Intel® Edison

Audio Setup Guide

May 2015

Revision 001

Document Number: 332434-001

Notice: This document contains information on products in the design phase of development. The information here is subject to change without notice. Do not finalize a design with this information.

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A “Mission Critical Application” is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked “reserved” or “undefined.” Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

Intel software products are copyrighted by and shall remain the property of Intel Corporation. Use, duplication, or disclosure is subject to restrictions stated in Intel's Software License Agreement, or in the case of software delivered to the government, in accordance with the software license agreement as defined in FAR 52.227-7013.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

The code names presented in this document are only for use by Intel to identify products, technologies, or services in development that have not been made commercially available to the public, i.e., announced, launched, or shipped. They are not "commercial" names for products or services and are not intended to function as trademarks.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature may be obtained by calling 1-800-548-4725 or by visiting Intel's website at *http://www.intel.com/design/literature.htm*.

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. See *http://www.intel.com/products/processor_number* for details.

Intel and the Intel logo are trademarks of Intel Corporation in the United States and other countries.

* Other brands and names may be claimed as the property of others.

Copyright © 2015 Intel Corporation. All rights reserved.

Contents

Figures

Revision History

Revision	Description	Date
001	Initial release.	May 1, 2015

§

1 Introduction

This document provides information about the audio subsystem on the Intel® Edison platform. It also provides hardware/software overviews, shows different ways to capture/playback audio through the hardware with an Intel® Edison board, shows how to play and record audio using Intel® Edison, and describes software stacks to develop audio applications or play with audio on the Intel® Edison platform.

1.1 References

Reference	Name	Number/location
331188	Intel® Edison Board Support Package User Guide	*http://www.intel.com/support/edison/sb/CS-035278.htm*
331189	Intel® Edison Compute Module Hardware Guide	*http://www.intel.com/support/edison/sb/CS-035274.htm*
331190	Intel® Edison Breakout Board Hardware Guide	*http://www.intel.com/support/edison/sb/CS-035252.htm*
331191	Intel® Edison Kit for Arduino* Hardware Guide	*http://www.intel.com/support/edison/sb/CS-035275.htm*
331192	Intel® Edison Native Application Guide	*http://www.intel.com/support/edison/sb/CS-035382.htm*
329686	Intel® Galileo and Intel® Edison Release Notes	*https://communities.intel.com/docs/DOC-23388*
332032	Intel® Edison Software Release Notes	
[GSG]	Intel® Edison Getting Started Guide	W: *https://communities.intel.com/docs/DOC-23147* M: *https://communities.intel.com/docs/DOC-23148* L: *https://communities.intel.com/docs/DOC-23149*
331438	Intel® Edison Wi-Fi Guide	*http://www.intel.com/support/edison/sb/CS-035380.htm*
331704	Intel® Edison Bluetooth* Guide	*http://www.intel.com/support/edison/sb/CS-035381.htm*
332434	Intel® Edison Audio Setup Guide	(This document)
[YPQSG]	Yocto Project Quick Start Guide	*http://www.yoctoproject.org/docs/current/yocto-project-qs/yocto-project-qs.html*
[YDM]	Yocto Developer Manual	*http://www.yoctoproject.org/docs/current/dev-manual/dev-manual.html*
[YKDM]	Yocto Kernel Developer Manual	*http://www.yoctoproject.org/docs/latest/kernel-dev/kernel-dev.html*

1.2 Terminology

Term	Definition
A2DP	Advanced Audio Distribution Profile
BT	Bluetooth
DSP	Digital signal processing.
HFP	Hands-Free Profile
HSP	Headset Profile
PCM	Pulse-code modulation
TDM	Time division multiplexing

§

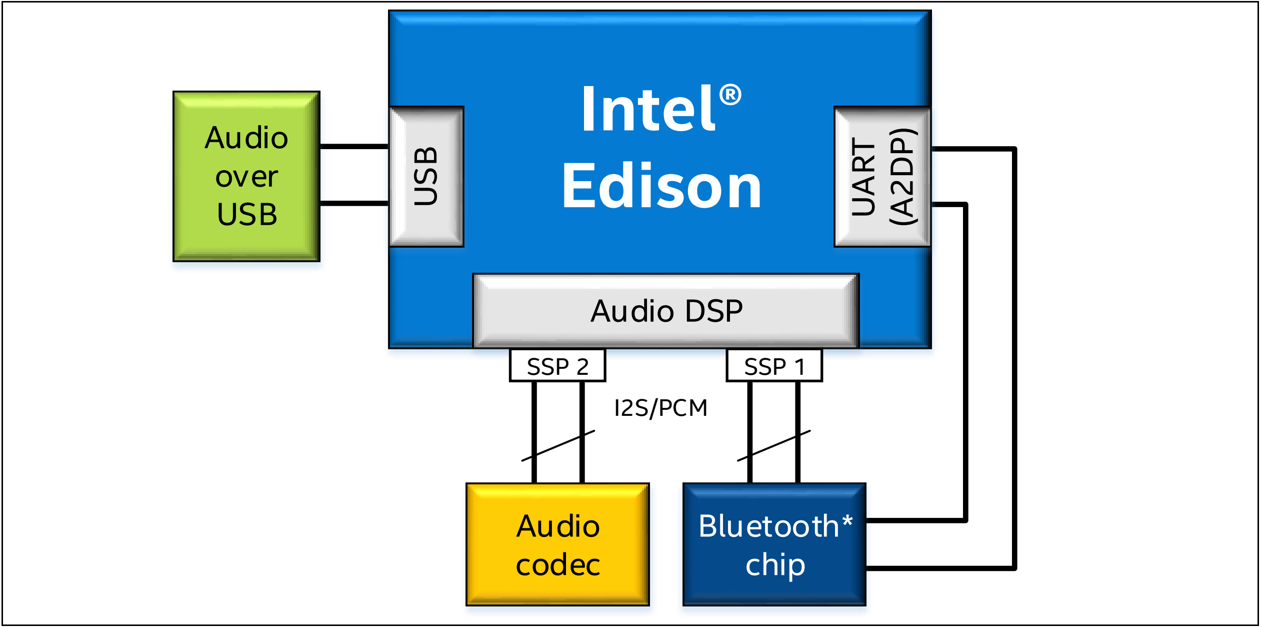
2 Hardware

The Intel® Edison development board has hardware capabilities—Bluetooth*, USB, and I2S I/O—that allow you to enable the following audio features:

- Audio over a USB device.
- Audio record from A2DP (sink) remote player.
- Audio playback to an A2DP remote device.
- Local audio playback/record by adding a hardware audio codec + microphone, speakers, or jack headset.

Figure 1 shows the audio architecture of the Intel® Edison development board.

Figure 1 Intel® Edison development board's audio hardware architecture



2.1 Hardware specifications

The Intel® Edison controller has two 4-wire ports dedicated to I2S/PCM audio transmission: **SSP1** and **SSP2**.

The SSP1 port is used by the Bluetooth* processor for the hands-free and headset audio profiles (HFP and HSP). It can also be used optionally by the UART for audio transmission. The SSP2 port is available for audio codec hardware.

Both I2S and PCM mode can be used with the audio DSP to record and play audio on an Intel® Edison board. The audio DSP has to be the **master** of the bus.

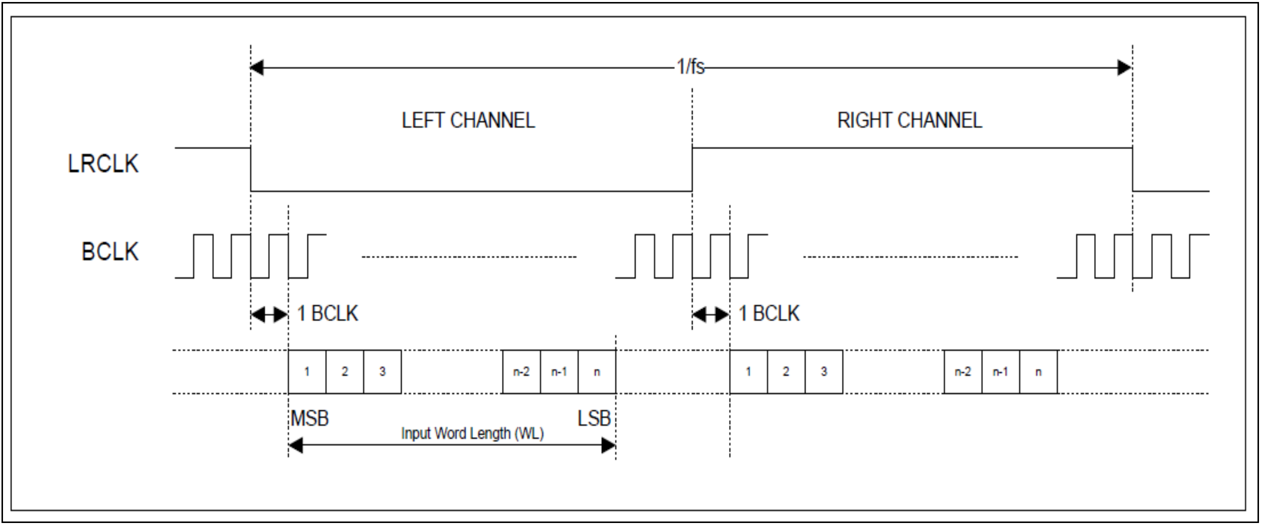
2.1.1 I2S Mode

I2S is a 4 wire digital audio interface (DAI) used in high-fidelity sound equipment, set-top boxes, and various portable devices. For more information, see *https://www.kernel.org/doc/Documentation/sound/alsa/soc/DAI.txt*.

I2S operating modes:

- I2S - MSB
- Left-justified
- Right-justified

Figure 2 Example for I2S left-justified



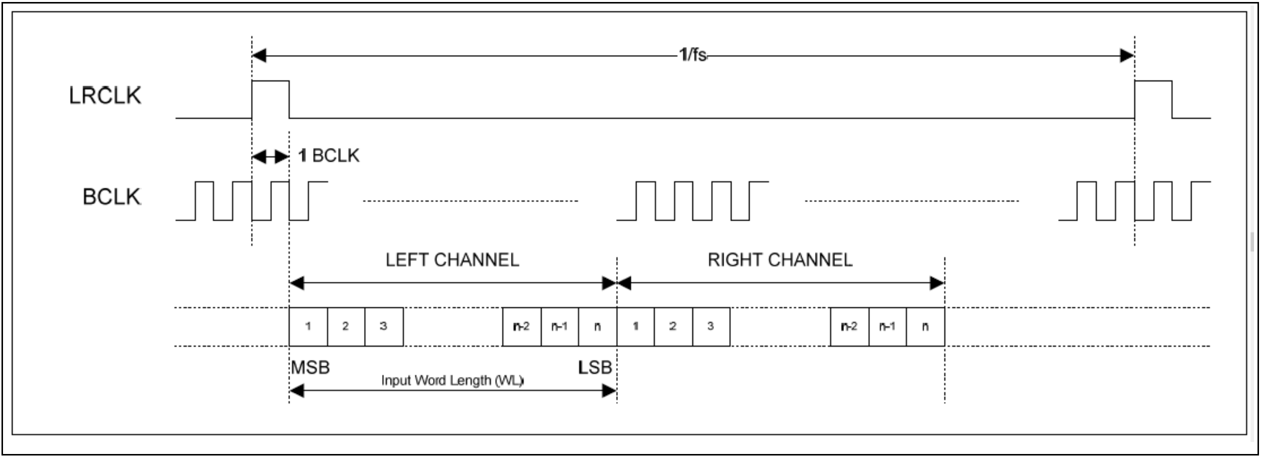
2.1.2 PCM mode

PCM is also a 4-wire interface, similar to I2S, that can support a flexible protocol. For more information, see *https://www.kernel.org/doc/Documentation/sound/alsa/soc/DAI.txt*.

PCM operating modes:

- Mode A – MSB (most significant bit) is transmitted on falling edge of first BCLK after FRAME/SYNC.
- Mode B – MSB (most significant bit) is transmitted on rising edge of FRAME/SYNC.

