Zen Hat Audio Guide – Kernel 6.1.x

by Brian Fraser
Last update: March 4, 2025

Guide has been tested on

BeagleY-AI (target):	**Debian 12.x**
PC OS (host):	**Debian 12.x**

This document guides the user through

1. Getting audio output from the Zen Hat
2. Getting audio output from a USB Audio Adapter
3. Playing PCM (wave) files via a C program through `asound`

Table of Contents

Formatting

1. Commands for the host Linux's console are show as:
   ```
   (host)$ echo "Hello PC world!"
   ```
2. Commands for the target (BeagleY-AI) Linux's console are shown as:
   ```
   (byai)$ echo "Hello embedded world!"
   ```
3. Almost all commands are case sensitive.

Revision History

- March 4, 2025: Convert to using BeagleY-AI

1. Zen Hat Audio: Installing Virtual Audio Overlay

For Linux to use the audio hardware on the Zen Hat, we must install a device tree file to tell the kernel how to access the hardware.

Linux must be told what hardware is connected to the CPU. It learns this at boot up using a Device Tree (file is a .DTB for the device tree binary). The boot loader (UBoot) is told what extra hardware configurations to load as device tree overlays.

Do the following just once (per board).

1.1 Install Files

1. Download the `ZenHatOverlayAndTLV320AIC3104Driver.zip` file from the course website and extract it into your NFS folder (`/mnt/remote/`)
2. On the target, copy the drivers for the Zen Hat's audio codec:
   - First check if they are already installed:
     ```
     (byai)$ cd /lib/modules/6.1.83-ti-arm64-r63/kernel/sound/soc/codecs/
     (byai)$ ls snd-soc-tlv320aic3x*
     ```
   - **If it lists two files, SKIP TO STEP 3.**
     If `ls` did not find any files, then:
     ```
     (byai)$ sudo cp /mnt/remote/snd-soc-tlv320aic3x* .
     ```
   - Register the new drivers with Linux so it will load them when needed:
     ```
     (byai)$ cd /lib/modules/6.1.83-ti-arm64-r63
     (byai)$ sudo depmod -a
     ```
2. On the target, copy the device tree overlay for the audio codec:
   ```
   (byai)$ cd /boot/firmware/overlays/
   (byai)$ sudo cp /mnt/remote/k3-am67a-beagley-ai-zenhat-audio.dtbo .
   ```

1.2 Load the Overlay

3. Backup `extlinux.conf`:

   This file is critical to controlling how UBoot starts the system. We will change it to load the audio overlay; however, first take a backup copy so we can recover from any error.

   ```
   (byai)$ cd /boot/firmware/extlinux
   (byai)$ sudo cp extlinux.conf extlinux.conf.before_audio
   ```
   - **WARNING:** Corrupting the **`extlinux.conf`** file may cause the BeagleY-AI to be unbootable. Be careful editing this file! You may be able to fix it by putting your uSD card into a card reader and editing the file on your computer.
4. Edit `extlinux.conf` to load the audio overlay
   - Edit `extlinux.conf`:
     ```
     (byai)$ sudo nano /boot/firmware/extlinux/extlinux.conf
     ```
   - Change the last section to be:

```
label microSD (default)
    kernel /Image
    append console=ttyS2,115200n8 root=/dev/mmcblk1p3 ro rootfstype=ext4 resume=/dev/mmcblk1p2 rootwait net.ifnames=0 quiet
    fdtdir /
    fdt /ti/k3-am67a-beagley-ai.dtb
    #fdtoverlays /overlays/<file>.dtbo
    fdtoverlays /overlays/k3-am67a-beagley-ai-zenhat-audio.dtbo
    #initrd /initrd.img
```

- If loading multiple overlays (SPI, PWM, ...) put them all on one line with the `ftdoverlays`.
- Ensure you **removed the #** on the line to uncomment it.

5. Reboot the target.
```
(byai)$ sudo reboot -f
```

6. Verify it worked:
   - Ensure the audio has loaded (first give it access to the internet):
```
(byai)$ sudo apt install alsa-utils
(byai)$ aplay -l     # lower-case L
**** List of PLAYBACK Hardware Devices ****
card 0: Audio [Zen Hat Audio], device 0: davinci-mcasp.0-tlv320aic3x-hifi
tlv320aic3x-hifi-0 [davinci-mcasp.0-tlv320aic3x-hifi tlv320aic3x-hifi-0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
card 1: HDMI [it66122 HDMI], device 0: davinci-mcasp.0-i2s-hifi i2s-hifi-0
[davinci-mcasp.0-i2s-hifi i2s-hifi-0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```
   - Ensure the I²C for bus 1 shows that address 0x18 is controlled by a driver (shows “UU”):
```
(byai)$ i2cdetect -y -r 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                         -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- UU 19 -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- 48 -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

