Zen Cape Audio Guide

by Brian Fraser
Last update: June 26, 2018

This document guides the user through:

1. Getting the audio playback working on the Zen cape.
2. Playing PCM (wave) files via a C program through `asound`.

Table of Contents

Formatting:

1. Host (desktop) commands starting with `$` are Linux console commands:
   ```
   $ echo "Hello world"
   ```
2. Target (board) commands start with #:
   ```
   # echo "On embedded board"
   ```
3. Almost all commands are case sensitive.

Revision History:

- Sept 25: Initial version for 2016. Covers generating new device tree to be able to load audio cape, accessing the cape manage, use hardware floating point libraries, and recovering from corrupted uEnv.txt.
- June 21, 2018: Added suggestion to double check wiring of headphones if no sound.
- June 26, 2018: Added troubleshooting for: `Unrecognized check name ‘unit_address_vs_reg’”`

1. Installing Virtual Audio Cape[1]

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

1.1 Accessing the Cape Manager

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). However, it is also useful to be able to tell Linux, while running, that the hardware is different. For this we use the cape manager.

The BeagleBone's cape manager allows the user to load device tree overlays (DTO) at runtime, from the terminal. We'll use this to tell the kernel about the audio hardware on the Zen cape.

Do the following just once (per board).

1. We'll need access to the cape manager's slots file to list what capes are loaded. Use the following command to have your `~/.profile` file create the environment variable `$SLOTS` to access to the cape manager[2]:
   ```
   # echo SLOTS=/sys/devices/platform/bone_capemgr/slots >> ~/.profile
   ```
2. In the future, you may want to view information about the pins on the board/CPU. Define `PINS` as follows:
   ```
   echo PINS=/sys/kernel/debug/pinctrl/44e10800.pinmux/pins >> ~/.profile
   ```
3. Since the `~/.profile` file is only loaded when you log in, you need to log out and log back in.
   ```
   # exit
   ```
   Then re-login to the BeagleBone.
4. Use `$SLOTS` to display what capes are loaded.
   ```
   # cat $SLOTS
    0: PF----  -1
    1: PF----  -1
    2: PF----  -1
    3: PF----  -1
   ```
   - This shows that there are currently no capes loaded. Later sections will load the audio cape.
5. We'll not need `$PINS` now; but you may need it later to view pin information using:
   ```
   # cat $PINS
   ```

1.2 BeagleBone Black: Unload Conflicting HDMI Cape

If using a BeagleBone Black, complete the following steps.
Skip this sub-section if using a BeagleBone Green.

Do the following just once (per board).

1. Edit `uEnv.txt` which configures what hardware is loaded at startup (make a backup copy):
   ```
   # cp /boot/uEnv.txt /boot/uEnv.bak.hdmi
   # nano /boot/uEnv.txt
   ```
2. Add the following lines to the ***top*** of `uEnv.txt`:
   ```
   # Disable DHMI for Zen audio
   optargs=capemgr.disable_partno=BB-BONELT-HDMI
   ```

[1] Steps based on guide at: http://elinux.org/BBB_Audio_Cape_RevB_Getting_Started

[2] If you are running an old kernel (v3.8) you'll need to instead use:
```
# echo SLOTS=/sys/devices/bone_capemgr*/slots >> ~/.profile
```

3. Reboot:
   ```
   # reboot
   ```
4. Check what capes are now loaded:
   ```
   # cat $SLOTS
    0: 54:PF---
    1: 55:PF---
    2: 56:PF---
    3: 57:PF---
    4: ff:P-O-L Bone-LT-eMMC-2G,00A0,Texas Instrument,BB-BONE-EMMC-2G
    5: ff:P-O-- Bone-Black-HDMI,00A0,Texas Instrument,BB-BONELT-HDMI
    6: ff:P-O-L Bone-Black-HDMIN,00A0,Texas Instrument,BB-BONELT-HDMIN
   ```
   - Note that the HDMI (#5) is "`P-O--`" with no `L`. This means it is not loaded.