Figure 3 Example PCM mode A



2.1.3 Hardware/software capabilities and limitations

The following restrictions apply:

- The audio DSP has to be the master of the bus.
- Not all configurations have been tested.
- The SSPx port supports I2S/PCM/PCM_TDM configurations (1 to 8 slots).

Hardware-supported audio configurations:

- Mode: I2S, PCM, TDM
- Sampling rate: 8, 16, 44.1, and 48 kHz
- Number of bits by channel: 16, 24, or 32 bits
- Number of channels: mono, dual mono, stereo; In TDM mode: 1 to 8 slots

Software-supported configuration on SSP2:

- Mode: TDM
- Sampling rate: 48 kHz
- Bits by channel: 24 bits
- Number of channels: 4 slots

SSP1 port tested configuration:

- I2S/PCM
- 8/16 kHz
- 16 bits
- Mono/dual mono/stereo

SSP2 port tested configuration:

- PCM TDM 4 slots
- 48 kHz
- 24 bits
- Mono/dual mono/stereo

§

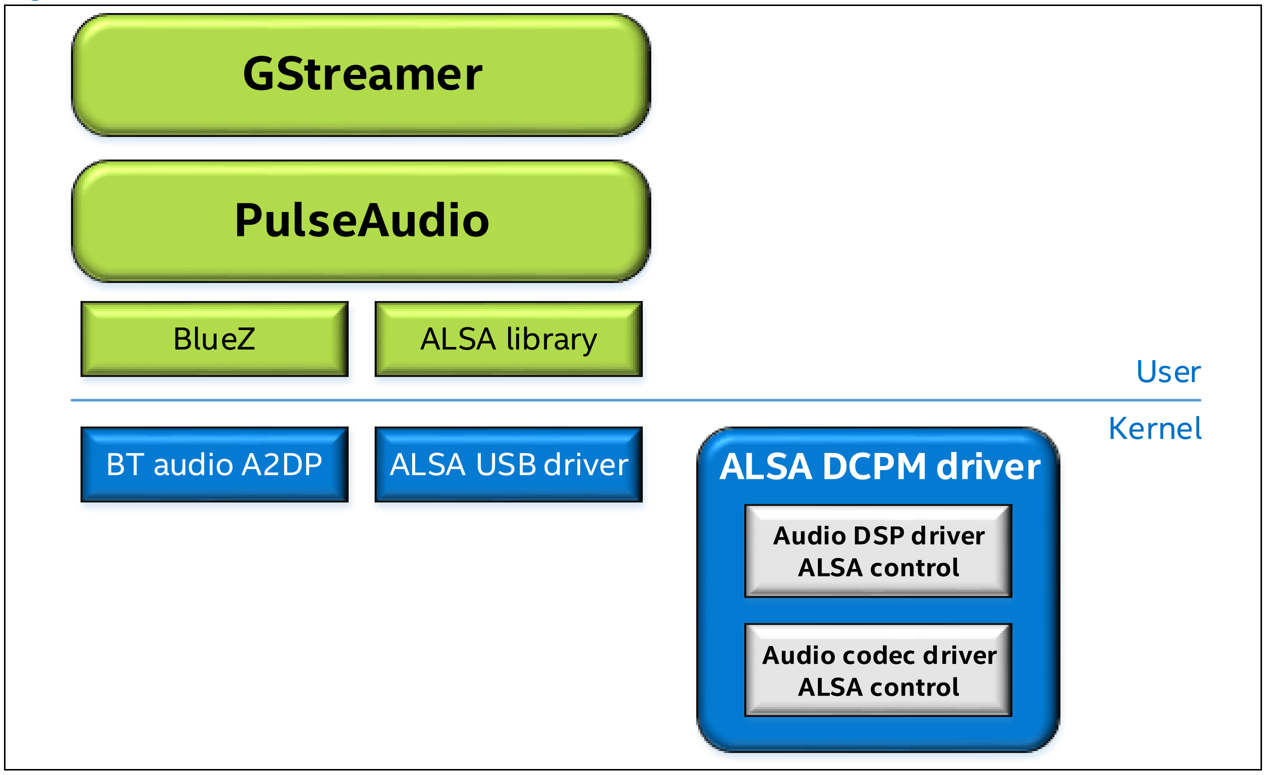
3 Software

The Intel® Edison prebuilt image supports the following libraries and tools that allow an Intel® Edison board to play back or record audio out of the box:

- **GStreamer**: A player that allows Intel® Edison to play WAV files.
- **PulseAudio:** A server manages the audio source/sink device.
- **BlueZ**: A Bluetooth* library on Linux*.
- **Alsa-lib:** The Linux* audio library.

Figure 4 shows the Intel® Edison audio software architecture.

Figure 4 Audio software architecture



3.1 GStreamer

GStreamer is the default player supported on the Intel® Edison board. It acts like a framework to develop/create streaming media applications. Designed to make it easy for applications to handle audio and video, GStreamer's framework makes it possible to develop any type of streaming multimedia applications.

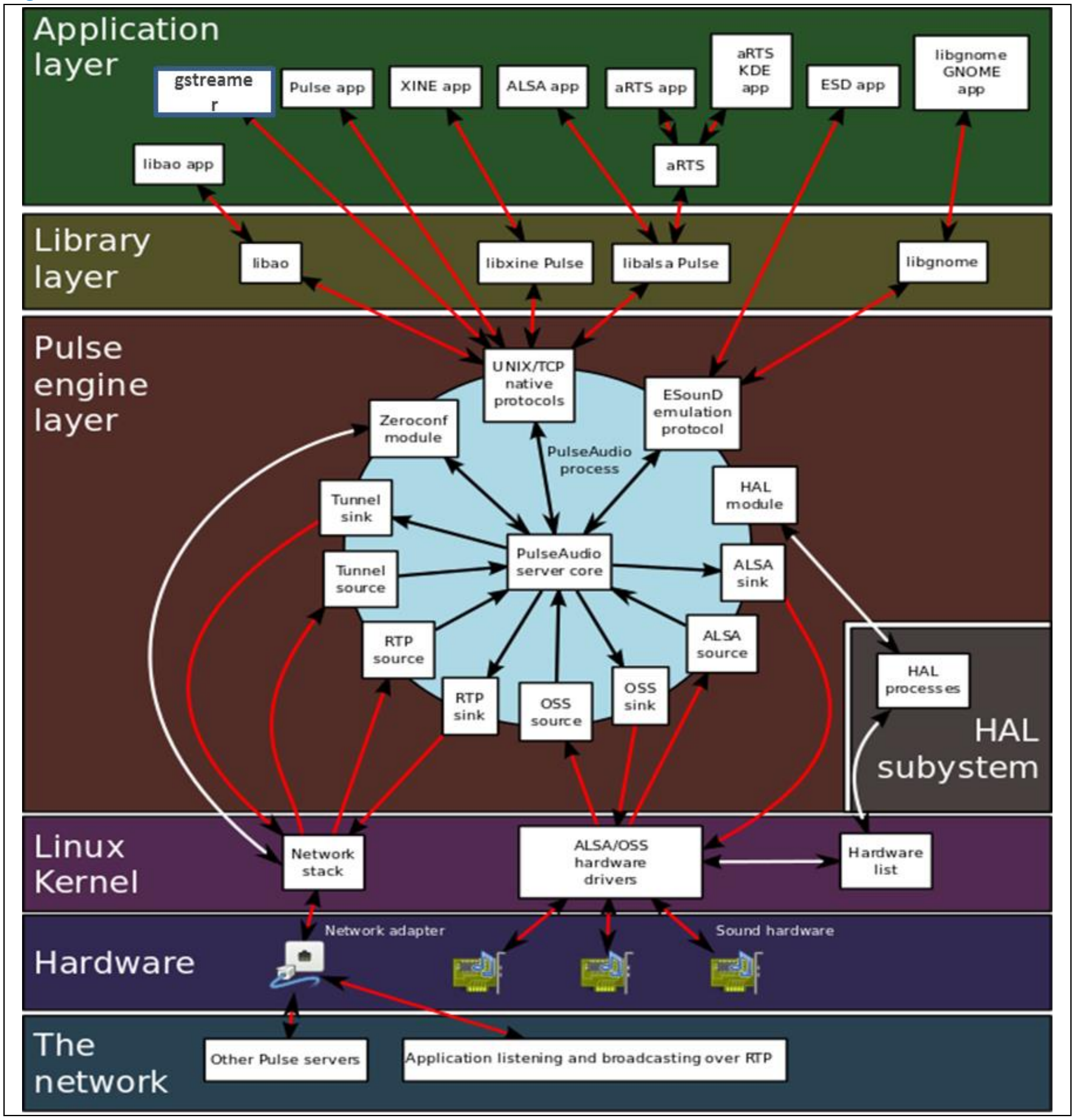
GStreamer already includes components for building a media player that can support a very wide variety of formats, including MP3, Ogg/Vorbis, MPEG-1/2, AVI, and QuickTime*. GStreamer supports a wide variety of codecs via plugins. The default Intel® Edison image currently ships with WAV, Vorbis, and FLAC audio formats.

Note: See *http://gstreamer.freedesktop.org* for more information.

3.2 PulseAudio server

PulseAudio is a POSIX-based sound server, designed for Linux* but also ported to other OSes. It features a background process that accepts sound input from one or more **sources** (input/capture devices) and redirects it to one or more **sinks** (output/playback devices).

Figure 5 PulseAudio architecture[1]

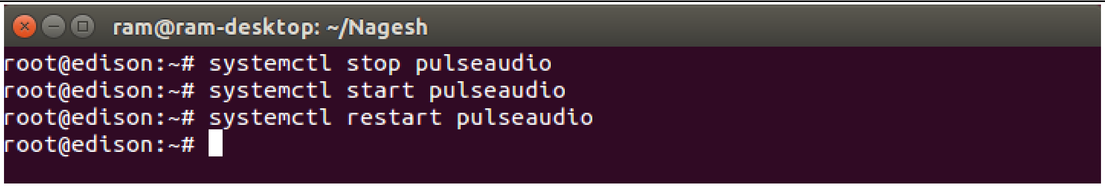


[1] *Diagram by Manuel Amador. Reprinted by permission.*

3.2.1 Procedure to use PulseAudio

The PulseAudio server launches when the Intel® Edison board boots up. You can stop/restart/start PulseAudio like any *systemd* service (Figure 6).

