-W1, see PCM/I2S configuration. To learn more about the different PCM/I2S configurations on codec side, see also Configuring the audio codec.
5. Connect with a remote Bluetooth device, like a smartphone, that supports the HFP Audio Gateway role. To start PulseAudio and oFono and connect to the remote Bluetooth device, see also Connect to a remote Bluetooth device.

```
root@tk1:~# bluetoothctl
[bluetooth]# connect 80:82:23:XX:XX:XX
Attempting to connect to 80:82:23:XX:XX:XX
[CHG] Device 80:82:23:XX:XX:XX Connected: yes
Connection successful
[Phone]# info
Device 80:82:23:XX:XX:XX
[…]
        UUID: Handsfree Audio Gateway   (0000111f-0000-1000-8000-00805f9b34fb)
[…]
[Phone]# quit
```

6. Initiate or answer a call using oFono test scripts. The test scripts use the D-Bus interface to control oFono.
   - First, enable the connected HFP modem:
     ```
     root@tk1:/usr/lib/ofono/test# ./enable-modem..
     Connecting modem /hfp/org/bluez/hci0/dev_80_82_23_XX_XX_XX...
     ```
   - To initiate an outgoing call to a phone number:
     ```
     root@tk1:/usr/lib/ofono/test# ./dial-number <phone number>
     Using modem /hfp/org/bluez/hci0/dev_80_82_23_XX_XX_XX
     /hfp/org/bluez/hci0/dev_80_82_23_XX_XX_XX/voicecall01
     ```
   - To accept an incoming call:
     ```
     root@tk1:/usr/lib/ofono/test# ./answer-calls
     [ /hfp/org/bluez/hci0/dev_80_82_23_XX_XX_XX ]
     [ /hfp/org/bluez/hci0/dev_80_82_23_XX_XX_XX/voicecall01 ] incoming
     ```
7. The local and remote HFP devices negotiate the air codec used for the voice connection, which can be either CVSD (NBS) or mSBC (WBS) – depending on the device capabilities. The codec negotiation is handled by the hands-free profile.
8. SCO/eSCO connection with the remote device is created using the legacy or enhanced SCO setup/accept commands.
   - The parameter values to be used with the enhanced SCO commands for NBS and WBS are described in Enhanced SCO commands – vendor specific parameters for NBS/WBS.
   - To ensure that WBS is enabled, the HCI_CMD_SET_WBS_CONNECTION command must be used before setting up or accepting the connection with legacy SCO commands and the air codec mSBC.
9. The NBS/WBS call is active after the SCO/eSCO connection has been established. Audio over PCM/I2S must be explicitly enabled for the SCO/eSCO connection handle with the HCI_CMD_HOST_PCM_I2S_AUDIO_CONFIG command, in case host control of the PCM/I2S interface is enabled through HCI_CMD_HOST_PCM_I2S_CONTROL_ENABLE.

   ```
   root@tk1:~# hcitool con
   Connections:
           > eSCO 80:82:23:XX:XX:XX handle 6 state 1 lm SLAVE
           > ACL 80:82:23:XX:XX:XX handle 11 state 1 lm MASTER AUTH ENCRYPT
   root@tk1:~# hcitool cmd 0x3F 0x006F 0x00 0x00 0x06 0x00 0x00 0x00
   ```

10. Voice call should be active using the speakers and microphone of the headset connected to the audio codec on the EVB and the voice audio should be audible on both sides.

5.2.1 PCM/I2S configuration

This section provides the vendor specific HCI commands used to configure the PCM/I2S interface of the modules. For more information about the corresponding settings for the audio codec on the EVBs, see also Configuring the audio codec.

