Guideline for TAS2781 Linux driver based on BBB

Kernel-5.10-rt

Revision history

Ver	Date	Author	Description
1.0	2022/9/14		Initial
1.1	2022/9/20		1st round Review & rework
1.2	2022/9/30		Add tinyalsa
1.3	2022/10/8		2nd round Review & rework
1.4	2022/10/10		Add CONFIG_COMPAT and FTC section

Information

tasdevice Driver	tas2781
7-bit I^2C Address	0x38 ~ 0x3F
Platform	Qualcomm/Mtk/LSI/BeagleBone Black/AMBA
Architecture	ALSA/Tiny-ALSA
Bus type	I2C
Kernel Version	V4.19 ~ V5.10

Release Package

Driver Package	Kconfig, Makefile	Config for compiling
	.c, & .h,	Source code
	dts.readme	Sample for dts setting
	regbin.json	File to generate a sample register setting file with Regbin tool.
	dspfw.json	File to generate a sample dsp firmware and acoustic params file with PPC3 tool.
	Verification report	
	.ftcfg file	Store speaker basic characterization params for speaker calibration
Tool_utility	FTC_tool	Source code for calibration
	PPC3 tool	Tool for sample dsp firmware and acoustic params and ftcfg file
	Non_Integrated_Bin_Tool_v1.3.4	Tool for regbin

Device Tree | I2C

- Configuration in DTS:

```
i2c2_pins: pinmux_i2c2_pins {
    pinctrl-single,pins = <
        AM33XX_PADCONF(AM335X_PIN_UART1_CTSN, PIN_INPUT_PULLUP, MUX_MODE3)
        AM33XX_PADCONF(AM335X_PIN_UART1_RTSN, PIN_INPUT_PULLUP, MUX_MODE3)
    >;
};
```

```
&i2c2 {
    pinctrl-names = "default";
    #address-cells = <1>;
    #size-cells = <0>;
    status = "okay";
    clock-frequency = <400000>;
    symlink = "bone/i2c/2";
    pinctrl-0 = <&i2c2_pins>;
    tasdevice: tasdevice@38 {
        status = "okay";
        #sound-dai-cells = <0>;
        compatible = "ti,tas2781";
        reg = <0x38>;
        ti,topleft-channel = <0x38>;
        ti,reset-gpio0 = <&gpio1 10 GPIO_ACTIVE_HIGH>;
        ti,irq-gpio = <&gpio1 15 0>;
    };
};
```

- How to check device

If the I2C is successfully registered, Check device through below commands, 2-0038 is the registered device. 0x38 is the i2c address for slave device.

ls /sys/bus/i2c/device/

```
debian@beaglebone:~$ ls /sys/bus/i2c/devices/
0-0024  0-0050  2-0038  i2c-0  i2c-2
```

- DTS file:

am335x-boneblack-hdmi.dtsi

Device Tree | SPI

- Configuration in DTS:

```
bb_spi0_pins: pinmux_bb_spi0_pins {
    pinctrl-single,pins = <
        AM33XX_PADCONF(AM335X_PIN_SPI0_SCLK, PIN_INPUT, MUX_MODE0)
        AM33XX_PADCONF(AM335X_PIN_SPI0_D0, PIN_INPUT, MUX_MODE0)
        AM33XX_PADCONF(AM335X_PIN_SPI0_D1, PIN_INPUT, MUX_MODE0)
        AM33XX_PADCONF(AM335X_PIN_SPI0_CS0, PIN_INPUT, MUX_MODE0)
    >;
};
  &i2c2 {
      pinctrl-names = "default";
      #address-cells = <1>;
      #size-cells = <0>;
      status = "okay";
      clock-frequency = <400000>;
      pinctrl-0 = <&i2c2_pins>;
      tasdevice: tasdevice@38 {
          status = "okay";
          #sound-dai-cells = <0>;
          compatible = "ti,tas2781";
          reg = <0x38>;
          ti,topleft-channel = <0x38>;
          ti,reset-gpio0 = <&gpio1 10 GPIO_ACTIVE_HIGH>;
          ti,irq-gpio = <&gpio1 15 0>;
      };
  };
```

- How to check device.