Figure 6 Pulseaudio start, stop, and restart operations

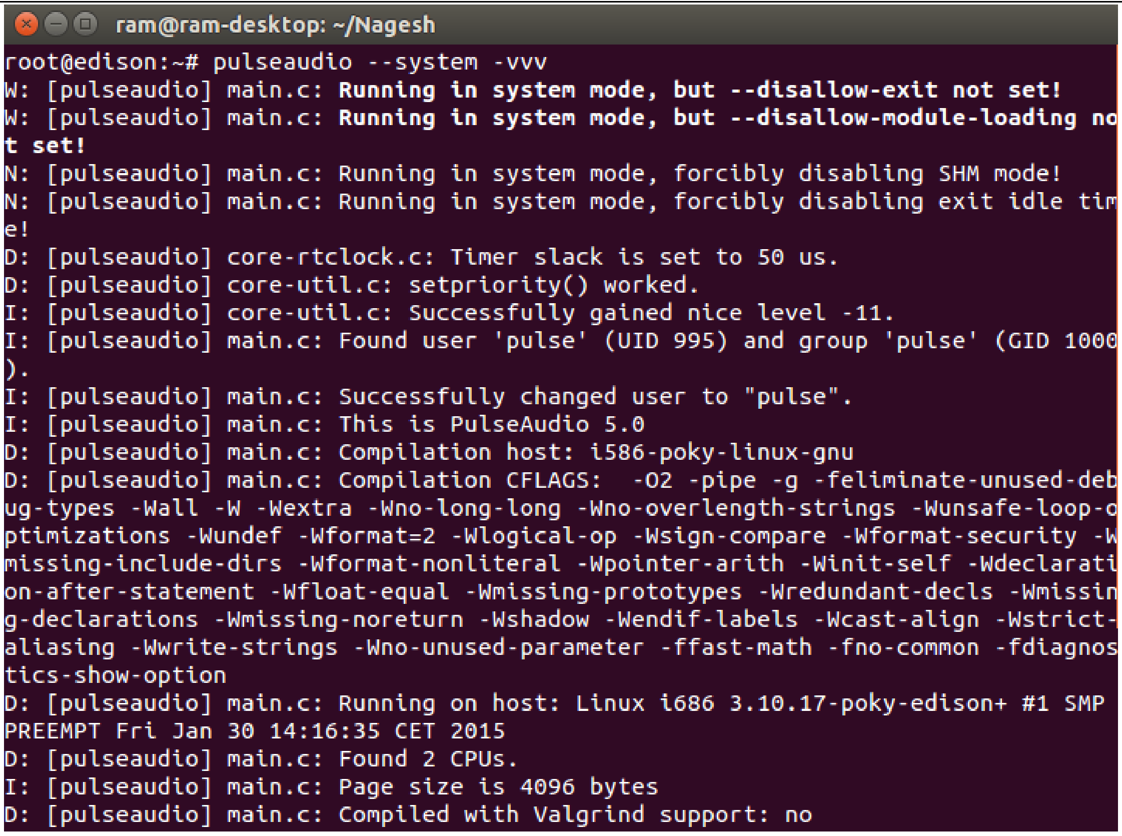


ram@ram-desktop: ~/Nagesh

```
root@edison:~# systemctl stop pulseaudio
root@edison:~# systemctl start pulseaudio
root@edison:~# systemctl restart pulseaudio
root@edison:~# 
```

We can also launch it by command line and add debug information with `-v` parameters (Figure 7).

Figure 7 PulseAudio start debug log enabled



ram@ram-desktop: ~/Nagesh

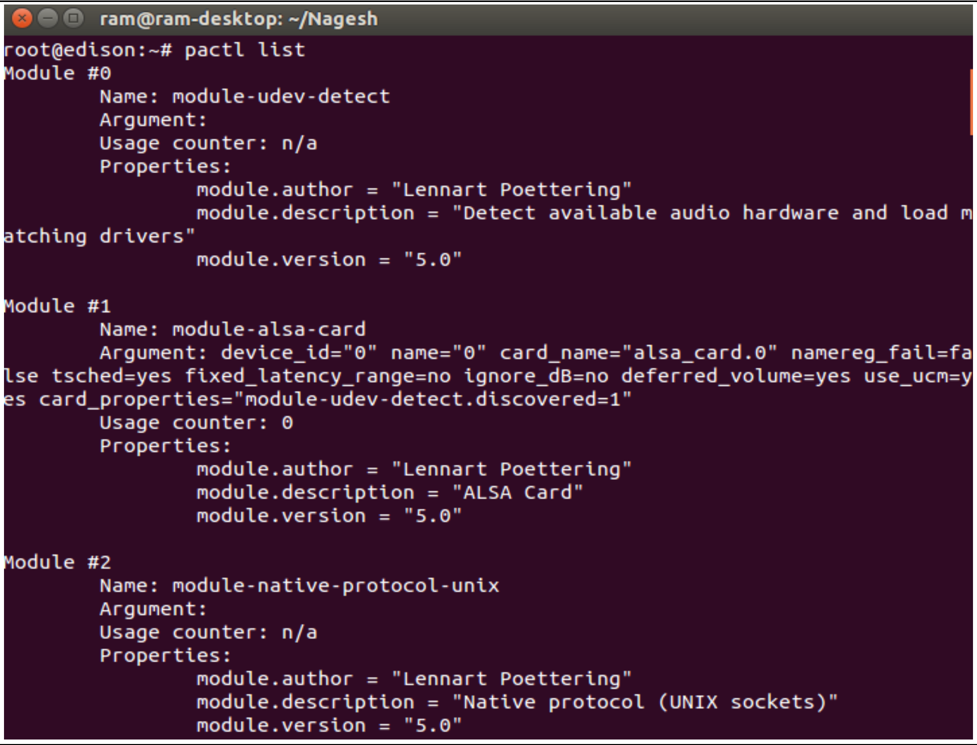
```
root@edison:~# pulseaudio --system -vvv
W: [pulseaudio] main.c: Running in system mode, but --disallow-exit not set!
W: [pulseaudio] main.c: Running in system mode, but --disallow-module-loading no
t set!
N: [pulseaudio] main.c: Running in system mode, forcibly disabling SHM mode!
N: [pulseaudio] main.c: Running in system mode, forcibly disabling exit idle tim
e!
D: [pulseaudio] core-rtclock.c: Timer slack is set to 50 us.
D: [pulseaudio] core-util.c: setpriority() worked.
I: [pulseaudio] core-util.c: Successfully gained nice level -11.
I: [pulseaudio] main.c: Found user 'pulse' (UID 995) and group 'pulse' (GID 1000
).
I: [pulseaudio] main.c: Successfully changed user to "pulse".
I: [pulseaudio] main.c: This is PulseAudio 5.0
D: [pulseaudio] main.c: Compilation host: i586-poky-linux-gnu
D: [pulseaudio] main.c: Compilation CFLAGS:  -O2 -pipe -g -feliminate-unused-deb
ug-types -Wall -W -Wextra -Wno-long-long -Wno-overlength-strings -Wunsafe-loop-o
ptimizations -Wundef -Wformat=2 -Wlogical-op -Wsign-compare -Wformat-security -W
missing-include-dirs -Wformat-nonliteral -Wpointer-arith -Winit-self -Wdeclarati
on-after-statement -Wfloat-equal -Wmissing-prototypes -Wredundant-decls -Wmissin
g-declarations -Wmissing-noreturn -Wshadow -Wendif-labels -Wcast-align -Wstrict-
aliasing -Wwrite-strings -Wno-unused-parameter -ffast-math -fno-common -fdiagnos
tics-show-option
D: [pulseaudio] main.c: Running on host: Linux i686 3.10.17-poky-edison+ #1 SMP
PREEMPT Fri Jan 30 14:16:35 CET 2015
D: [pulseaudio] main.c: Found 2 CPUs.
I: [pulseaudio] main.c: Page size is 4096 bytes
D: [pulseaudio] main.c: Compiled with Valgrind support: no
```

As a unique "root" user of the Intel® Edison image, you must launch PulseAudio with the `--system` parameter. You can use the `-h` option to get more parameter capabilities, like `--dump-resample-methods`, to optimize further desired configurations.

When PulseAudio starts, you may choose to display all your source and sink devices with the **pactl** binary (**P**ulse**A**udio **Co**n**t**ro**l**) to view their capabilities (Figure 8). Do the following:

1. Enter *pactl list*.

Figure 8 pactl list command



ram@ram-desktop: ~/Nagesh

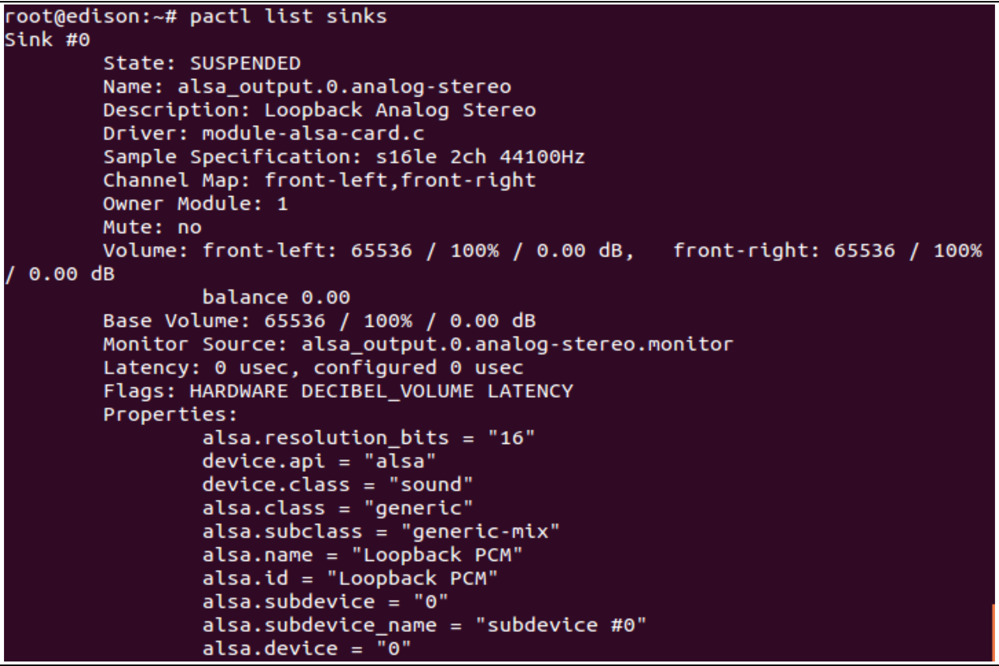
```
root@edison:~# pactl list
Module #0
        Name: module-udev-detect
        Argument:
        Usage counter: n/a
        Properties:
                module.author = "Lennart Poettering"
                module.description = "Detect available audio hardware and load m
atching drivers"
                module.version = "5.0"

Module #1
        Name: module-alsa-card
        Argument: device_id="0" name="0" card_name="alsa_card.0" namereg_fail=fa
lse tsched=yes fixed_latency_range=no ignore_dB=no deferred_volume=yes use_ucm=y
es card_properties="module-udev-detect.discovered=1"
        Usage counter: 0
        Properties:
                module.author = "Lennart Poettering"
                module.description = "ALSA Card"
                module.version = "5.0"

Module #2
        Name: module-native-protocol-unix
        Argument:
        Usage counter: n/a
        Properties:
                module.author = "Lennart Poettering"
                module.description = "Native protocol (UNIX sockets)"
                module.version = "5.0"
```

2. Enter *pactl list sinks*.

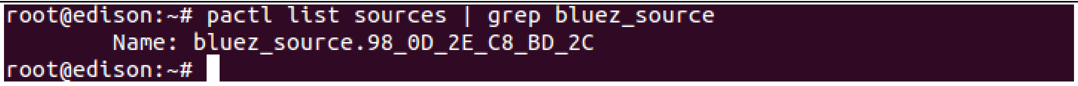
Figure 9 pactl list sinks command



```
root@edison:~# pactl list sinks
Sink #0
        State: SUSPENDED
        Name: alsa_output.0.analog-stereo
        Description: Loopback Analog Stereo
        Driver: module-alsa-card.c
        Sample Specification: s16le 2ch 44100Hz
        Channel Map: front-left,front-right
        Owner Module: 1
        Mute: no
        Volume: front-left: 65536 / 100% / 0.00 dB,   front-right: 65536 / 100%
/ 0.00 dB
                balance 0.00
        Base Volume: 65536 / 100% / 0.00 dB
        Monitor Source: alsa_output.0.analog-stereo.monitor
        Latency: 0 usec, configured 0 usec
        Flags: HARDWARE DECIBEL_VOLUME LATENCY
        Properties:
                alsa.resolution_bits = "16"
                device.api = "alsa"
                device.class = "sound"
                alsa.class = "generic"
                alsa.subclass = "generic-mix"
                alsa.name = "Loopback PCM"
                alsa.id = "Loopback PCM"
                alsa.subdevice = "0"
                alsa.subdevice_name = "subdevice #0"
                alsa.device = "0"
```

3. Enter *pactl list sources*.

Figure 10 pactl list sources command



```
root@edison:~# pactl list sources | grep bluez_source
        Name: bluez_source.98_0D_2E_C8_BD_2C
root@edison:~# 
```

Note: You can use *pactl* to set the default device to use, mute a device, or load a specific module.

PulseAudio server starts with a set of default configurations about audio stream capabilities and the modules to be loaded at start. Audio stream capabilities like sampling rates, resample methods, are defined in the file */etc/pulse/daemon.conf* and the modules loaded are defined in */etc/pulse/system.pa*.