1. Define the PCM/I2S settings (HCI_CMD_WRITE_PCM_I2S_SETTINGS)
   - PCM/I2S master mode:
     ```
     root@tk1:~# hcitool cmd 0x3F 0x0007 0x02
     ```
   - PCM/I2S slave mode:
     ```
     root@tk1:~# hcitool cmd 0x3F 0x0007 0x00
     ```

2. Define the PCM/I2S sync settings (HCI_CMD_WRITE_PCM_I2S_SYNC_SETTINGS)
   - PCM, 8 kHz sync:
     ```
     root@tk1:~# hcitool cmd 0x3F 0x0028 0x03 0x00 0x03
     ```
   - PCM, 16 kHz sync:
     ```
     root@tk1:~# hcitool cmd 0x3F 0x0028 0x03 0x00 0x07
     ```
   - I2S, 8 kHz sync:
     ```
     root@tk1:~# hcitool cmd 0x3F 0x0028 0x03 0x0E 0x03
     ```
   - I2S, 16 kHz sync:
     ```
     root@tk1:~# hcitool cmd 0x3F 0x0028 0x03 0x0E 0x07
     ```

☞ The 16 kHz synchronization settings can be used for both wideband and narrowband speech.

3. Define the PCM link settings (HCI_CMD_WRITE_PCM_LINK_SETTINGS)

```
root@tk1:~# hcitool cmd 0x3F 0x0029 0x04 0x00
```

5.2.2 Command examples

5.2.2.1 NBS PCM slave using legacy SCO commands

1. Apply standard voice settings

```
root@tk1:~# hcitool cmd 0x03 0x0026 0x60 0x00
```

2. Configure PCM slave mode with 8 kHz sync

```
root@tk1:~# hcitool cmd 0x3F 0x001D 0x01
root@tk1:~# hcitool cmd 0x3F 0x0007 0x00
root@tk1:~# hcitool cmd 0x3F 0x0028 0x03 0x00 0x03
root@tk1:~# hcitool cmd 0x3F 0x0029 0x04 0x00
```

3. Configure audio codec for NBS PCM master mode

```
root@tk1:~# i2cset -y 0 0x10 0x02 0x00 0x16 0x00 0x00 0xac 0x21 0x00 0x06 0x33 0x00
   0x34 0x00 0x01 0x00 0x8a i
```

4. Start HFP call using legacy SCO commands

5.2.2.2 WBS PCM slave using enhanced SCO commands

1. Apply standard voice settings

```
root@tk1:~# hcitool cmd 0x03 0x0026 0x60 0x00
```

2. Enable host control of the PCM/I2S audio interface

```
root@tk1:~# hcitool cmd 0x3F 0x0070 0x01
```

3. Configure PCM slave mode with 16 kHz sync

```
root@tk1:~# hcitool cmd 0x3F 0x001D 0x01
root@tk1:~# hcitool cmd 0x3F 0x0007 0x00
root@tk1:~# hcitool cmd 0x3F 0x0028 0x03 0x00 0x07
root@tk1:~# hcitool cmd 0x3F 0x0029 0x04 0x00
```

4. Configure audio codec for WBS PCM master mode

```
root@tk1:~# i2cset -y 0 0x10 0x02 0x00 0x17 0x00 0x00 0xac 0x21 0x00 0x06 0x33 0x00
   0x34 0x00 0x01 0x00 0x8a i
```

5. Start HFP call using enhanced SCO commands
6. Enable audio over PCM/I2S for the SCO/eSCO connection handle

```
root@tk1:~# hcitool cmd 0x3F 0x006F 0x00 0x00 <2-byte handle little endian> 0x00
   0x00
```

Appendix

A Vendor specific HCI commands for Bluetooth audio data path and PCM/I2S

A.1 HCI_CMD_SET_SCO_DATA_PATH

This command selects between PCM/I2S and the host data path. If the audio data path is changed from the host to PCM/I2S, ensure that this command is issued before any other PCM/I2S configuration commands.