1.3 Build New Device Tree

Do the following just once (per board) to prepare the BeagleBone for loading the audio cape.[3]

1. Ensure your BeagleBone has internet access:
   ```
   # ping google.ca
   ```
   - If it fails, see the networking guide on how to enable full internet access to your device.
2. Identify the version of the Linux kernel you are running (need not match output shown here):
   ```
   # uname -r
   4.1.22-ti-r59
   ```
3. Clone the Git repository containing the device tree (source) files from the Linux kernel. Checkout the branch matching your kernel version from the previous step.
   ```
   # cd ~
   # git clone -b 4.1.x https://github.com/RobertCNelson/dtb-rebuilder
   ```
   - Where the branch you clone must match the output of your `uname -r` command:
     - If you have version `4.1.15-ti-rt-r43` use `4.1.x`
     - If you have version `4.1.22-ti-r59` use `4.1.x`
     - If you have version `4.4.12-ti-r31` use `4.4.x`
     - You can see the list of all possible branches:
       ```
       # git ls-remote --heads https://github.com/RobertCNelson/dtb-rebuilder
       ```
       and then select the version most suited to your kernel version (don't include the "`refs/heads/`"). Don't use the `ti` branches.
4. A new device tree is required to support the on-board MMC (for booting the board) and loading the audio cape. Download the required device tree (source) file:[4]
   ```
   # cd dtb-rebuilder/src/arm/
   # wget http://www.cs.sfu.ca/CC/433/bfraser/other/guide-code/cmpt433-bone-audio-mmc.dts
   ```
5. Build the device trees. The makefile assumes the device tree compiler (`dtc`) is at a different location than it may be on your BeagleBone's; hence the need to set `DTC`:
   ```
   # cd ../../
   # DTC=/usr/bin/dtc make all
   ```
   - If it is unable to find the `dtc` program, change the path in the above command to the location found by:
     ```
     # whereis dtc
     ```

[3] This is required because there is a bug in the timer drivers that don't allow you to load the audio cape at runtime with the default configuration. Here we are correcting the default configuration to allow for the cape to be loaded.

[4] Original source http://pastebin.com/raw/vny3wpnq, referenced by this discussion: https://groups.google.com/forum/#!topic/beagleboard/vb3ansvFD6Y

6. Check that the build succeeded:
```
# ls src/arm/cmpt433-bone-audio-mmc.dtb
```
   - If the file is not found, then check for build error messages. See troubleshooting steps.
7. Copy the compiled device tree into the directory where UBoot can load it at boot:
```
# cp src/arm/cmpt433-bone-audio-mmc.dtb /boot/dtbs/4.1.22-ti-r59/
```
   - Changing the target folder to match the version of your kernel.
   - **NOTE: ENSURE YOU COPY THE BINARY (.dtb) FILE, NOT THE SOURCE (.dts)!**
8. Edit `uEnv.txt` to load this device tree at boot (making a backup first):
```
# cp /boot/uEnv.txt /boot/uEnv.bak.audio
# nano /boot/uEnv.txt
```
9. At the top of the file, add the following to load the new device tree:
```
dtb=cmpt433-bone-audio-mmc.dtb
```
   - Ensure that there are no other statements in the file which set the dtb= variable (which would override the above statement).
   - **DOUBLE CHECK THIS STATEMENT!** Getting it wrong can prevent your board from booting.
10. Reboot the board so it loads the new device tree at startup.
```
# reboot
```
   - You may see the following warnings a number of times; you may ignore these (it seems!)
```
PM: am33xx_prepare_push_sram_idle: EMIF function copy failed
```
11. Load the audio-cape, as described in the next sub-section.
12. Troubleshooting
   - If you are unable to clone the `dtb-rebuilder` repository, ensure you have followed the networking guide to give your BeagleBone access to the internet.
   - If the `wget` for the `cmpt433-….dts` file fails with `404 Not Found` then double check the URL is correct (case sensitive). The "`CC`" in the URL will expand to "`CourseCentral`" by the server; this should not be a problem.
   - If building the `cmpt433-bone-audio-mmc.dtb` file fails with the error:
     `"fatal error: am33xx-es2.dtsi: No such file or directory"`