If the SPI is successfully registered, Check device through below commands, spi0.0 is the registered device.

ls /sys/bus/spi/device/

```
debian@beaglebone:~$ ls /sys/bus/spi/devices/
spi0.0
```

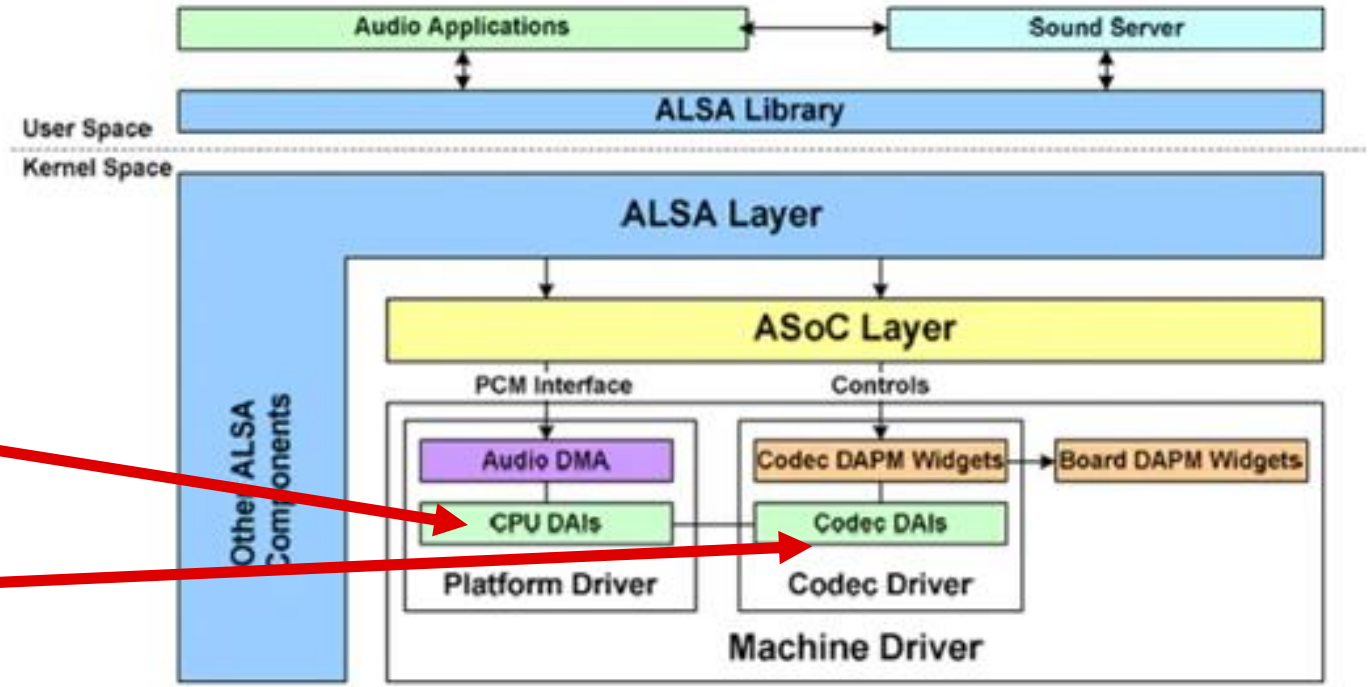
- DTS file:

am335x-boneblack-hdmi.dtsi

Device Tree | Sound card

```
&mcasp0 {
    #sound-dai-cells = <0>;
    pinctrl-names = "default";
    pinctrl-0 = <&mcasp0_pins>;
    status = "okay";
    op-mode = <0>;  /* MCASP_IIS_MODE */
    tdm-slots = <8>;
    serial-dir = <  /* 0: INACTIVE, 1: TX, 2: RX */
        2 2 1 0
    >;
    tx-num-evt = <32>;
    rx-num-evt = <32>;
};
sound {
    compatible = "simple-audio-card";
    simple-audio-card,name = "TI BeagleBone Black";
    simple-audio-card,dai-link@0 {
        format = "dsp_a";
        bitclock-master = <&sound0_master>;
        frame-master = <&sound0_master>;
        sound0_master: cpu {
            sound-dai = <&mcasp0>;
            clocks = <&clk_mcasp0>;
        };

        codec {
            sound-dai = <&tasdevice>;
        };
    };
};
```