7. Troubleshooting

- The following script may be very useful for finding out the state of the system:

```
#! /bin/bash
# Script to show debug information about audio overlay

echo "==> i2cdetect -y -r 1"
i2cdetect -y -r 1

echo "==> ls /proc/device-tree/chosen/overlays/"
ls /proc/device-tree/chosen/overlays/

echo "==> dmesg | grep -iE 'deferred|audio|snd|sound|tlv|alsa|mcasp|-->'"
dmesg | grep -iE 'deferred|audio|snd|sound|tlv|alsa|mcasp|-->'

echo "==> cat /sys/kernel/debug/devices_deferred"
cat /sys/kernel/debug/devices_deferred

echo "==> lsmod | grep -i tlv"
lsmod | grep -i tlv

echo "==> gpioinfo | grep -i GPIO26"
gpioinfo | grep -i GPIO26

echo "==> aplay -l"
aplay -l
```

- If the command:
  `(byai)$ ls /proc/device-tree/chosen/overlays/`
  does not list `k3-am67asbeaglemod-audio`, then there is likely something incorrect with `extlinux.conf` file, or the `.dtbo` file is not in the correct location.
- If the command:
  `(byai)$ cat /sys/kernel/debug/devices_deferred`
  shows anything about the audio driver, it likely means that you either did not copy the `.ko.xz` driver files into the correct location, or did not correctly run `depmod`.
- General debugging of device tree issues:
  - Use the serial port (via an external debug probe) to view the messages that UBoot prints out during startup. If your kernel is booting, then also have a good look for messages in `dmesg` as it may tell you what goes wrong after UBoot configures the system.
  - If Robert C. Nelson wrote it, it's probably correct.

2. (Optional) Setting up a USB Audio Adapter

If you are using a USB audio adapter (such as an ENVEL USB to 3.5mm Audio Adapter), the following steps will configure the board to be able to play audio. Note that not all USB audio adapters will work: I purchased four from Amazon and only 2 functioned with the others preventing the board from turning on, booting correctly, or loading a driver.

1. On the target, install the sound tools and libraries (`aplay`, `alsamixer`, `lsusb`)
   ```
   (byai)$ sudo apt install usbutils alsa-utils i2c-tools
   ```
2. Plug in your USB Audio Adapter to your target. Ensure it is detected:
   ```
   (byai)$  lsusb
   ```
   Possible output:
   ```
   Bus 001 Device 003: ID 1b3f:2008 Generalplus Technology Inc. USB Audio Device
   Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
   ```
   - If needed, run `dmesg` to find out information about how the Linux kernel tried to load drivers, etc.
   - This guide tested with:
     "Generalplus Technology Inc. USB Audio Device" from Envel
     "C-Media Electronics, Inc. Audio Adapter (Unitek Y-247A)" from Cable Creation
3. Set the default sound card for ALSA on the target:
   - View sound cards available:
     ```
     (byai)$ cat /proc/asound/cards
     ```
     Output:
     ```
     1 [Device         ]: USB-Audio - USB Audio Device
             GeneralPlus USB Audio Device at usb-musb-hdrc.1-1, full speed
     ```
   - Set the default, changing the "1" below to the desired device number listed above:
     ```
     (byai)$ sudo nano /etc/asound.conf
     ```

     Set contents to:
     ```
     defaults.pcm.card 1
     defaults.ctl.card 1
     ```

3. Play Audio

1. On the target, install the sound tools and libraries (`aplay`, `alsamixer`, `lsusb`)
   `(byai)$ sudo apt install usbutils alsa-utils i2c-tools`
2. Plug in speakers or headphones into the audio output on the Zen Hat (green 3.5mm socket).
   - **WARNING:** When testing your audio, do not put headphones in your ear. A very loud sound is possible if there are problems, and this could cause an injury to your ear.