PulseAudio comes with a set of modules with their own roles. Not all modules available in the PulseAudio build are embedded in the default Intel® Edison image, and the ones present in the image are not necessarily loaded by default when PulseAudio starts. The list of modules ready to be used with the default Intel® Edison image can be retrieved with the following command:

```
pulseaudio --dump-modules
```

We can load any of the listed modules with the pactl tool:

```
pactl load-module loopback-module source=xxx sink=xxx
```

To embed any module from the PulseAudio library and add it to your custom image, look at this sample file:

```
${ROOT_EDISON}/device-software/meta-edison-distro/recipes-
multimedia/pulseaudio/pulseaudio_5.0.bbappend
```

This file embeds the A2DP Bluetooth* module.

§

4 Producing Audio with an Intel® Edison Device

This section explains various ways to get sound with an Intel® Edison board using loopback methods, which are interactive and allow an Intel® Edison board to behave as a connected device.

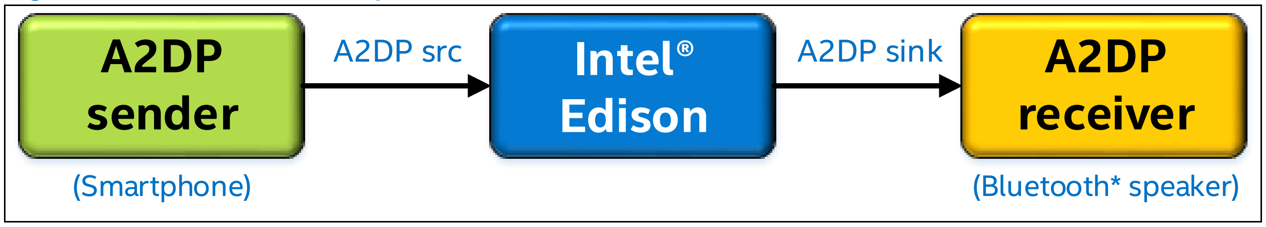
There are three ways to play audio through an Intel® Edison device:

- Loopback A2DP stream to an A2DP remote device.
- Loopback audio from and over a USB headset.
- Loopback A2DP stream over a USB sound device.

4.1 Loopback audio from A2DP source to an A2DP Bluetooth headset

Figure 11 shows what this process looks like graphically.

Figure 10 A2DP audio loopback

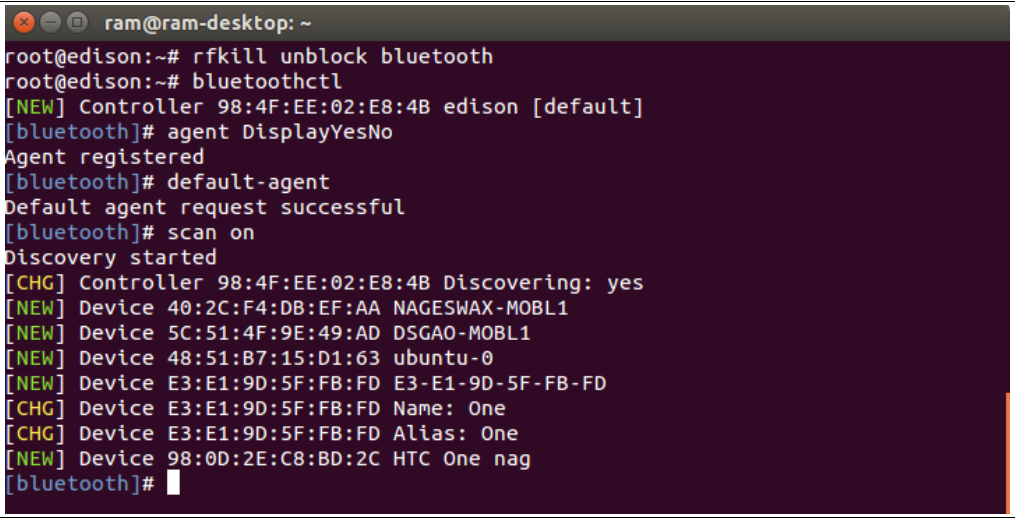


Note: By default, an Intel® Edison device's Bluetooth* name is **BlueZ 5.24**. This is the name that may appear on your player devices.

Step-by-step guide

1. Start the *bluetoothctl* controller (Figure 11).

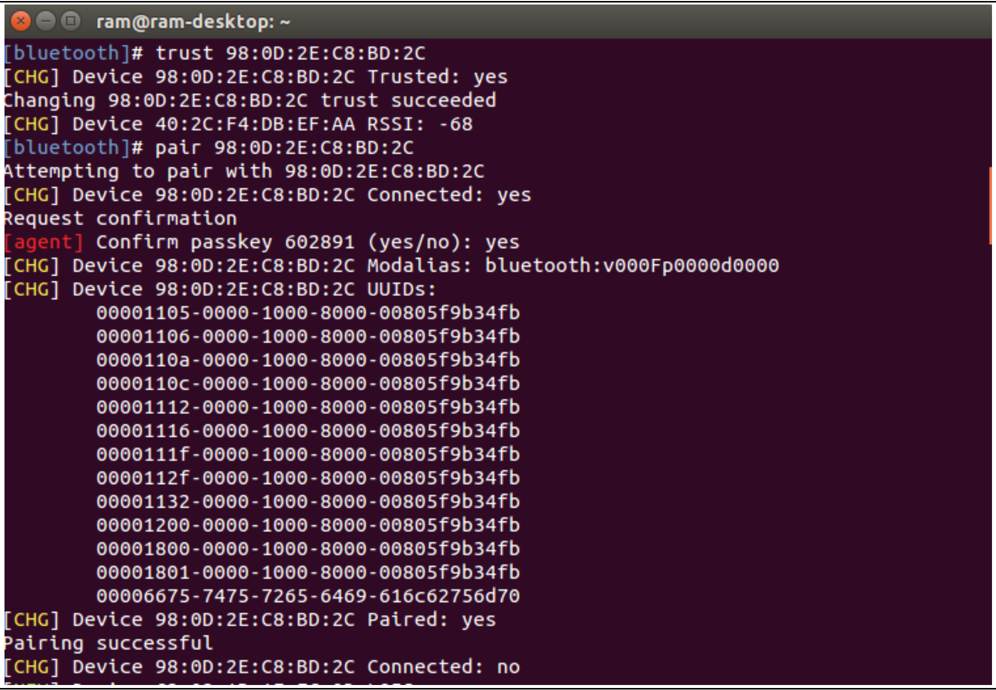
Figure 11 Enabling Bluetooth* with the unblock command



```
root@edison:~# rfkill unblock bluetooth
root@edison:~# bluetoothctl
[NEW] Controller 98:4F:EE:02:E8:4B edison [default]
[bluetooth]# agent DisplayYesNo
Agent registered
[bluetooth]# default-agent
Default agent request successful
[bluetooth]# scan on
Discovery started
[CHG] Controller 98:4F:EE:02:E8:4B Discovering: yes
[NEW] Device 40:2C:F4:DB:EF:AA NAGESWAX-MOBL1
[NEW] Device 5C:51:4F:9E:49:AD DSGAO-MOBL1
[NEW] Device 48:51:B7:15:D1:63 ubuntu-0
[NEW] Device E3:E1:9D:5F:FB:FD E3-E1-9D-5F-FB-FD
[CHG] Device E3:E1:9D:5F:FB:FD Name: One
[CHG] Device E3:E1:9D:5F:FB:FD Alias: One
[NEW] Device 98:0D:2E:C8:BD:2C HTC One nag
[bluetooth]#
```

2. Pair the Intel® Edison device with the audio source device (Android device). In this case, *HTC One nag*.

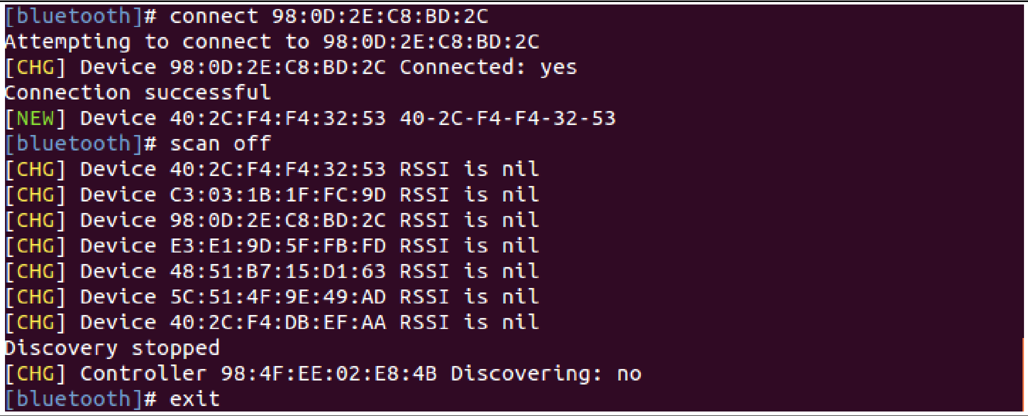
Figure 12 Using the bluetoothctl command to scan and pair with the audio devices as source



```
ram@ram-desktop: ~
[bluetooth]# trust 98:0D:2E:C8:BD:2C
[CHG] Device 98:0D:2E:C8:BD:2C Trusted: yes
Changing 98:0D:2E:C8:BD:2C trust succeeded
[CHG] Device 40:2C:F4:DB:EF:AA RSSI: -68
[bluetooth]# pair 98:0D:2E:C8:BD:2C
Attempting to pair with 98:0D:2E:C8:BD:2C
[CHG] Device 98:0D:2E:C8:BD:2C Connected: yes
Request confirmation
[agent] Confirm passkey 602891 (yes/no): yes
[CHG] Device 98:0D:2E:C8:BD:2C Modalias: bluetooth:v000Fp0000d0000
[CHG] Device 98:0D:2E:C8:BD:2C UUIDs:
        00001105-0000-1000-8000-00805f9b34fb
        00001106-0000-1000-8000-00805f9b34fb
        0000110a-0000-1000-8000-00805f9b34fb
        0000110c-0000-1000-8000-00805f9b34fb
        00001112-0000-1000-8000-00805f9b34fb
        00001116-0000-1000-8000-00805f9b34fb
        0000111f-0000-1000-8000-00805f9b34fb
        0000112f-0000-1000-8000-00805f9b34fb
        00001132-0000-1000-8000-00805f9b34fb
        00001200-0000-1000-8000-00805f9b34fb
        00001800-0000-1000-8000-00805f9b34fb
        00001801-0000-1000-8000-00805f9b34fb
        00006675-7475-7265-6469-616c62756d70
[CHG] Device 98:0D:2E:C8:BD:2C Paired: yes
Pairing successful
[CHG] Device 98:0D:2E:C8:BD:2C Connected: no
```