SET_SCO_DATA_PATH				
OCF	0x001D			
OGF	0x3F			
	Name	**Description**		**Size**
Command parameters	Voice Path	Selects between PCM/I2S and host data path		1 Octet
		Value	**Description**	
		0x00	Host	
		0x01	PCM/I2S	
		0x02	I2S (EMMY-W1 series only)	
		All other values	Reserved	
Return parameters	Status	Command status		1 Octet
		Value	**Description**	
		0x00	Command succeeded	
		All other values	Command failed	
Events	Command complete event returned			

A.2 HCI_CMD_WRITE_PCM_I2S_SETTINGS

This HCI command configures the following PCM/I2S settings:

- Tx/Rx port as port A or port B for PCM interface
- Master/Slave mode for PCM/I2S interface
- PCM/I2S synchronization generated from frame tick or system clock tick

The PCM clock is normally terminated after the last data bit has been transmitted. The PCM clock can be configured so that it is available continuously.

WRITE_PCM_I2S_SETTINGS				
OCF	0x0007			
OGF	0x3F			
	Name	**Description**		**Size**
Command parameters	PCM/I2S Link Settings	Configures PCM/I2S settings		1 Octet
		Bit(s)	**Description**	
		7:5	Reserved	
		4	PCM clock on • 0 = PCM clock is terminated after last data bit has been transmitted (applicable for PCM interface) • 1 = make PCM/I2S clock available continuously	
		3	Reserved	
		2	PCM synchronization source • 0 = PCM synchronization page generated from system clock • 1 = PCM synchronization page generated from frame clock	
		1	Master/Slave • 0 = PCM/I2S I/F slave, external PCM/I2S clock, and synchronization • 1 = PCM/I2S I/F master, internal PCM/I2S clock, and synchronization	
		0	PCM direction • 0 = port A receive, port B transmit • 1 = port A transmit, port B receive	
Return parameters	Status	Command status		1 Octet
		Value	**Description**	
		0x00	Command succeeded	
		All other values	Command failed	
Events	Command complete event returned			

A.3 HCI_CMD_WRITE_PCM_I2S_SYNC_SETTINGS

This command configures the following synchronization settings:

- PCM short frame synchronization or I2S justified mode
- PCM synchronization/I2S LRCLK frequency to 8 kHz or 16 kHz
- Voice data path control settings when voice is configured over the host interface

WRITE_PCM_I2S_SYNC_SETTINGS				
OCF	0x0028			
OGF	0x3F			
	Name	**Description**		**Size**
Command parameters	Sync settings 1	Controls how voice samples are moved between memory and receive/transmit buffers in hardware. It is advisable to set these bits to 1 for optimal performance (Default = 0x03).		1 Octet
		Bit(s)	**Description**	
		7:2	Reserved	
		1	ISS (IramSyncSource) only valid if IF = host and ISR = Fixed Rate • 0 = ISR not aligned to frame tick • 1 = ISR aligned to frame tick (set to 1)	
		0	ISR (IramSyncRate) only valid if IF = host • 0 = bursts controlled by Tx/Rx of voice packets • 1 = fixed rate of 8 ksamples/s	
	Sync settings 2	Configures PCM/I2S synchronization settings (Default = 0x0300 (PCM interface))		2 Octets
		Bit(s)	**Description**	
		10	16k synchronization in PCM/I2S • 0 = 8k synchronization/I2S LRCLK • 1 = 16k synchronization/I2S LRCLK	
		9	Clock setting • 0 = select 2 MHz • 1 = select 2.048 MHz clock for PCM/I2S clock	
		8	Main clock • 0 = disable generation of PCM/I2S main clock • 1 = enable	
		7:5	Reserved	
		4	I2S data in left or right channel (applicable only in I2S master mode) • 0 = data sent in right channel • 1 = data sent in left channel	
		3	Clock alignment for I2S DIN data sampling (applicable only in I2S mode) • 0 = data is sampled on falling edge of interface clock • 1 = data is sampled on rising edge of interface clock	
		2	Clock alignment for I2S DOUT data driven (applicable only in I2S mode) • 0 = data is driven on rising edge of interface clock • 1 = data is driven on falling edge of interface clock	
		1:0	Mode • 00 = PCM short synchronization • 01 = PCM long synchronization (EMMY-W1/JODY-W2 only) • 10 = I2S justified audio mode	
Return parameters	Status	Command status		1 Octet
		Value	**Description**	
		0x00	Command succeeded	
		All other values	Command failed	
Events	Command complete event returned			