Kconfig & Makefile

Kconfig in sound/soc/codecs

```
config SND_SOC_I2C_AND_SPI
    tristate
    default m if I2C=m
    default y if I2C=y
    default y if SPI_MASTER=y

menu "CODEC drivers"

comment "tas-device"
source "sound/soc/codecs/tasdevice/Kconfig"

imply SND_SOC_TAS571X
imply SND_SOC_TAS5720
imply SND_SOC_TAS6424
imply SND_SOC_TASDEVICE
imply SND_SOC_PCMDEVICE
imply SND_SOC_TDA7419
imply SND_SOC_TFA9879
```

Makefile in sound/soc/codecs

```
obj-$(CONFIG_SND_SOC_TASDEVICE) += tasdevice/
obj-$(CONFIG_SND_SOC_PCMDEVICE) += pcmdevice/
obj-$(CONFIG_SND_SOC_88PM860X)  += snd-soc-88pm860x.o
obj-$(CONFIG_SND_SOC_AB8500_CODEC)  += snd-soc-ab8500-codec.o
obj-$(CONFIG_SND_SOC_AC97_CODEC)    += snd-soc-ac97.o
obj-$(CONFIG_SND_SOC_AD1836)    += snd-soc-ad1836.o
```

Modification in defconfig for I2C interface

- Check whether following items have been enabled in arch/arm/configs/ti_sdk_am3x_release_defconfig, if not, enable them as following
  - CONFIG_I2C=y
  - CONFIG_I2C_GPIO=y
  - CONFIG_I2C_OMAP=y
  - CONFIG_I2C_MUX=y

Modification in defconfig for SPI interface

- Check whether following items have been enabled in arch/arm/configs/ti_sdk_am3x_release_defconfig, if not, enable them as following
  - CONFIG_SPI=y
  - CONFIG_SPI_GPIO=y
  - CONFIG_SPI_SPIDEV=y
  - CONFIG_SPI_MASTER=y

Modification in defconfig for Sound Card based on I2C or SPI

- Add following items in arch/arm/configs/ti_sdk_am3x_release_defconfig.
  - CONFIG_SOUND=y
  - CONFIG_SND=y
  - CONFIG_SND_SOC=y
  - CONFIG_SND_SIMPLE_CARD=y
  - CONFIG_SND_SOC_TASDEVICE=y
  - CONFIG_TASDEVICE_CODEC=y

CONFIG_COMPAT

- Enable CONFIG_COMPAT on 64-bit kernel
  - The whole point of CONFIG_COMPAT is to allow running 32-bit userspace code on a 64-bit kernel. So no, it doesn't need to enable on a 32-bit kernel.

Firmware Setting in defconfig

- Compile the bin file into image.
  - Add following settings into defconfig file

```
CONFIG_FW_LOADER=y
CONFIG_EXTRA_FIRMWARE_DIR="firmware"
CONFIG_EXTRA_FIRMWARE="tas2781_regbin.bin tas2781_dsp.bin tas2781_cal_38.bin"
```

    - CONFIG_EXTRA_FIRMWARE_DIR is path where the bin file is stored into the kernel root. “firmware” is the relative path in the compiling base path.
    - CONFIG_EXTRA_FIRMWARE is the name of firmware name. if multiple bin files are required, input the file name format like CONFIG_EXTRA_FIRMWARE = “a.bin b.bin c.bin”
    - If this method is enabled, the fw will be loaded during system bootup.
  - During debug, pushing the bin file into proper folder of the target device is more convenient than compiling into image. Once compiling into zImage, every time the bin file is changed, it should be recompiled into zImage instead of copied into the *fw_path*

Firmware & ftcfg file naming

file	File name	Remark
regbin	tas2781_regbin.bin	Generated by regbin tool
Dsp firmware & params	tas2781_dsp.bin	Generated by ppc3
ftcfg	tas2781_0x38.ftcfg tas2781_0x39.ftcfg …	Generated by ppc3
Calibrated file	tas2781_cal_0x38.bin tas2781_cal_0x39.bin …	Generated by ftc tool

Compile Image with tas2781 driver

- Generate .config
  - #sudo make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- ti_sdk_am3x_release_defconfig
- Compile the image
  - #sudo make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- zImage dtbs -j16