     then edit the file:
```
# nano src/arm/cmpt433-bone-audio-mmc.dts
```
     Delete the line (or comment it out with //):
```
#include "am33xx-es2.dtsi"
```
     Rebuild the project with:
```
# DTC=/usr/bin/dtc make all
```
   - If the build fails with the error:
```
/bin/sh: 1: /usr/local/bin/dtc: not found
… failed
… Error 127
… failed
… Error 2
```
     It likely means you did not correctly specify the path to DTC. Instead of "make all" run:
```
# DTC=/usr/bin/dtc make all
```
   - If your board is unable to boot after editing `/boot/uEnv.txt` then follow the steps in the last section of this guide for *Recovering from Corrupted uEnv.txt*. Double check that you copied the `cmpt433-….dtb` file instead of the .dts file.

- While building the device trees, if you get the error:

  ```
  "FATAL ERROR: Unrecognized check name 'unit_address_vs_reg'"
  ```

  Then update your version of the device tree compiler:

  ```
  # apt-get update
  # apt-get install device-tree-compiler
  ```

1.4 Load Audio Cape

Do this each time you boot the target (unless you do the step to make it permanent).

1. Manually load the audio cape with the following command:
   ```
   # echo BB-BONE-AUDI-02 > $SLOTS
   ```
   - Check it is loaded (`BB-BONE-AUDI`); you may see others if you are on a BeagleBone Black:
     ```
     # cat $SLOTS
      0: PF----  -1
      1: PF----  -1
      2: PF----  -1
      3: PF----  -1
      4: P-O-L-   0 Override Board Name,00A0,Override Manuf,BB-BONE-AUDI-02
     ```
   - Loading the cape must be done each time you reboot the target.
   - You may see the following warning a number of times; it seems you may ignore it:
     ```
     [...] PM: am33xx_prepare_push_sram_idle: EMIF function copy failed
     ```
2. To make the audio virtual cape be loaded each time you reboot, instead of always running the above command, do the following:
   - Edit the cape manager config file:
     ```
     # nano /etc/default/capemgr
     ```
   - Add the following line at the end:
     ```
     CAPE=BB-BONE-AUDI-02
     ```
   - Now when you reboot, it will automatically load the audio cape.
3. Troubleshooting:
   - If you get a `No such file or directory` error when trying to `cat $SLOTS`, you likely have not defined `$SLOTS` in `~/.profile`, or you entered the path incorrectly for your OS.
   - When echoing `BB-BONE-AUDI-02` to the cape manager's `$SLOTS`, if you see:
     ```
     -bash: echo: write error: No such file or directory
     ```
     it likely means that you do not have the `BB-BONE-AUDI-02-00A0.dtbo` file in `/lib/firmware`. Double check it has not been deleted (should be part of standard BeagleBone distribution).

     View the expected file:
     ```
     # ls -la /lib/firmware/BB-BONE-AUDI-02-00A0.dtbo
     -rw-r--r-- 1 root root 2.8K Jun 10 15:49 /lib/firmware/BB-BONE-AUDI-02-00A0.dtbo
     ```
   - If you see the error:
     ```
     -bash: echo: write error: Invalid argument
     ```
     With `dmesg` showing:
     ```
     [...] of_resolve_phandles: Could not find symbol 'clk_mcasp0'
     [...] bone_capemgr bone_capemgr: slot #4: Failed to resolve tree
     ```
     It likely means that your default device tree does not support all the functionality required by audio virtual cape. Follow the directions in the previous section for changing your default device tree to a modified version. Double check your `uEnv.txt` file.
   - If you see the error:
     ```
     -bash: echo: write error: File exists
     ```
     It likely means that the cape is already loaded; check the `$SLOTS` file to see.

     It may means that there is a pin conflict with a different cape; check `dmesg` for info.