A.4 HCI_CMD_WRITE_PCM_LINK_SETTINGS

This command configures the PCM link settings, used to control the PCM slots for each SCO/eSCO link. The command should be issued after the HCI reset and before setting up the SCO links.

WRITE_PCM_LINK_SETTINGS				
OCF	0x0029			
OGF	0x3F			
	Name	**Description**		**Size**
Command parameters	PCM Link settings	Configures PCM link settings. Default: • 0x0004 = 1st SCO link (PCM logical slot = 0, PCM start slot = 1) • 0x0045 = 2nd SCO link (PCM logical slot = 1, PCM start slot = 17 (0x11)) • 0x0086 = 3rd SCO link (PCM logical slot = 2, PCM start slot = 33 (0x21))		2 Octets
		Bit(s)	**Description**	
		15:12	Reserved	
		13:10	Each bit corresponds to 1 of 4 PCM timeslots. If set to 0, the slot is used by Bluetooth.	
		9:2	PCM slot relative to start of PCM synchronization	
		1:0	PCM slots to be used • 00 = PCM logical slot 0 • 01 = PCM logical slot 1 • 10 = PCM logical slot 2 • 11 = PCM logical slot 3	
Return parameters	Status	Command status		1 Octet
		Value	**Description**	
		0x00	Command succeeded	
		All other values	Command failed	
Events	Command complete event returned			

A.5 HCI_CMD_SET_I2S_SETTINGS

☞ Only applicable for EMMY-W1 series modules with the SCO data path set to I2S (0x02).

This command configures the following I2S settings:

- I2S justified mode
- I2S master/slave role
- LRCLK and BCLK inversion
- I2S slave sync/async mode
- Selects internal or external clock for I2S slave

SET_I2S_SETTINGS				
OCF	0x0079			
OGF	0x3F			
	Name	**Description**		**Size**
Command parameters	I2S settings			2 Octets
		Bit(s)	**Description**	
		8	Generate CCLK • 0 = Disable CCLK generation • 1 = Enable CCLK generation (applicable for I2S master and slave using internal clock)	
		7	Slave external clock invert • 0 = external CCLK not inverted • 1 = external CCLK inverted	
		6	Slave internal clock • 0 = use external CCLK • 1 = use internal CCLK	
		5	Slave sync mode • 0 = Slave async mode • 1 = Slave sync mode	
		4	I2S role • 0 = Slave • 1 = Master	
		3	BCLK invert • 0 = no inversion • 1 = BCLK inverted (applicable only for I2S master)	
		2	LRCLK invert • 0 = no inversion • 1 = LRCLK inverted (applicable only for I2S master)	
		1:0	Mode • 00 = I2S mode	
Return parameters	Status	Command status		1 Octet
		Value	**Description**	
		0x00	Command succeeded	
		All other values	Command failed	
Events	Command complete event returned			

A.6 HCI_CMD_SET_I2S_LINK_SETTINGS

☞ Only applicable for EMMY-W1 series modules with the SCO data path set to I2S (0x02).

This command configures the sample rate for the LRCLK and bus width for I2S.