Audio card registration

- BBB use mcasp0 to output the audio data. If the DTS was configured correctly, the driver will create sound card device and kcontrols, we can check below commands to confirm that.

ls /dev/snd

ls /sys/bus/i2c/devices/2-0038

cat /proc/asound/pcm

cat /proc/asound/cards

```
debian@beaglebone:~$ ls /dev/snd/
by-path  controlC0  pcmC0D0c  pcmC0D0p  timer
debian@beaglebone:~$ ls /sys/bus/i2c/devices/2-0038
act_addr  driver          fwload    of_node  regbininfo_list  subsystem
dev_addr  dspfw_config    modalias  power    regcfg_list      uevent
devinfo   dspfwinfo_list  name      reg      regdump
debian@beaglebone:~$ cat /proc/asound/pcm
00-00: davinci-mcasp.0-tasdevice_codec tasdevice-codec.2-0038-0 : davinci-mcasp.0-tasdevice_codec tasdevice-codec.2-0038-0 : playback 1 : capture 1
debian@beaglebone:~$ cat /proc/asound/cards
 0 [Black          ]: TI_BeagleBone_B - TI BeagleBone Black
                      TI BeagleBone Black
```

Audio card | Kcontrols

- Please use below commands to check Kcontrols created in driver.

#amixer contents

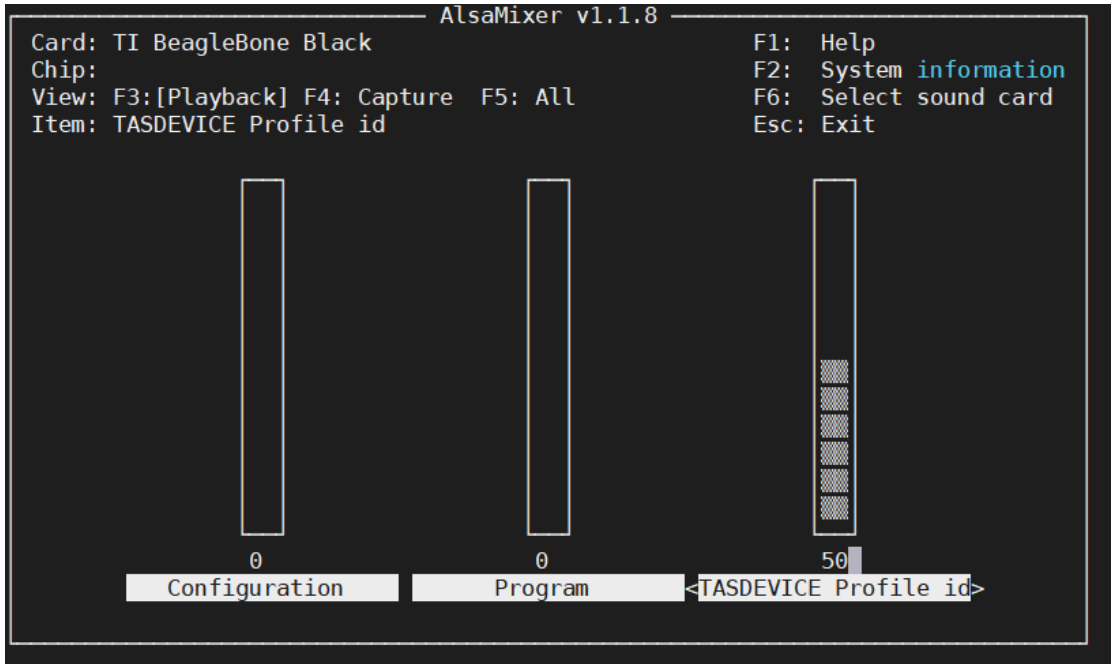
```
debian@beaglebone:~$ amixer contents
numid=3,iface=MIXER,name='Configuration'
  ; type=INTEGER,access=rw------,values=1,min=0,max=1,step=0
  : values=0
numid=2,iface=MIXER,name='Program'
  ; type=INTEGER,access=rw------,values=1,min=0,max=1,step=0
  : values=0
numid=1,iface=MIXER,name='TASDEVICE Profile id'
  ; type=INTEGER,access=rw------,values=1,min=0,max=6,step=1
  : values=0
```

#tinymixer contents

```
debian@beaglebone:~$ tinymix contents
Number of controls: 3
ctl     type    num     name                                value
0       INT     1       TASDEVICE Profile id                0 (range 0->6)
1       INT     1       Program                             0 (range 0->1)
2       INT     1       Configuration                       0 (range 0->1)
```

- Also we can use Graphical interface to adjust these kcontrols. Push esc to exit.