3. Connecting Intel® Edison device with the audio source device (Figure 12).

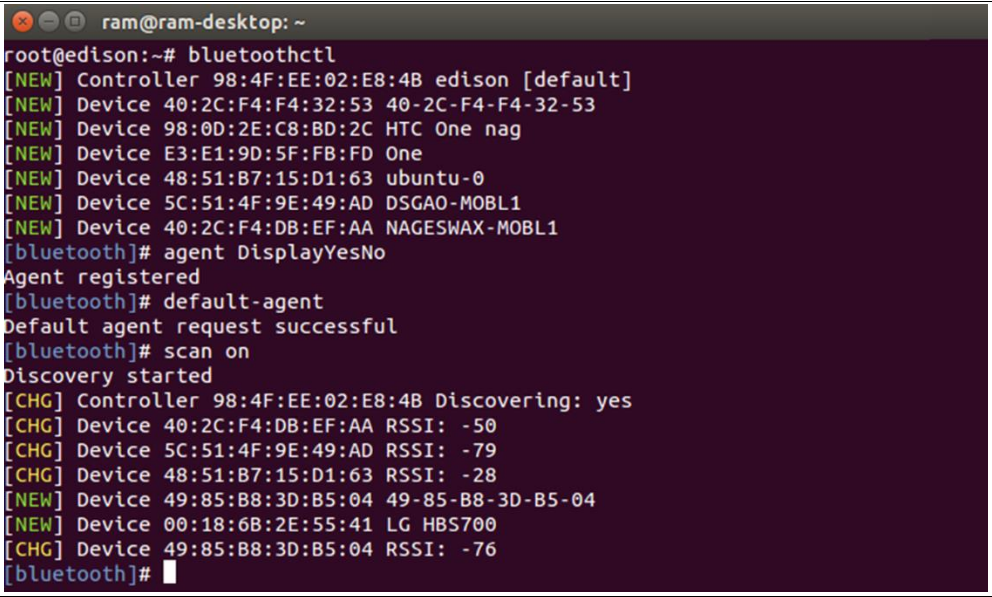
Figure 13 Connect the Bluetooth audio source device



```
[bluetooth]# connect 98:0D:2E:C8:BD:2C
Attempting to connect to 98:0D:2E:C8:BD:2C
[CHG] Device 98:0D:2E:C8:BD:2C Connected: yes
Connection successful
[NEW] Device 40:2C:F4:F4:32:53 40-2C-F4-F4-32-53
[bluetooth]# scan off
[CHG] Device 40:2C:F4:F4:32:53 RSSI is nil
[CHG] Device C3:03:1B:1F:FC:9D RSSI is nil
[CHG] Device 98:0D:2E:C8:BD:2C RSSI is nil
[CHG] Device E3:E1:9D:5F:FB:FD RSSI is nil
[CHG] Device 48:51:B7:15:D1:63 RSSI is nil
[CHG] Device 5C:51:4F:9E:49:AD RSSI is nil
[CHG] Device 40:2C:F4:DB:EF:AA RSSI is nil
Discovery stopped
[CHG] Controller 98:4F:EE:02:E8:4B Discovering: no
[bluetooth]# exit
```

4. Scan the A2DP receiver (sink) device (the Bluetooth* headset) (Figure 14).

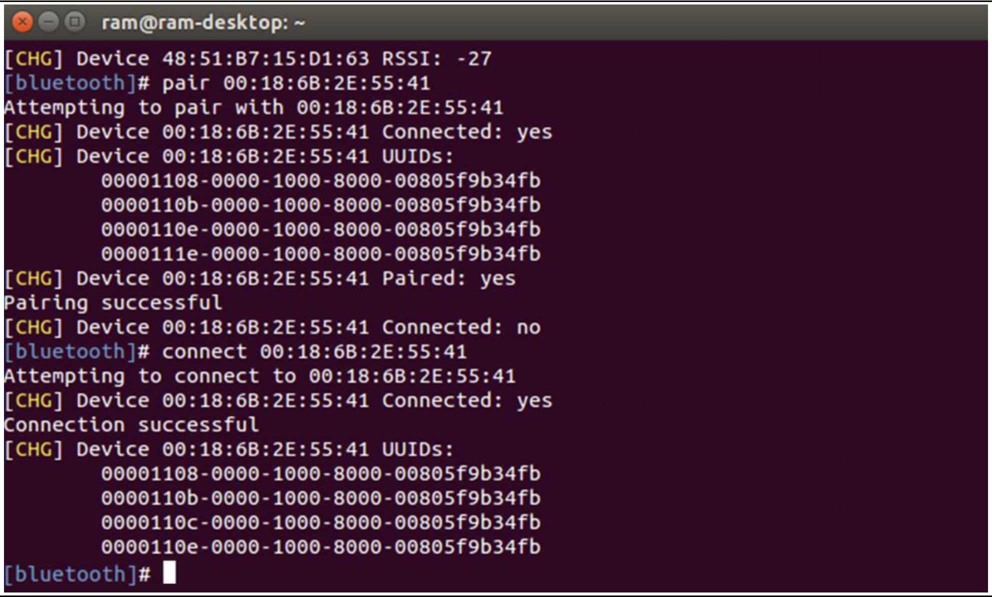
Figure 14 Using the bluetoothctl command to scan for audio receive device (sink)



```
root@edison:~# bluetoothctl
[NEW] Controller 98:4F:EE:02:E8:4B edison [default]
[NEW] Device 40:2C:F4:F4:32:53 40-2C-F4-F4-32-53
[NEW] Device 98:0D:2E:C8:BD:2C HTC One nag
[NEW] Device E3:E1:9D:5F:FB:FD One
[NEW] Device 48:51:B7:15:D1:63 ubuntu-0
[NEW] Device 5C:51:4F:9E:49:AD DSGAO-MOBL1
[NEW] Device 40:2C:F4:DB:EF:AA NAGESWAX-MOBL1
[bluetooth]# agent DisplayYesNo
Agent registered
[bluetooth]# default-agent
Default agent request successful
[bluetooth]# scan on
Discovery started
[CHG] Controller 98:4F:EE:02:E8:4B Discovering: yes
[CHG] Device 40:2C:F4:DB:EF:AA RSSI: -50
[CHG] Device 5C:51:4F:9E:49:AD RSSI: -79
[CHG] Device 48:51:B7:15:D1:63 RSSI: -28
[NEW] Device 49:85:B8:3D:B5:04 49-85-B8-3D-B5-04
[NEW] Device 00:18:6B:2E:55:41 LG HBS700
[CHG] Device 49:85:B8:3D:B5:04 RSSI: -76
[bluetooth]#
```

5. Pair and connect the A2DP receiving device (Figure 15).

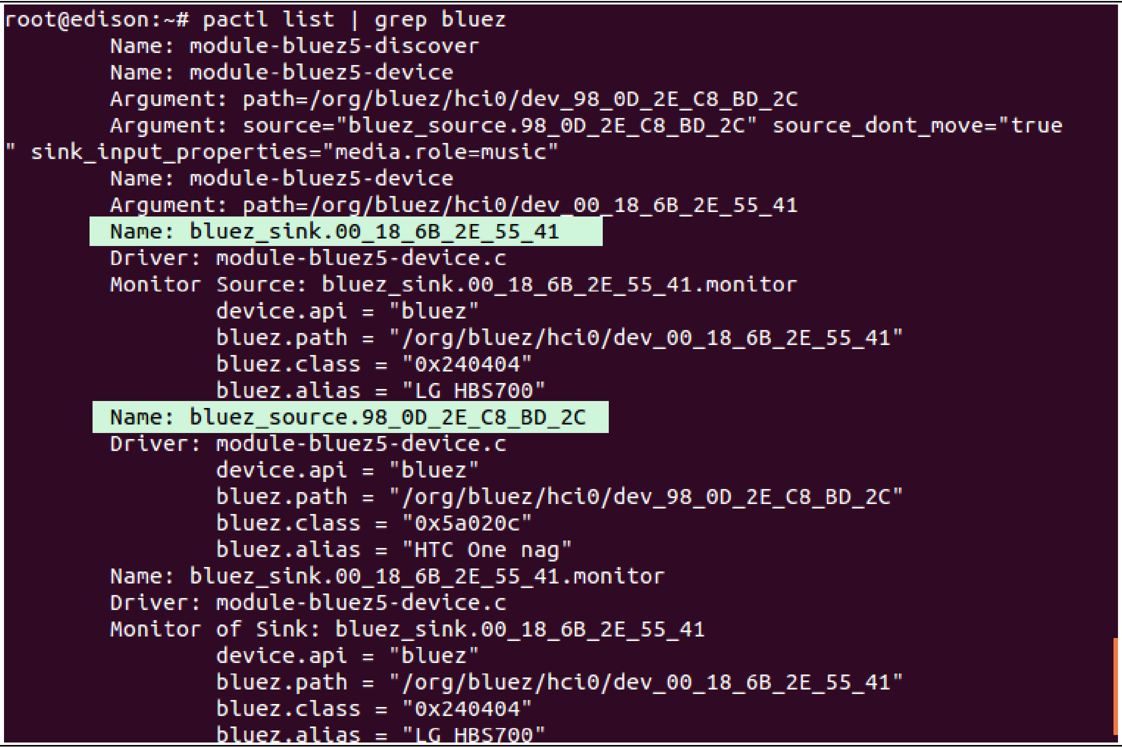
Figure 15 Pair and connect the audio sink device with the Intel® Edison device



```
[CHG] Device 48:51:B7:15:D1:63 RSSI: -27
[bluetooth]# pair 00:18:6B:2E:55:41
Attempting to pair with 00:18:6B:2E:55:41
[CHG] Device 00:18:6B:2E:55:41 Connected: yes
[CHG] Device 00:18:6B:2E:55:41 UUIDs:
        00001108-0000-1000-8000-00805f9b34fb
        0000110b-0000-1000-8000-00805f9b34fb
        0000110e-0000-1000-8000-00805f9b34fb
        0000111e-0000-1000-8000-00805f9b34fb
[CHG] Device 00:18:6B:2E:55:41 Paired: yes
Pairing successful
[CHG] Device 00:18:6B:2E:55:41 Connected: no
[bluetooth]# connect 00:18:6B:2E:55:41
Attempting to connect to 00:18:6B:2E:55:41
[CHG] Device 00:18:6B:2E:55:41 Connected: yes
Connection successful
[CHG] Device 00:18:6B:2E:55:41 UUIDs:
        00001108-0000-1000-8000-00805f9b34fb
        0000110b-0000-1000-8000-00805f9b34fb
        0000110c-0000-1000-8000-00805f9b34fb
        0000110e-0000-1000-8000-00805f9b34fb
[bluetooth]#
```

6. Check if both your A2DP devices are recognized in PulseAudio, then get the source and sink name of both your devices (Figure 16).

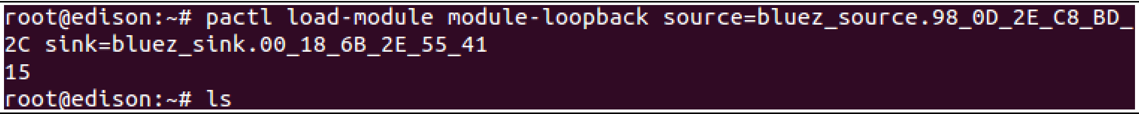
Figure 16 Using the pactl list command to display the BlueZ (source and sink Bluetooth*) devices



```
root@edison:~# pactl list | grep bluez
        Name: module-bluez5-discover
        Name: module-bluez5-device
        Argument: path=/org/bluez/hci0/dev_98_0D_2E_C8_BD_2C
        Argument: source="bluez_source.98_0D_2E_C8_BD_2C" source_dont_move="true
" sink_input_properties="media.role=music"
        Name: module-bluez5-device
        Argument: path=/org/bluez/hci0/dev_00_18_6B_2E_55_41
        Name: bluez_sink.00_18_6B_2E_55_41
        Driver: module-bluez5-device.c
        Monitor Source: bluez_sink.00_18_6B_2E_55_41.monitor
                device.api = "bluez"
                bluez.path = "/org/bluez/hci0/dev_00_18_6B_2E_55_41"
                bluez.class = "0x240404"
                bluez.alias = "LG HBS700"
        Name: bluez_source.98_0D_2E_C8_BD_2C
        Driver: module-bluez5-device.c
                device.api = "bluez"
                bluez.path = "/org/bluez/hci0/dev_98_0D_2E_C8_BD_2C"
                bluez.class = "0x5a020c"
                bluez.alias = "HTC One nag"
        Name: bluez_sink.00_18_6B_2E_55_41.monitor
        Driver: module-bluez5-device.c
        Monitor of Sink: bluez_sink.00_18_6B_2E_55_41
                device.api = "bluez"
                bluez.path = "/org/bluez/hci0/dev_00_18_6B_2E_55_41"
                bluez.class = "0x240404"
                bluez.alias = "LG HBS700"
```

7. Set command to loopback A2DP source device and A2DP sink device (Bluetooth* headset).

Figure 17 Load loopback module with source and sink device as parameters



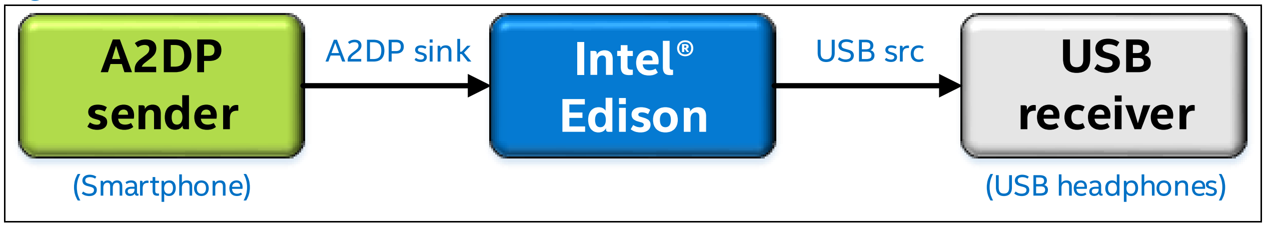
```
root@edison:~# pactl load-module module-loopback source=bluez_source.98_0D_2E_C8_BD_
2C sink=bluez_sink.00_18_6B_2E_55_41
15
root@edison:~# ls
```

8. Play the audio on the source device (Android device). You should be able to hear the audio in the Bluetooth* headset.

4.2 Loopback audio from A2DP smartphone over a USB headset

Figure 18 shows what this process looks like graphically.