     Just drape the head-phones beside your ears so you can hear it, but not be injured if it goes wrong. Once you know the audio levels are fine, then using head-phones normally is fine.
3. Save a WAVE file to your NFS folder on your host (say, `sample.wav`).
4. On the target, play the file with:
   `(byai)$` **`aplay sample.wav`**
5. Change the volume:
   `(byai)$` **`alsamixer`**
   - Use the left/right arrows to select different channels to adjust.
   - Use the up/down arrows to change the volume.
   - Press 'M' to mute or unmute channels.
   - Press ESC to exit (and save changes).
   - Change the volume of wave data playback by changing the `PCM` channel.
6. Troubleshooting:
   - You can list available playback devices to ensure the configuration succeeded:
     ```
     (byai)$ aplay -l
     **** List of PLAYBACK Hardware Devices ****
     card 0: Audio [Zen Hat Audio], device 0: davinci-mcasp.0-tlv320aic3x-hifi
     tlv320aic3x-hifi-0 [davinci-mcasp.0-tlv320aic3x-hifi tlv320aic3x-hifi-0]
       Subdevices: 1/1
       Subdevice #0: subdevice #0
     card 1: HDMI [it66122 HDMI], device 0: davinci-mcasp.0-i2s-hifi i2s-hifi-0
     [davinci-mcasp.0-i2s-hifi i2s-hifi-0]
       Subdevices: 1/1
       Subdevice #0: subdevice #0
     ```
     - You can also list PCM playback devices:
       `(byai)$` **`aplay -L`**
     - You can view same info:
       `(byai)$` **`cat /proc/asound/cards`**
   - You can play back white noise to test the hardware:
     `(byai)$` **`speaker-test`**
   - You can playback a sound on a specific card using a command such as:
     `(byai)$ aplay -D default:CARD=Device ./my-file.wav`
     Where the `CARD` parameter can be found by running `` `aplay -L` ``, look for "`CARD=`"
   - If the output from the software looks like it should be playing audio but no sound is generated:
     - Ensure you have speakers or headphones plugged into the green audio-out jack, and that you have fully pushed in the connector (may hear a small click as it goes in all the way).
     - Ensure you have loaded the audio overlay. Use "aplay -l" to verify that you see the "Zen Hat Audio" and likely the HDMI as well.

- If you get the following error with `alsamixer`, it means the audio overlay is not loaded: `"This sound device does not have any controls"`
- If you get the following error with `aplay` while using a USB audio device, it likely means you have not yet set your default audio device, or your USB Audio Adapter is not connected:

```
ALSA lib confmisc.c:767:(parse_card) cannot find card '0'
ALSA lib conf.c:4745:(_snd_config_evaluate) function snd_func_card_driver
returned error: No such file or directory
ALSA lib confmisc.c:392:(snd_func_concat) error evaluating strings
ALSA lib conf.c:4745:(_snd_config_evaluate) function snd_func_concat
returned error: No such file or directory
ALSA lib confmisc.c:1246:(snd_func_refer) error evaluating name
ALSA lib conf.c:4745:(_snd_config_evaluate) function snd_func_refer
returned error: No such file or directory
ALSA lib conf.c:5233:(snd_config_expand) Evaluate error: No such file or
directory
ALSA lib pcm.c:2660:(snd_pcm_open_noupdate) Unknown PCM default
aplay: main:830: audio open error: No such file or directory
```

- If using a USB audio device, if you see an error like: `"simple.c:283:snd_mixer_selem_get_playback_volume_range: Assertion 'elem' failed."` Check to ensure you have setup `/etc/asound.conf` correctly with no typos.

4. Other Tools Command-line Tools

4.1 Recording (Untested)

If you are using an audio adapter that supports recording (the Zen Hat does not), then use:

1. Record with:
   ```
   (byai)$ arecord -r 44100 -f S16_LE -c 2 testRecording.wav
   ```
2. Things to do to prove out the recording capabilities more:
   - Types of microphones, and mic settings need to be investigated.
   - Volume controls for recording need to be investigated.
   - Tested using an audio cable from Zen headphone jack back into Zen mic port. Able to record poor quality quiet audio using the above command. Recommend using an audio file processing tool (such as GoldWave for Windows) to view recordings while debugging.

4.2 MP3 Player

1. Install the `mpg123` package:
   ```
   (byai)$ sudo apt update
   (byai)$ sudo apt install mpg123
   ```
   - For this to work, you must have internet access. Test by pinging Google.
   - Note that installing this pulls in a number of other packages at the same time.
2. Copy an MP3 file to your NFS share.
3. Play the MP3:
   ```
   (byai)$ mpg123 sample.mp3
   ```
   - You can view the amount of CPU consumed during playback by loading a new terminal to the target:
     ```
     (byai)$ top
     ```
     Or install the program `btop` and use it. It looks much better!
     This will show you the CPU usage of `mplayer` (<1% on my test).
   - You can also use other command-line MP3 players such as `mplayer` (`apt` install it). Takes up ~250 MB to install its libraries.