#alsamixer



Driver nodes I | Introduction

- In order to debug driver freely, several driver nodes have been defined,

```
debian@beaglebone:/sys/bus/i2c/devices/2-0038$ ls
act_addr  driver          fwload    of_node  regbininfo_list  subsystem
dev_addr  dspfw_config    modalias  power    regcfg_list      uevent
devinfo   dspfwinfo_list  name      reg      regdump
```

- Check below path to access above driver nodes
  - /sys/bus/i2c/devices/2-0038
  - /sys/class/i2c-adapter/i2c-2/2-0038
  - /sys/class/i2c-dev/i2c-2/device/2-0038

Driver nodes II | i2caddr/devinfo

- #cat dev_addr

  Illustration: show the active i2c address

```
debian@beaglebone:/sys/bus/i2c/devices/2-0038$ cat dev_addr
ndev:1
addr0:38
```

- #cat devinfo

  Illustration: Get the basic information of audio device on the board

```
debian@beaglebone:/sys/bus/i2c/devices/2-0038$ cat devinfo
No.     DevTyp  Addr
0       audev   0x38
```

Driver nodes II | reg

- #echo chn 0xBK 0xPG 0xRG 0xXX > reg

  Illustration: Write a value to a certain register

  – chn is channel no, must be 1-digital
  – BK, PG, RG & XX must be 2-digital HEX
  – eg: 0 0x00 0x00 0x05 0x07 > reg

- #cat reg

  Illustration: Read back the value from the register which have been echoed before

```
debian@beaglebone:/sys/bus/i2c/devices/2-0038$ echo 0 0x00 0x00 0x05 0x07 > reg
debian@beaglebone:/sys/bus/i2c/devices/2-0038$ cat reg
i2c - addr: 0x38
Chn0B0x00P0x00R0x05:0x07
```

Driver nodes III | regdump

- #echo chn 0xBK 0xPG > regdump

  Illustration: the command dump all the registers of the specific page

  – chn is channel no, must be 1-digital

  – BK & PG must be 2-digital HEX

- #cat regdump

  Illustration: run the echo command, show the 7-bit i2c address of the chip and dump the registers

PAGE00-regdump

PAGE01-regdump

Driver nodes IV | regbininfo_list &

- #cat regbininfo_list

  Illustration: list the regbin version and dump the name of all the audio cases from regbin file, If wanted detailed info from specific audio case, kindly use the drive node **regcfg_list**

```
debian@beaglebone:/sys/bus/i2c/devices/2-0038$ cat regbininfo_list
Regbin File Version: 0x0105
conf 00: I2S-16bit-echoreference-slot0
conf 01: I2S-16bit-echoreference-slot2
conf 02: I2S-16bit-IV-slot0
conf 03: TDM-16bit-echoreference-slot0
conf 04: dsp-16bit-slot0
conf 05: echo-ref
```

- #cat dspfwinfo_list

  Illustration: list the detail of fwdsp bin file and dump the program name and configuration name.

```
debian@beaglebone:/sys/bus/i2c/devices/2-0038$ cat dspfwinfo_list
tas2781_dsp.bin
Git format
Generated by PPC3V0x14200

ndev: 1
mnPrograms: 1
        ProgramName:    Tuning Mode
mnConfigurations: 2
        nConfig:0
        ConfigName:configuration_audio_Tuning Mode_48 KHz_s1_0  Program:0x00

        nConfig:1
        ConfigName:calibration_Tuning Mode_48 KHz_s1_0  Program:0x00
```