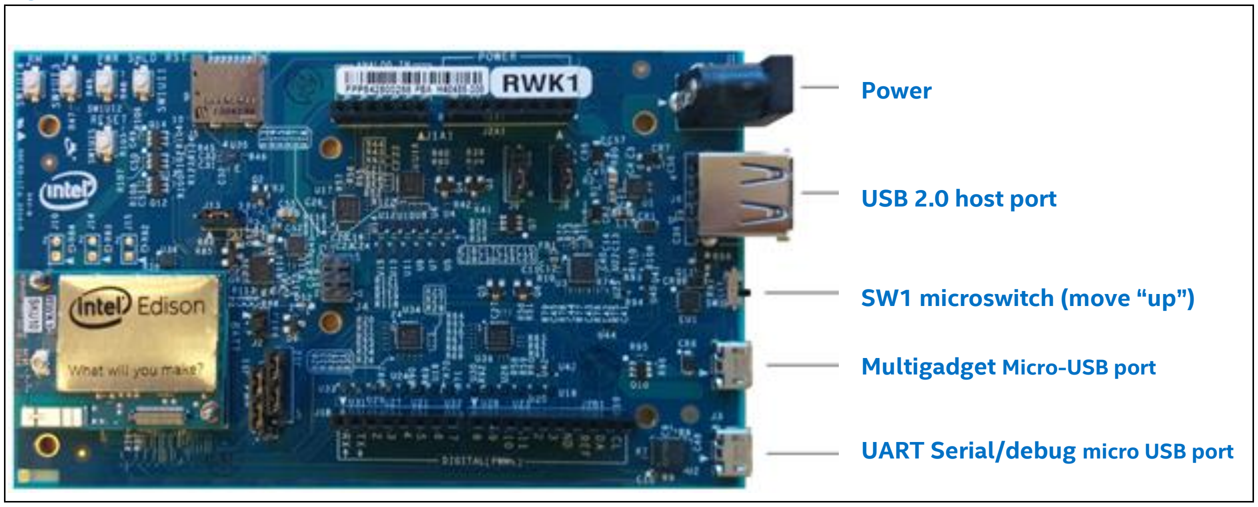
Figure 18 Bluetooth* A2DP audio source to USB sink device



Step-by-step guide

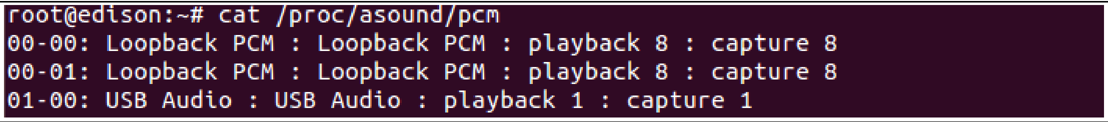
1. Activate USB host mode by moving SW1 up toward the power socket. (You need an external power source in this mode.)

Figure 19 SW1 on an Intel® Edison board on an Arduino* expansion board


Power
USB 2.0 host port
SW1 microswitch (move "up")
Multigadget Micro-USB port
UART Serial/debug micro USB port

2. Plug in your USB headset.
3. Verify that the device is correctly mounted (Figure 20).

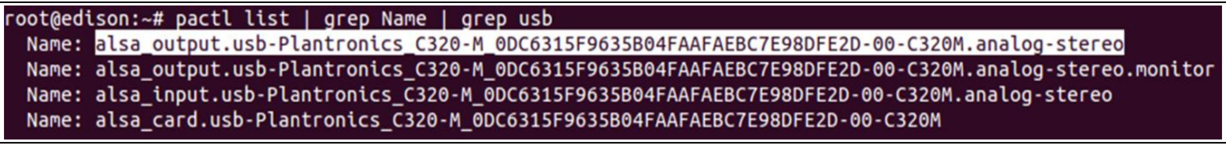
Figure 20 Using the cat command to check whether USB headset is detected



```
root@edison:~# cat /proc/asound/pcm
00-00: Loopback PCM : Loopback PCM : playback 8 : capture 8
00-01: Loopback PCM : Loopback PCM : playback 8 : capture 8
01-00: USB Audio : USB Audio : playback 1 : capture 1
```

4. Since a USB headset is plugged in, you can see the number of playback and capture streams handled by this device.

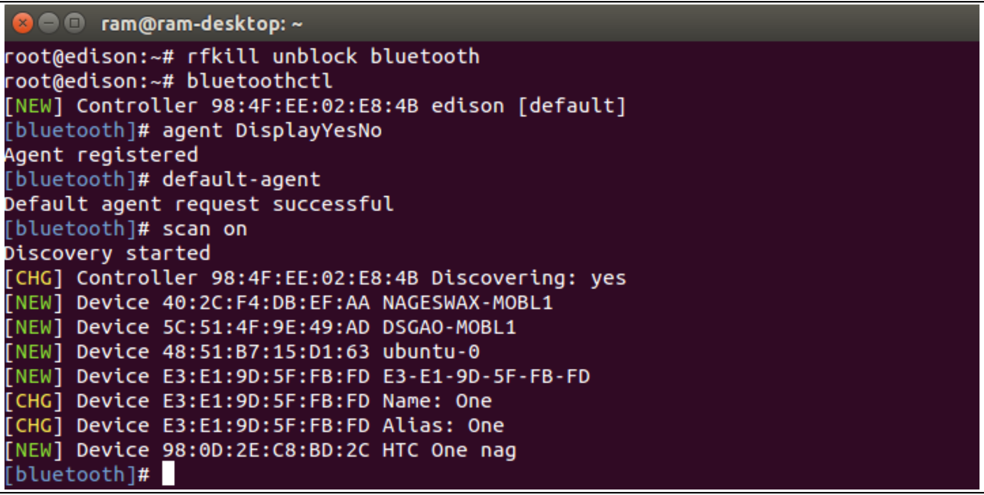
Figure 21 Using the pactl command to get the USB source and sink device names



```
root@edison:~# pactl list | grep Name | grep usb
  Name: alsa_output.usb-Plantronics_C320-M_0DC6315F9635B04FAAFAEBC7E98DFE2D-00-C320M.analog-stereo
  Name: alsa_output.usb-Plantronics_C320-M_0DC6315F9635B04FAAFAEBC7E98DFE2D-00-C320M.analog-stereo.monitor
  Name: alsa_input.usb-Plantronics_C320-M_0DC6315F9635B04FAAFAEBC7E98DFE2D-00-C320M.analog-stereo
  Name: alsa_card.usb-Plantronics_C320-M_0DC6315F9635B04FAAFAEBC7E98DFE2D-00-C320M
```

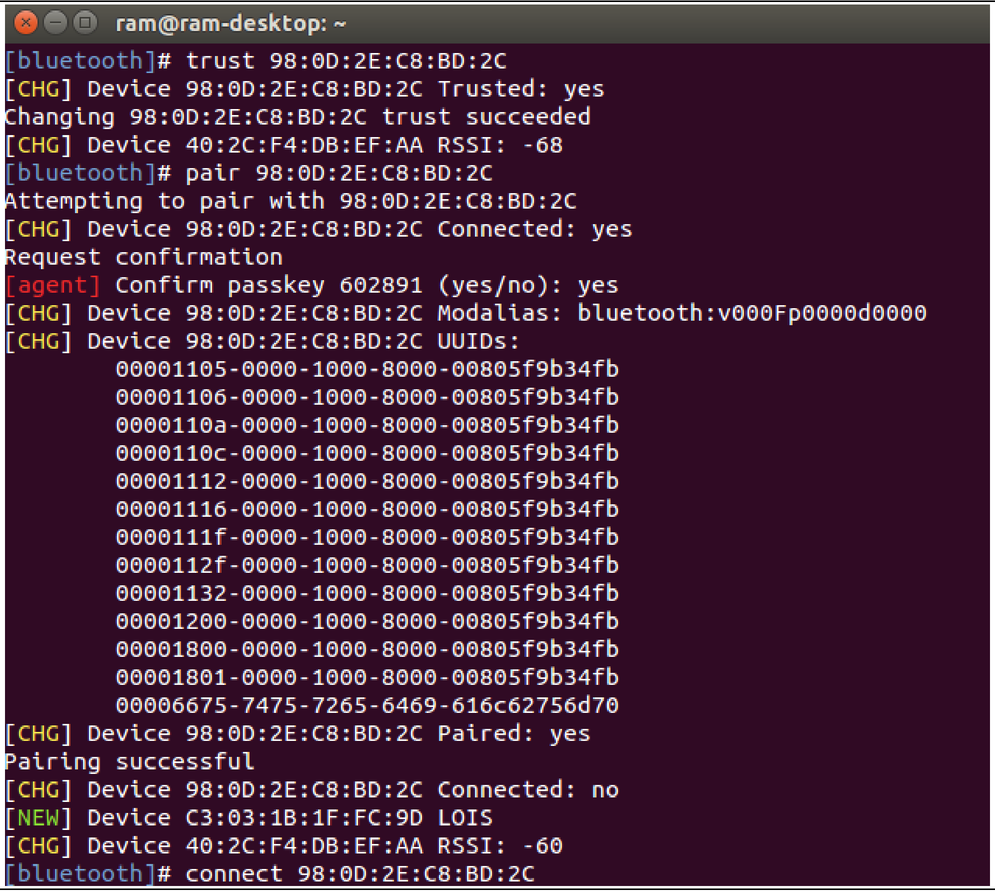
5. Connect your A2DP player device (Android smartphone) and check if it is listed in PulseAudio as a source device (Figure 22).