SET_I2S_LINK_SETTINGS				
OCF	0x007A			
OGF	0x3F			
	Name	**Description**		**Size**
Command parameters	LRCLK			1 Octet
		Value	**Description**	
		0x00	8 kHz	
		0x01	16 kHz	
	Bits per Channel			1 Octet
		Value	**Description**	
		0x10	16 bits	
		0x20	32 bits	
Return parameters	Status	Command status		1 Octet
		Value	**Description**	
		0x00	Command succeeded	
		All other values	Command failed	
Events	Command complete event returned			

In I2S master mode, BCLK is configured using the “Bits per Channel” and LRCLK parameters as follows: BCLK = 2 * Bits per Channel * LRCLK

The configuration for NBS with LRCLK = 8 kHz is as follows:

- Bits per Channel = 0x10 (16) = 256 kHz BCLK
- Bits per Channel = 0x20 (32) = 512 kHz BCLK

The configuration for WBS with LRCLK = 16 kHz is as follows:

- Bits per Channel = 0x10 (16) = 512 kHz BCLK
- Bits per Channel = 0x20 (32) = 1024 kHz BCLK

A.7 HCI_CMD_SET_WBS_CONNECTION

☞ Legacy command to enable wideband speech. Not supported on JODY-W3 series modules.

This command enables or disables wideband speech for the next SCO/eSCO link and informs the controller to use the mSBC codec. This command must be called each time before creating the SCO/eSCO connection for WBS when using legacy SCO commands:

- `HCI_Setup_Synchronous_Connection`
- `HCI_Accept_Synchronous_Connection_Request`

SET_WBS_CONNECTION				
OCF	**0x0073**			
OGF	**0x3F**			
	Name	**Description**		**Size**
Command parameters	Next_SCO_-Connection_-WBS	Enables or disables WBS for next SCO connection		1 Octet
		Value	**Description**	
		0x00	Disable WBS for next SCO connection	
		0x01	Enable WBS for next SCO connection	
Return parameters	Status	Command status		1 Octet
		Value	**Description**	
		0x00	Command succeeded	
		All other values	Command failed	
Events	Command complete event returned			

A.8 HCI_CMD_HOST_PCM_I2S_AUDIO_CONFIG

This command initializes and configures the PCM/I2S interface. The command is sent in the following instances:

- For single HFP call scenario:
  - To enable audio over PCM/I2S after HFP call is started
  - To disable audio over PCM/I2S for an active HFP call

☞ Dual HFP call scenario implementation is currently not supported.

HOST_PCM_I2S_AUDIO_CONFIG				
OCF	0x006F			
OGF	0x3F			
	Name	**Description**		**Size**
Command parameters	Action			1 Octet
		Value	**Description**	
		0x00	Start audio on PCM/I2S	
		0x01	Stop audio on PCM/I2S	
	Operation mode			1 Octet
		Value	**Description**	
		0x00	Normal mode (This mode is used when only 1 voice call needs to be active at a time)	
		Others	Reserved	
	SCO handle 1	eSCO connection handle		2 Octets
	SCO handle 2	eSCO connection handle (This parameter is valid only if 2 eSCO connections are present)		2 Octets
Return parameters	Status	Command status		1 Octet
		Value	**Description**	
		0x00	Command succeeded	
		All other values	Command failed	
Events	Command complete event returned			

A.9 HCI_CMD_HOST_PCM_I2S_CONTROL_ENABLE

This command enables or disables control of the audio interface through the host software (Bluetooth stack). It should be sent when the host needs to control the PCM/I2S interface and if multiple HFP calls are required at the same time. The command signals to the controller that the host software provides the PCM/I2S configuration. The following standard Bluetooth HCI commands are to be used in conjunction with this command:

- `HCI_Enhanced_Setup_Synchronous_Connection`
- `HCI_Enhanced_Accept_Synchronous_Connection_Request`

HOST_PCM_I2S_CONTROL_ENABLE				
OCF	0x0070			
OGF	0x3F			
	Name	**Description**		**Size**
Command parameters	Action	Enables or disables host software control over PCM/I2S		1 Octet
		Value	**Description**	
		0x00	The controller manages the PCM/I2S lines (default)	
		0x01	The host software manages the PCM/I2S lines	
Return parameters	Status	Command status		1 Octet
		Value	**Description**	
		0x00	Command succeeded	
		All other values	Command failed	
Events	Command complete event returned			