4.3 Text to Speech

1. Install the text-to-speech engine:
   ```
   (byai)$ sudo apt update
   (byai)$ sudo apt install espeak
   ```
2. Generate a wave file:
   ```
   (byai)$ espeak 'All your bits are belong to us.' -w test.wav
   ```
3. Playback the audio:
   ```
   (byai)$ aplay test.wav
   ```

5. Cross-compile C Program to Play PCM Audio

1. On the *BeagleY-AI*, install `asound`

   ```
   (byai)$ sudo apt update
   (byai)$ sudo apt install libasound2
   ```

   - You can check for the necessary files using:

     ```
     (byai)$ ls /usr/lib/aarch64-linux-gnu/libasound.so.2*
     /usr/lib/aarch64-linux-gnu/libasound.so.2
     /usr/lib/aarch64-linux-gnu/libasound.so.2.0.0
     ```

2. On the *host*, install the `asound` development library (for the header files)

   ```
   (host)$ sudo apt update
   (host)$ sudo apt install libasound2-dev
   ```

3. The *host* will need the `arm64` version of the `asound` library files in order to cross-compile the application. To do this we will install the `arm64` library on the host. (Very useful for cross-compiling libraries!)

   - Setup the host to be able to install `arm64` files and install ALSA:

     ```
     (host)$ sudo dpkg --add-architecture arm64
     (host)$ sudo apt update
     (host)$ sudo apt install libasound2-dev:arm64
     ```

   - Verify the library installed:

     ```
     (host)$ ls /usr/lib/aarch64-linux-gnu/libasound.so
     /usr/lib/aarch64-linux-gnu/libasound.so
     ```

4. Download the `wave_player.c` example from the course website.

   - In a directory on your host, such as in `~/cmpt433/work/pcmExample/`, copy the `wave_player.c` and `Makefile`.
   - Create a `wave-files/` sub-directory of this folder, and copy a wave file into it. For example, copy in the provided drum sounds wave files.
   - Note that the program assumes the files are 16-bit, signed little endian, 44.1kHz, mono files (which is true of the drum sounds). If your sounds are different, you'll need to change the settings in `wave_player.c`

5. Cross-compile the example code by running make:

   ```
   (host)$ cd ~/cmpt433/work/pcmExample/
   (host)$ make
   ```

   - Makefile will build the `wave_player.c` code into `wave_player` and place it in `~/cmpt433/public/myApps/`.
   - It links against the locally installed (on the host) `arm64` library which it will find automatically in `/usr/lib/aarch64-linux-gnu/`
   - The Makefile includes the `-lasound` flag, which is needed for the compiler to link against the `libasound.so` library.
   - The `wav` target in the `Makefile` copies the `wave-files/` folder into the `myApps/` folder.
   - See comments in `wave_player.c` for details on how application works.

6. Run the application:

   ```
   (byai)$ cd /mnt/remote/myApps/
   (byai)$ wave_player
   ```

   - You should hear the drum sound selected in the .c file. The drum sounds are are quite short.
   - For reference, the part of the application which actually sends data to be played is the call to

`snd_pcm_writei()` in the `Audio_playFile()` function.

This call is blocking: it waits until the data has been transmitted to the ALSA sub-system for playback. However, there is some hardware buffering, so the sound may not have actually stopped when `snd_pcm_writei()` returns.

You can use this delay to send more data, hopefully fast enough so that the sound has no jitter. Or, if you want to exit, you may want to call `snd_pcm_drain()` first so that all buffers play out without clipping the end of your wave file.

7. Troubleshooting:
   - When trying to compile, if you get the following error:
     `fatal error: alsa/asoundlib.h: No such file or directory`

     Ensure you have `libasound2-dev` installed on your host PC.
   - When running `wave_player`, if you see:
     `error while loading shared libraries: libasound.so.2: cannot open shared object file: No such file or directory`

     Then you likely need to install `libasound2` on the target.
   - If you don't hear any sound when running `wave_player`, use `aplay` to ensure your hardware is configured correctly (such as not loaded the overlay) and your mixer level is set correctly.

5.1 Cross-compiling ALSA using CMake

The website includes a sample project to cross-compile the above application using CMake. It includes the following changes to `CMakeList.txt` for the component that using the ALSA code (such as your main application):

1. Open your **`app/CMakeLists.txt`** file.
2. Declare that it uses the ALSA library (will automatically find the `arm64` library installed in the above steps):
   ```
   # ALSA support
   find_package(ALSA REQUIRED)
   target_link_libraries(wave_player_cmake LINK_PRIVATE asound)
   ```
3. Copy the wave files folder to your NFS folder:
   ```
   # Copy the folder of WAVE files to NFS
   add_custom_command(TARGET wave_player_cmake POST_BUILD
    COMMAND "${CMAKE_COMMAND}" -E copy_directory
      "${CMAKE_SOURCE_DIR}/wave-files"
      "~/cmpt433/public/myApps/wave-files"
    COMMENT "Copying WAVE files to public NFS directory")
   ```

Change "`wave_player_cmake`" to be the name of your executable.