Figure 22 Using the bluetoothctl scan command



```
ram@ram-desktop: ~
root@edison:~# rfkill unblock bluetooth
root@edison:~# bluetoothctl
[NEW] Controller 98:4F:EE:02:E8:4B edison [default]
[bluetooth]# agent DisplayYesNo
Agent registered
[bluetooth]# default-agent
Default agent request successful
[bluetooth]# scan on
Discovery started
[CHG] Controller 98:4F:EE:02:E8:4B Discovering: yes
[NEW] Device 40:2C:F4:DB:EF:AA NAGESWAX-MOBL1
[NEW] Device 5C:51:4F:9E:49:AD DSGAO-MOBL1
[NEW] Device 48:51:B7:15:D1:63 ubuntu-0
[NEW] Device E3:E1:9D:5F:FB:FD E3-E1-9D-5F-FB-FD
[CHG] Device E3:E1:9D:5F:FB:FD Name: One
[CHG] Device E3:E1:9D:5F:FB:FD Alias: One
[NEW] Device 98:0D:2E:C8:BD:2C HTC One nag
[bluetooth]#
```

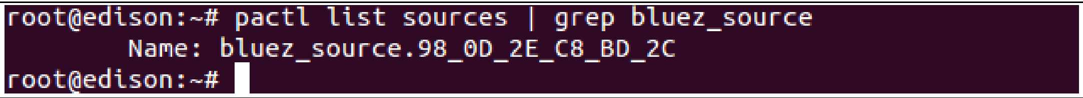
Figure 23 Audio source device pairing and connecting



```
ram@ram-desktop: ~
[bluetooth]# trust 98:0D:2E:C8:BD:2C
[CHG] Device 98:0D:2E:C8:BD:2C Trusted: yes
Changing 98:0D:2E:C8:BD:2C trust succeeded
[CHG] Device 40:2C:F4:DB:EF:AA RSSI: -68
[bluetooth]# pair 98:0D:2E:C8:BD:2C
Attempting to pair with 98:0D:2E:C8:BD:2C
[CHG] Device 98:0D:2E:C8:BD:2C Connected: yes
Request confirmation
[agent] Confirm passkey 602891 (yes/no): yes
[CHG] Device 98:0D:2E:C8:BD:2C Modalias: bluetooth:v000Fp0000d0000
[CHG] Device 98:0D:2E:C8:BD:2C UUIDs:
        00001105-0000-1000-8000-00805f9b34fb
        00001106-0000-1000-8000-00805f9b34fb
        0000110a-0000-1000-8000-00805f9b34fb
        0000110c-0000-1000-8000-00805f9b34fb
        00001112-0000-1000-8000-00805f9b34fb
        00001116-0000-1000-8000-00805f9b34fb
        0000111f-0000-1000-8000-00805f9b34fb
        0000112f-0000-1000-8000-00805f9b34fb
        00001132-0000-1000-8000-00805f9b34fb
        00001200-0000-1000-8000-00805f9b34fb
        00001800-0000-1000-8000-00805f9b34fb
        00001801-0000-1000-8000-00805f9b34fb
        00006675-7475-7265-6469-616c62756d70
[CHG] Device 98:0D:2E:C8:BD:2C Paired: yes
Pairing successful
[CHG] Device 98:0D:2E:C8:BD:2C Connected: no
[NEW] Device C3:03:1B:1F:FC:9D LOIS
[CHG] Device 40:2C:F4:DB:EF:AA RSSI: -60
[bluetooth]# connect 98:0D:2E:C8:BD:2C
```

6. Check if your A2DP device is recognized in PulseAudio and get the source name of your device. (It should start with *bluez_source*.)

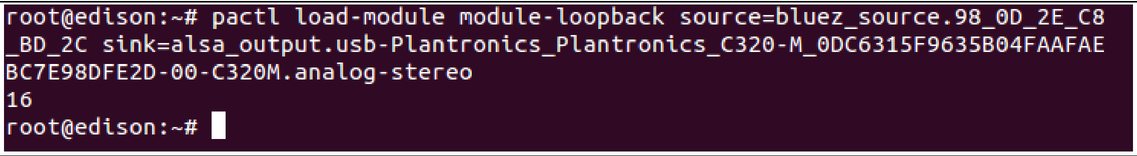
Figure 24 Using the pactl list command to get the audio source



```
root@edison:~# pactl list sources | grep bluez_source
        Name: bluez_source.98_0D_2E_C8_BD_2C
root@edison:~#
```

7. Loopback the A2DP source device (Android device) to the USB playback device so that the audio playing on the Android device is routed to the USB headset connected to Intel® Edison device (Figure 25).

Figure 25 Using the pactl list command to get the audio source



```
root@edison:~# pactl load-module module-loopback source=bluez_source.98_0D_2E_C8
_BD_2C sink=alsa_output.usb-Plantronics_Plantronics_C320-M_0DC6315F9635B04FAAFAE
BC7E98DFE2D-00-C320M.analog-stereo
16
root@edison:~#
```

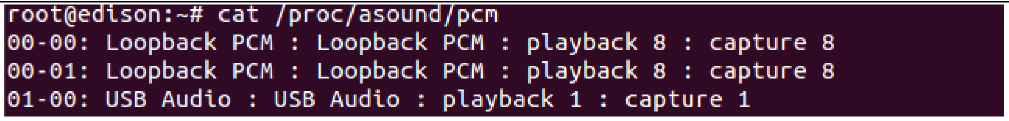
8. Play an audio file on your A2DP audio source device and check for audio on the headset.

4.3 Loopback audio from and over a USB headset

Step-by-step guide

1. Activate the USB host mode by moving the SW1 to the right position. (You need an external power source in this mode.)
2. Plug in your USB headset.
3. Check if the device is correctly mounted (Figure 26).

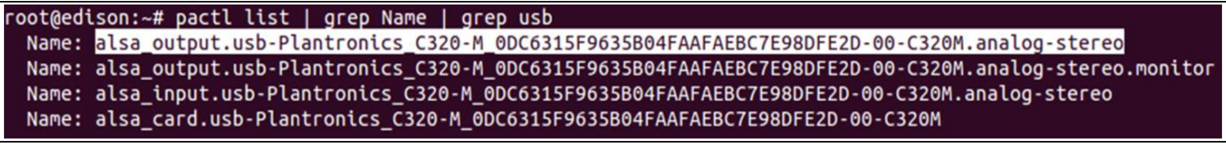
Figure 26 Using the pactl list command to get the audio source



```
root@edison:~# cat /proc/asound/pcm
00-00: Loopback PCM : Loopback PCM : playback 8 : capture 8
00-01: Loopback PCM : Loopback PCM : playback 8 : capture 8
01-00: USB Audio : USB Audio : playback 1 : capture 1
```

4. Since a USB headset is plugged in, you can see the number of playback and capture streams handled by this device.

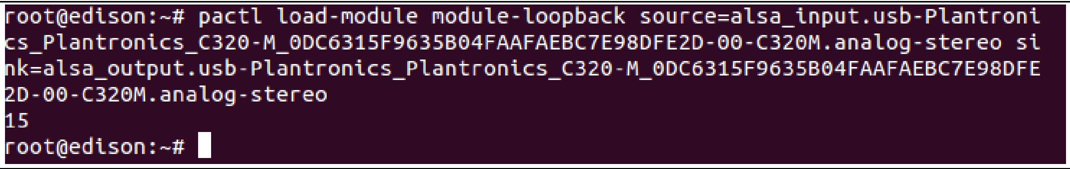
Figure 27 A2DP audio loopback



```
root@edison:~# pactl list | grep Name | grep usb
  Name: alsa_output.usb-Plantronics_C320-M_0DC6315F9635B04FAAFAEBC7E98DFE2D-00-C320M.analog-stereo
  Name: alsa_output.usb-Plantronics_C320-M_0DC6315F9635B04FAAFAEBC7E98DFE2D-00-C320M.analog-stereo.monitor
  Name: alsa_input.usb-Plantronics_C320-M_0DC6315F9635B04FAAFAEBC7E98DFE2D-00-C320M.analog-stereo
  Name: alsa_card.usb-Plantronics_C320-M_0DC6315F9635B04FAAFAEBC7E98DFE2D-00-C320M
```

5. Loopback the USB source device (USB Mic) to the USB playback device so that the audio received from the MIC is routed to the USB headset connected to Intel® Edison device (Figure 28).

Figure 28 A2DP audio loopback



```
root@edison:~# pactl load-module module-loopback source=alsa_input.usb-Plantroni
cs_Plantronics_C320-M_0DC6315F9635B04FAAFAEBC7E98DFE2D-00-C320M.analog-stereo si
nk=alsa_output.usb-Plantronics_Plantronics_C320-M_0DC6315F9635B04FAAFAEBC7E98DFE
2D-00-C320M.analog-stereo
15
root@edison:~#
```

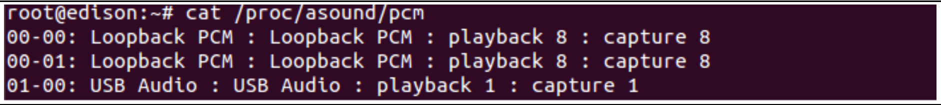
6. The user can play an audio file or speak through the microphone and verify that the sound is audible on the headset. By default, it should be audible as soon as the setup is complete.

4.4 Audio record and play through USB headset

Step-by-step guide

1. Activate the USB host mode by moving the SW1 to right position. (You need an external power source in this mode.)
2. Plug your USB headset.
3. Check if the device is correctly mounted (Figure 29).

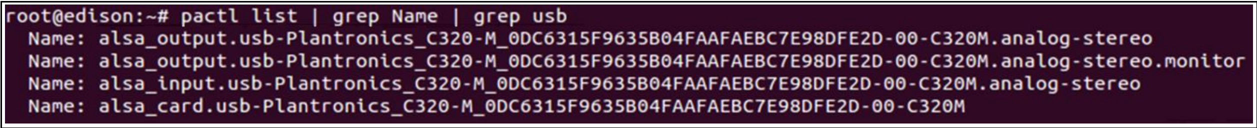
Figure 29 Using the pactl list command to get the audio source



```
root@edison:~# cat /proc/asound/pcm
00-00: Loopback PCM : Loopback PCM : playback 8 : capture 8
00-01: Loopback PCM : Loopback PCM : playback 8 : capture 8
01-00: USB Audio : USB Audio : playback 1 : capture 1
```

4. Since a USB headset is plugged in, you can see the number of playback and capture streams handled by this device.

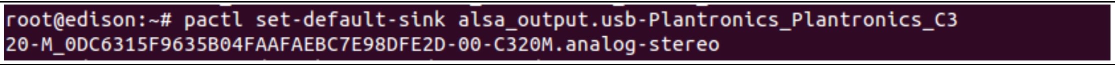
Figure 30 A2DP audio lists



```
root@edison:~# pactl list | grep Name | grep usb
  Name: alsa_output.usb-Plantronics_C320-M_0DC6315F9635B04FAAFAEBC7E98DFE2D-00-C320M.analog-stereo
  Name: alsa_output.usb-Plantronics_C320-M_0DC6315F9635B04FAAFAEBC7E98DFE2D-00-C320M.analog-stereo.monitor
  Name: alsa_input.usb-Plantronics_C320-M_0DC6315F9635B04FAAFAEBC7E98DFE2D-00-C320M.analog-stereo
  Name: alsa_card.usb-Plantronics_C320-M_0DC6315F9635B04FAAFAEBC7E98DFE2D-00-C320M
```

5. Set the default output sink device as USB headset (Figure 31).

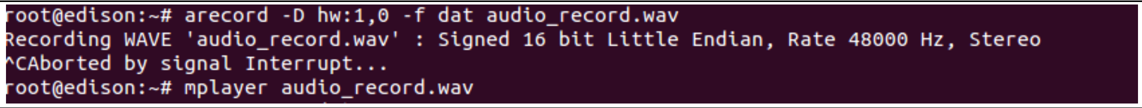
Figure 31 Set default audio output device



```
root@edison:~# pactl set-default-sink alsa_output.usb-Plantronics_Plantronics_C3
20-M_0DC6315F9635B04FAAFAEBC7E98DFE2D-00-C320M.analog-stereo
```

6. The device is ready for recording and playing of audio. Run the *arecord* command to record audio from the MIC and store it into local file on the Intel® Edison device (Figure 32).

Figure 32 Using the arecord command to record audio

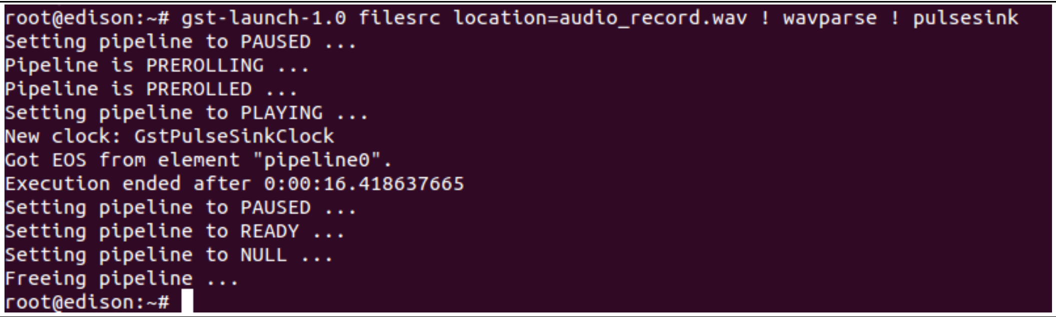


```
root@edison:~# arecord -D hw:1,0 -f dat audio_record.wav
Recording WAVE 'audio_record.wav' : Signed 16 bit Little Endian, Rate 48000 Hz, Stereo
^CAborted by signal Interrupt...
root@edison:~# mplayer audio_record.wav
```

Note: Check the card ID and device ID of the USB headset MIC by using `arecord -l` command and change the card ID and device ID number accordingly in the above command.