A.10 Enhanced SCO commands – vendor specific parameters for NBS/WBS

The enhanced SCO setup and accept commands have multiple parameters that define the audio settings, including some vendor specific PCM data formats and data size parameters. These parameters must be set correctly for NBS and WBS audio call setup. Table 4 shows the required parameter values for NBS and WBS audio calls for use with the following standard Bluetooth HCI commands:

- `HCI_Enhanced_Setup_Synchronous_Connection`
- `HCI_Enhanced_Accept_Synchronous_Connection_Request`

Other parameters, such as bandwidth, maximum latency, packet type, and re-transmission effort can be set as recommended in the HFP specification [3].

☞ The use of enhanced SCO commands is mandatory for supporting wideband speech on JODY-W3 series modules and must be used in conjunction with the commands: `HCI_CMD_HOST_PCM_I2S_CONTROL_ENABLE` and `HCI_CMD_HOST_PCM_I2S_AUDIO_CONFIG`

Enhanced SCO command parameters	NBS	WBS
Transmit Coding Format	0x00 = µ-law 0x01 = A-law 0x02 = CVSD 0x03 = reserved	0x05 = mSBC
Receive Coding Format		
Transmit Codec Frame Size	0x3C	0x3C
Receive Codec Frame Size		
Input Coded Data Size	0x10	0x08
Output Coded Data Size		
Input Coding Format	0x04	0x04
Output Coding Format		
Input PCM Data Format	0x02	0x02
Output PCM Data Format		
Input PCM Sample Payload MSB Position	0x00	0x00
Output PCM Sample Payload MSB Position		
Input Datapath	0x01	0x01
Output Datapath		
Input Transport Unit Size	0x10	0x10
Output Transport Unit Size		

Table 4: Vendor specific enhanced SCO command parameters

B PulseAudio patches to enable wideband speech

```
diff --git a/src/modules/bluetooth/backend-ofono.c b/src/modules/bluetooth/backend-
   ofono.c
index 755df9e..ba3e9d0 100644
--- a/src/modules/bluetooth/backend-ofono.c
+++ b/src/modules/bluetooth/backend-ofono.c
@@ -384,6 +384,7 @@ static void hf_audio_agent_register(pa_bluetooth_backend *hf) {
     pa_assert_se(m = dbus_message_new_method_call(OFONO_SERVICE, "/",
   HF_AUDIO_MANAGER_INTERFACE, "Register"));

     codecs[ncodecs++] = HFP_AUDIO_CODEC_CVSD;
+    codecs[ncodecs++] = HFP_AUDIO_CODEC_MSBC;

     pa_assert_se(dbus_message_append_args(m, DBUS_TYPE_OBJECT_PATH, &path,
   DBUS_TYPE_ARRAY, DBUS_TYPE_BYTE, &pcodecs, ncodecs,
                                           DBUS_TYPE_INVALID));
@@ -529,7 +530,7 @@ static DBusMessage *hf_audio_agent_new_connection(DBusConnection *c,
   DBusMessage

     card = pa_hashmap_get(backend->cards, path);

-    if (!card || codec != HFP_AUDIO_CODEC_CVSD || card->transport->state ==
   PA_BLUETOOTH_TRANSPORT_STATE_PLAYING) {
+    if (!card || (codec != HFP_AUDIO_CODEC_CVSD && codec != HFP_AUDIO_CODEC_MSBC) ||
   card->transport->state == PA_BLUETOOTH_TRANSPORT_STATE_PLAYING) {
         pa_log_warn("New audio connection invalid arguments (path=%s fd=%d, codec=%d)",
   path, fd, codec);
         pa_assert_se(r = dbus_message_new_error(m, "org.ofono.Error.InvalidArguments",
   "Invalid arguments in method call"));
         return r;
diff --git a/src/modules/bluetooth/module-bluez5-device.c
   b/src/modules/bluetooth/module-bluez5-device.c
index 065fcaa..15ac9ac 100644
--- a/src/modules/bluetooth/module-bluez5-device.c
+++ b/src/modules/bluetooth/module-bluez5-device.c
@@ -56,6 +56,9 @@ PA_MODULE_VERSION(PACKAGE_VERSION);
 PA_MODULE_LOAD_ONCE(false);
 PA_MODULE_USAGE("path=<device object path>");

+#define HFP_AUDIO_CODEC_CVSD    0x01
+#define HFP_AUDIO_CODEC_MSBC    0x02
+
 #define MAX_PLAYBACK_CATCH_UP_USEC (100 * PA_USEC_PER_MSEC)
 #define FIXED_LATENCY_PLAYBACK_A2DP (25 * PA_USEC_PER_MSEC)
 #define FIXED_LATENCY_PLAYBACK_SCO (125 * PA_USEC_PER_MSEC)
@@ -1149,7 +1152,7 @@ static void transport_config(struct userdata *u) {
     if (u->profile == PA_BLUETOOTH_PROFILE_HEADSET_HEAD_UNIT || u->profile ==
   PA_BLUETOOTH_PROFILE_HEADSET_AUDIO_GATEWAY) {
         u->sample_spec.format = PA_SAMPLE_S16LE;
         u->sample_spec.channels = 1;
-        u->sample_spec.rate = 8000;
+        u->sample_spec.rate = (u->transport->codec == HFP_AUDIO_CODEC_CVSD) ? 8000 :
   16000;
     } else {
         sbc_info_t *sbc_info = &u->sbc_info;
         a2dp_sbc_t *config;
```