2. Configure and Play Audio

1. Plug in speakers or headphones into the audio output on the Zen cape (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.
2. Save a WAVE file to your NFS folder on your host (say, sample.wav).
3. On the target, play the file with:
    ```
    # aplay sample.wav
    ```
4. Change the volume:
    ```
    # 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.
5. Troubleshooting:
    - You can list available playback devices to ensure the configuration succeeded:
      ```
      # aplay -l
      **** List of PLAYBACK Hardware Devices ****
      card 0: B [AudioCape Rev B], device 0: davinci-mcasp.0-tlv320aic3x-hifi \
            tlv320aic3x-hifi-0 []
        Subdevices: 1/1
        Subdevice #0: subdevice #0
      ```
    - You can play back white noise to test the hardware:
      ```
      # speaker-test
      ```
    - If you seem to playback OK 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).
    - If you get the following error with `alsamixer`, it likely means the audio cape is not loaded:
      ```
      cannot open mixer: No such file or directory
      ```
    - If you get the error:
      ```
      Unable to set hw params for playback: Invalid argument
      ```
      It likely means your device tree is incorrect; see Section 1.3.
    - If you get the error:
      ```
      ALSA lib confmisc.c:768:(parse_card) cannot find card '0'
      ...
      ```
      It likely means that you have not loaded the audio virtual cape. Try:
      ```
      # echo BB-BONE-AUDI-02 > $SLOTS
      ```

3. Other Tools Command-line Tools

3.1 Recording (Untested)

1. Record with:
   ```
   # 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.

3.2 MP3 Player

1. Install the `mplayer` package:
   ```
   # apt-get update
   # apt-get install mplayer
   ```
   - For this to work, you must have internet access. Test by pinging Google.
   - Note: 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:
   ```
   # mplayer sample.mp3
   ```
4. You can view the amount of CPU consumed:
   ```
   # mplayer -quiet sample.mp3
   ```
   Then press Ctrl-Z to suspend. Then allow it to keep running in background:
   ```
   # bg
   # top
   ```
   - This will show you the CPU usage of `mplayer` (~30% on my test).
   - When you are done with `top`, press Q . To move `mplayer` back to the foreground, type:
     ```
     #fg
     ```

3.3 Text to Speech[5]

1. Install the text-to-speech engine:
   ```
   # apt-get update
   # apt-get install libttspico-utils
   ```
2. Generate a wave file:
   ```
   # pico2wave -w testAudio.wav 'All your bits are belong to us.'
   ```
3. Playback the audio:
   ```
   # aplay testAudio.wav
   ```

[5] Steps from Molloy 2014, Exploring BeagleBone, Ch12.

4. Play PCM Audio from C

1. On the **host**, install the `asound` development library (for the header files)
   ```
   $ sudo apt-get update
   $ sudo apt-get install libasound2-dev
   ```
2. On the **BeagleBone**, check if `asound` is already installed:
   ```
   # cd /usr/lib/arm-linux-gnueabihf/
   # ls libasound*
   libasound.so.2  libasound.so.2.0.0
   ```
   - If you don't see a libasound.so.2, then install the `asound` library:
     ```
     # apt-get update
     # apt-get install libasound2
     ```
3. Your host will need a copy of `asound`'s `.so` file from the BeagleBone in order to build an application which uses `asound` to run on the target. So copy the `.so` file to the NFS mounted shared folder:
   - On the **host**, create a folder on the shared NFS space
     ```
     $ mkdir ~/cmpt433/public/asound_lib_BBB
     $ chmod a+rwx ~/cmpt433/public/asound_lib_BBB
     ```
   - On the **BeagleBone**, copy the file to the mount and name it `libasound.so`:
     ```
     # cd /usr/lib/arm-linux-gnueabihf/
     # cp libasound.so.2.0.0 /mnt/remote/asound_lib_BBB/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:
   ```
   $ cd ~/cmpt433/work/pcmExample/
   $ make
   ```
   - Makefile will build the `wave_player.c` code into `wave_player` and place it in `~/cmpt433/public/myApps/`. To do this, it needs the `asound` library you copied in previous steps. GCC is told to look for any necessary libraries in the following command already in the Makefile:
     ```
     LFLAGS = -L$(HOME)/cmpt433/public/asound_lib_BBB
     ```
   - The `wav` target in the `Makefile` also 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:
   ```
   # cd /mnt/remote/myApps/
   # 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 either 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`
   - If you don't hear any sound when running `wave_player`, use `aplay` to ensure your hardware is configured correctly.
   - If you see an error:
     ```
     ALSA lib confmisc.c:768:(parse_card) cannot find card '0'
     ```
     it likely means that you don't have the `BB-BONE-AUDI-02` cape loaded.