7. After recording for some time, press Ctrl+C and run the GStreamer command to play the *audio_record.wav* audio file (Figure 33).

Figure 33 GStreamer to play the recorded audio



```
root@edison:~# gst-launch-1.0 filesrc location=audio_record.wav ! wavparse ! pulsesink
Setting pipeline to PAUSED ...
Pipeline is PREROLLING ...
Pipeline is PREROLLED ...
Setting pipeline to PLAYING ...
New clock: GstPulseSinkClock
Got EOS from element "pipeline0".
Execution ended after 0:00:16.418637665
Setting pipeline to PAUSED ...
Setting pipeline to READY ...
Setting pipeline to NULL ...
Freeing pipeline ...
root@edison:~#
```

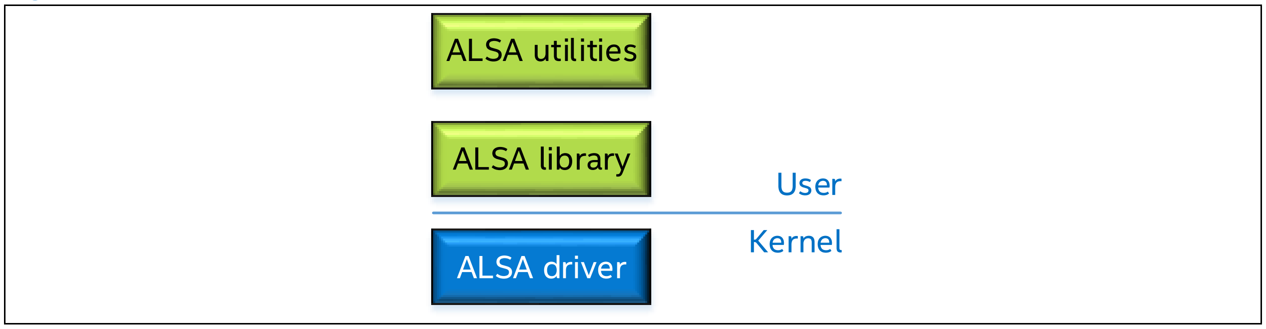
8. You should be able to play the recorded audio from the USB headset.

§

5 Advanced Linux Sound Architecture (ALSA)

ALSA is a framework and integral part of the Linux kernel that provides APIs for sound card device drivers. Some of the goals of the ALSA project at its inception were automatic configuration of sound card hardware and graceful handling of multiple sound devices in a system. ALSA is released under the GNU General Public License (GPL) and the GNU Lesser General Public License (LGPL).

Figure 34 ALSA architecture



5.1 ALSA user space

5.1.1 ALSA utilities

The ALSA utilities package contains a set of utility interfaces that control the sound card operations like get information, create a capture, or a playback stream. These are some of the most commonly used utilities:

- **amixer** - Display, get control on Alsa driver controls.
- **arecord** - Create a capture stream from an audio device.
- **aplay** - Play a file and play back stream to an audio device.
- **alsactl** - Store or restore an audio device configuration.

Note: See *http://www.linuxfromscratch.org/blfs/view/6.3/multimedia/alsa-utils.html* for more information. These utilities depend on Alsa-lIb.

5.1.2 ALSA lib

The ALSA lib interface APIs are nothing but the interface to the ALSA drivers. Developers working on audio who want to use ALSA need to use these interfaces to achieve native ALSA support for their applications. See *http://www.alsa-project.org/alsa-doc/alsa-lib* for more information.

5.2 ALSA kernel space

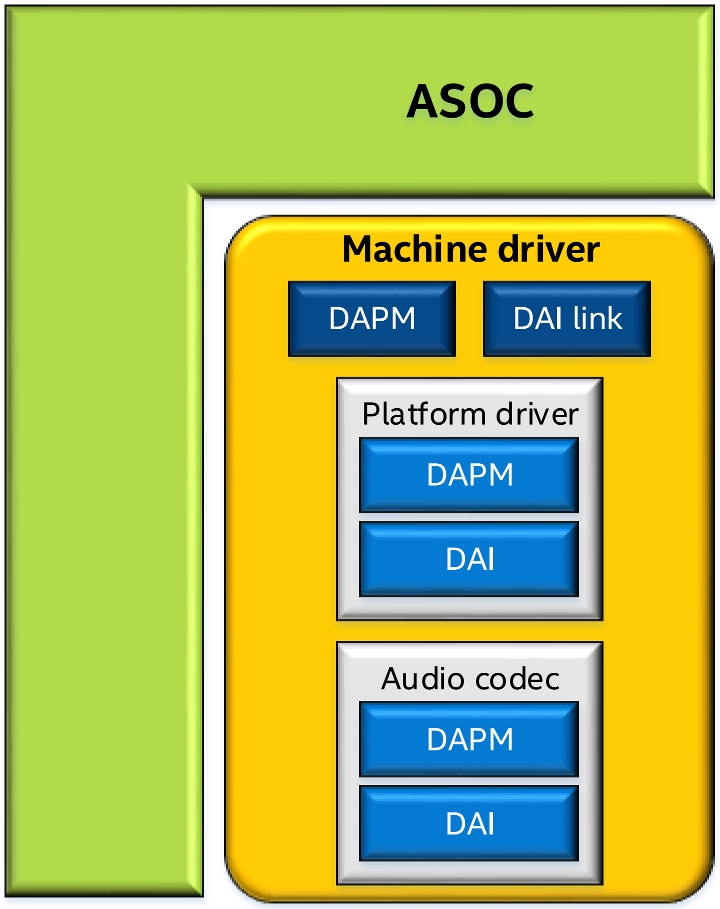
5.2.1 ALSA-driver

The ALSA-driver layer is designed to make it easy to implement and handle audio devices.

As the Intel® Edison board is based on an SoC, we will focus on the ASOC (ALSA system-on-a-chip) implementation, which was designed to provide better ALSA support for embedded system-on-chip processors. ASOC basically splits an embedded audio system into multiple reusable component drivers.

- **The audio codec** - Contains all audio controls (like mixers, gain, etc.), audio interface capabilities (support/configuration of I2S), I/O, and power management definition (DAPM). Must be platform independent.
- **The platform driver** - Support of the DMA engine, **D**igital **A**udio **I**nterface drivers (I2S/PCM/AC97), and if necessary, audio DSP controls + **DAPM** support.
- **The machine driver** - This is the glue between the two previous parts and the management of external events (like an event when a headset is plugged in).

Figure 35 ALSA driver in ASOC



See *https://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git/tree/Documentation/sound/alsa/soc/overview.txt* for more information.

§

6 Audio DSP

The Intel® Atom™ processor on the Intel® Edison board has an embedded audio DSP. (See section 2.1 for in-depth hardware specification details.) The SSP2 port allows platform makers to plug in an audio codec to add sound enhancement to the Intel® Edison board. The four wires can be easily accessed with the breakout board on **J19 (pin 11/10) and J20 (pin 11/10)** connectors. Refer to the *Intel® Edison Breakout Board Hardware Guide* (331190) for details (*http://www.intel.com/support/edison/sb/CS-035252.htm)*. The SSP1 port is used by the Bluetooth* chip. It can be used for the HSP and HFP profiles in PCM mode (vs. HCI mode, where the audio is sent through the UART).

This section is mainly intended for kernel developers.

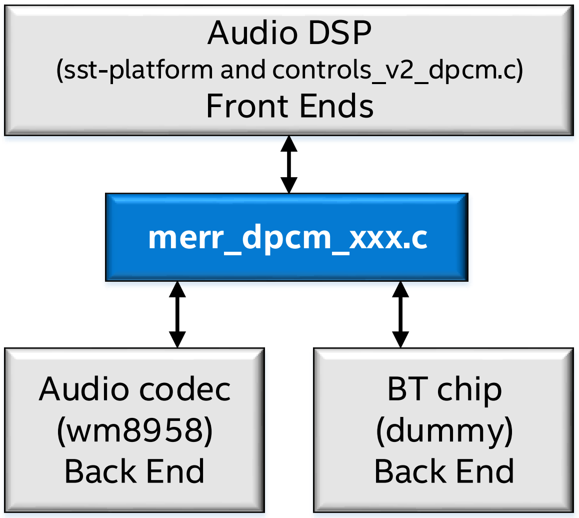
You can view the driver reference code at in the Intel® Edison kernel source tree here:

```
${ROOT_EDISON}/linux/sound/soc/intel
```

The Intel driver supports dynamic PCM (DPCM). It allows you to switch during runtime the DAI back end (like audio codec, Bluetooth* chip) connected to a DAI front end (media port, voip port of audio DSP).

See *https://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git/tree/Documentation/sound/alsa/soc/DPCM.txt* for more information.

Figure 36 Flow between ASOC components



6.1 Implementing a dummy codec with the audio DSP

The audio DSP exposes ALSA control to route, configure audio stream from/to the available **F**ront **E**nd (**FE**) to the connected **B**ack **E**nd (**BE**) and the audio DSP, as seen in section 2.1, is **master** of the SSP ports. So even if there is no audio codec, we can drive the SSP2 port and check the signals with an oscilloscope.

6.1.1 Create a machine driver with a dummy codec

1. Look at the dummy implementation of the driver at the following location:

```
${ROOT_EDISON}/linux/sound/soc/intel/board/merr_dpcm_dummy.c
```

There are two significant parts to this file:

```
struct snd_soc_dai_link merr_msic_dailink[] = {
 [MERR_DPCM_AUDIO] = {
 .name = "Media Audio Port",
.stream_name = "Edison Audio",
.cpu_dai_name = "Headset-cpu-dai",
....
          /* back ends */
{
 .name = "SSP2-Codec",
 .be_id = 1,
 .cpu_dai_name = "ssp2-codec",
 .platform_name = "sst-platform",
 .no_pcm = 1,
 .codec_dai_name = "snd-soc-dummy-dai",
 .codec_name = "snd-soc-dummy",
...
};
```

This structure defines the **FE (front end)** and **BE (back end)** supported by the dummy DPCM machine driver.

```
static const struct snd_soc_dapm_route map[] = {
 { "Dummy Playback", NULL, "codec_out0" },
 { "Dummy Playback", NULL, "codec_out1" },
 { "codec_in0", NULL, "Dummy Capture" },
 { "codec_in1", NULL, "Dummy Capture" },
};
```

This structure defines the interconnections between the FE (audio DSP) I/O widgets codec_in and codec_out:

```
${root_edison}/linux/sound/soc/intel/platform-libs/controls_v2_dpcm.c
```

...and the BE I/O (dummy codec) widgets linked for dummy playback and dummy capture:

```
${root_edison}/linux/sound/soc/soc-utils.c
```

2. To register the new machine driver, refer to :

```
${root_edison}/arch/x86/platform/intel-
mid/device_libs/platform_mrfld_audio.c
void *mrfld_sst_audio_platform_data(void *info)
```

This is a basic implementation of a dummy codec with the audio DSP. It allows user to stream an audio playback to the SSP2 port.

You can find the default audio DSP configuration associated in the *asound.state* file located in the */var/lib/alsa* folder. This configuration has to be set for any codec implementation as it configures the audio DSP path to capture or play back an audio stream through the SSP2 port, and before any open devices.

6.1.2 Testing

By default, the dummy machine driver is selected by the kernel command line with the argument "audio_codec=dummy". Set it to "wm8958" if you have a **wm8958 codec** connected to the Intel® Edison board.

Use GStreamer to play a **Wav file** on the ALSA device and check the signals with an oscilloscope.

§