C Glossary

Abbreviation	Definition
A2DP	Advanced Audio Distribution Profile
ACL	Asynchronous Connection-oriented [logical transport]
AG	Audio Gateway
EVB	Evaluation Board
EVK	Evaluation Kit
HCI	Host Controller Interface
HFP	Hands-Free Profile
HF	Hands-Free unit
I2C	Inter-Integrated Circuit
I2S	Inter-IC Sound
NBS	Narrowband Speech
PCM	Pulse-Code Modulation
SCO/eSCO	(Extended) Synchronous Connection Oriented [logical transport]
SDIO	Secure Digital Input Output
UART	Universal Asynchronous Receiver-Transmitter
WBS	Wideband Speech

Table 5: Explanation of the abbreviations and terms used

Related documentation

[1] JODY-W2 system integration manual, UBX-18068879
[2] JODY-W3 system integration manual, UBX-19011209
[3] Bluetooth specifications list, https://www.bluetooth.com/specifications/profiles-overview/
[4] BlueZ, http://www.bluez.org/
[5] PulseAudio, https://www.freedesktop.org/wiki/Software/PulseAudio/Documentation/User/Bluetooth/
[6] oFono, https://01.org/ofono
[7] MAX9860, 16-Bit Mono Audio Voice Codec, https://www.maximintegrated.com/en/products/analog/audio/MAX9860.html
[8] Toradex Ixora Carrier Board, https://www.toradex.com/products/carrier-board/ixora-carrier-board
[9] EVK-JODY-W2 user guide, UBX-19027118
[10] EVK-JODY-W3 user guide, UBX-20030840
[11] EVK-MAYA-W1 user guide, UBX-21039658
[12] Apalis TK1 data sheet, https://docs.toradex.com/103129-apalis-tk1-datasheet.pdf

☞ For product change notifications and regular updates of u-blox documentation, register on our website, www.u-blox.com.

Revision history

Revision	Date	Name	Comments
R01	09-Nov-2021	mzes	Initial release
R02	20-May-2022	mzes	Added description for EVK-MAYA-W1. Removed I2S support for JODY-W2 in PCM/I2S interface. Added Command examples.

Contact

For further support and contact information, visit us at www.u-blox.com/support.