5. Recovering from Corrupted uEnv.txt

If you edit `/boot/uEnv.txt` and it becomes corrupted, or you load a device tree which does not support the onboard emmc then your board may fail to boot. These steps should help you recover.

1. View your board's boot process using the serial port on the board (via the `screen` program).

   - If you see:

     ```
     Checking for: /uEnv.txt ...
     Checking for: /boot.scr ...
     Checking for: /boot/boot.scr ...
     Checking for: /boot/uEnv.txt ...
     ** Invalid partition 2 **
     ....
     ```

     It likely means you have deleted `/boot/uEnv.txt`

   - If you see:

     ```
     mount: can't find /root in /etc/fstab
     Target filesystem doesn't have requested /sbin/init.
     mount: mounting /dev on /root/dev failed: No such file or directory
     No init found. Try passing init= bootarg.
     ...
     BusyBox v1.22.1 (Debian 1:1.22.0-9+deb8u1) built-in shell (ash)
     Enter 'help' for a list of built-in commands.

     /bin/sh: can't access tty; job control turned off
     (initramfs)
     ```

     It likely means you don't have emmc support correctly loaded in your device tree. Revert to a previous version of `uEnv.txt` (*you do have a backup, right?*) and then change the device tree to be a version which supports the emmc (likely with "emmc" in the file name).

     It may also mean that your on-board emmc has been corrupted and needs to be reflashed.

2. Reboot your board (may need to use reset button on BeagleBone).
   When booting begins, you should see something like:

   ```
   U-Boot 2016.03-00001-g148e520 (Jun 06 2016 - 11:27:44 -0500), Build:
   jenkins-github_Bootloader-Builder-395

          Watchdog enabled
   I2C:   ready
   DRAM:  512 MiB
   Reset Source: Global external warm reset has occurred.
   Reset Source: Global warm SW reset has occurred.
   Reset Source: Power-on reset has occurred.
   MMC:   OMAP SD/MMC: 0, OMAP SD/MMC: 1
   Using default environment

   Net:   <ethaddr> not set. Validating first E-fuse MAC
   cpsw, usb_ether
   Press SPACE to abort autoboot in 2 seconds
   =>
   ```

3. Press SPACE as soon as it begins booting to enter the UBoot prompt.

4. Listing files in the `/boot/` folder:

   ```
   => ext4ls mmc 1:1 /boot
   ```

5. Copy a file, changing `SOURCE` and `TARGET` as needed:
   ```
   => ext4load mmc 1:1 0x82000000 SOURCE
   => ext4write mmc 1:1 0x82000000 TARGET ${filesize}
   ```
   - For example, to make a backup copy of your current `uEnv.txt` use:
     ```
     => ext4load mmc 1:1 0x82000000 /boot/uEnv.txt
     => ext4write mmc 1:1 0x82000000 /boot/uEnv.bak.uboot ${filesize}
     ```
   - For example, to restore `/boot/uEnv.bak.audio` use:
     ```
     => ext4load mmc 1:1 0x82000000 /boot/uEnv.bak.audio
     => ext4write mmc 1:1 0x82000000 /boot/uEnv.txt ${filesize}
     ```
   - Note the `${filesize}` variable is set when you do an `ext4load` command.
6. Boot the board, which loads `/boot/uEnv.txt`:
   ```
   => boot
   ```
7. Troubleshooting:
   - You can view the contents of a file using:
     ```
     => ext4load mmc 1:1 0x82000000 /boot/uEnv.txt
     => md 0x82000000
     ```