Driver nodes V | regcfg_list

```
debian@beaglebone:/sys/bus/i2c/devices/2-0038$ echo 00 >regcfg_list
debian@beaglebone:/sys/bus/i2c/devices/2-0038$ [ 8108.293276] tasdevice-codec 2-0038: regcfg: count = 3
[ 8108.293320] tasdevice-codec 2-0038: [regcfg_list]cfg= 0, cnt=3

debian@beaglebone:/sys/bus/i2c/devices/2-0038$ cat regcfg_list
Conf 00: I2S-16bit-echoreference-slot0
block type:PRE_POWER_UP  device idx = 0x00
                SINGLE BYTE:
                        BOOK0x00 PAGE0x00 REG0x0e VALUE = 0xc4
                        BOOK0x00 PAGE0x00 REG0x0f VALUE = 0x40
                        BOOK0x00 PAGE0x00 REG0x5c VALUE = 0xd9
                        BOOK0x00 PAGE0x00 REG0x0d VALUE = 0x01
                        BOOK0x00 PAGE0x00 REG0x10 VALUE = 0x04
                        BOOK0x00 PAGE0x00 REG0x16 VALUE = 0x40
                        BOOK0x00 PAGE0x01 REG0x17 VALUE = 0xc8
                BURST:
                        BOOK0x00 PAGE0x04
                        REG0x30 = 0x00 REG0x31 = 0x00 REG0x32 = 0x00 REG0x33 = 0x01
                BURST:
                        BOOK0x00 PAGE0x08
                        REG0x18 = 0x00 REG0x19 = 0x00 REG0x1a = 0x00 REG0x1b = 0x00
                SINGLE BYTE:
                        BOOK0x00 PAGE0x08 REG0x1c VALUE = 0x00
                BURST:
                        BOOK0x00 PAGE0x08
                        REG0x28 = 0x40 REG0x29 = 0x00 REG0x2a = 0x00 REG0x2b = 0x00
                BURST:
                        BOOK0x00 PAGE0x0a
                        REG0x48 = 0x00 REG0x49 = 0x00 REG0x4a = 0x00 REG0x4b = 0x00
                BURST:
                        BOOK0x00 PAGE0x0a
                        REG0x58 = 0x40 REG0x59 = 0x00 REG0x5a = 0x00 REG0x5b = 0x00
                SINGLE BYTE:
                        BOOK0x00 PAGE0x00 REG0x02 VALUE = 0x00
block type:PRE_SHUTDOWN  device idx = 0x00
                SINGLE BYTE:
                        BOOK0x00 PAGE0x00 REG0x02 VALUE = 0x02
```

- #echo CG > regcfg_list
  - CG is conf NO, it should be 2-digital decimal
  - eg: echo 00 > regcfg_list
- #cat regcfg_list
  - Illustration: dump the register setting of the audio case specified by echo command

Bypass mode

- Write a shell script to test bypass mode. Use #amixer contents to check the kcontrol list, then set the parameters we used. “Program” is 0 when only use tuning mode.
  - PS: for detail about porting Tinyalsa, please see appendix.

Shell script with Alsa

```
#! /bin/bash
amixer set "Program" 1
amixer set "Configuration" 0
amixer set “TASDEVICE Profile id” 0
aplay --device="hw:0,0" test.wav &
sleep 1
amixer set “TASDEVICE Profile id” 5
arecord -c 2 -f S16_LE -r 48000 -d 15 --device="hw:0,0" 16bit-echoref.wav &
```

Shell script with Tinyalsa

```
#! /bin/bash
tinymix set "Program" 1
tinymix set "Configuration" 0
tinymix set "TASDEVICE Profile id" 0
tinyplay test.wav &
sleep 1
tinymix set 'TASDEVICE Profile id' 5
tinycap 16bit-echoref.wav &
```

Tuning mode

- Write a shell script to test tuning mode. “Program” is 1 when use tuning mode, then select the configuration number.
  - PS: For details about fwdsp bin file, please use “cat dspfwinfo_list”.

Shell script with Alsa

```
#! /bin/bash
amixer set "Program" 0
amixer set "Configuration" 0
amixer set 'TASDEVICE Profile id' 4
aplay --device="hw:0,0" test.wav &
sleep 1
amixer set 'TASDEVICE Profile id' 5
arecord -c 2 -f S16_LE -r 48000 -d 15 --device="hw:0,0" 16bit-echoref.wav &
```

Shell script with Tinyalsa

```
#! /bin/bash
tinymix set "Program" 0
tinymix set "Configuration" 0
tinymix set "TASDEVICE Profile id" 4
tinyplay test.wav &
sleep 1
tinymix set 'TASDEVICE Profile id' 5
tinycap 16bit-echoref.wav &
```

FTC Tool Package

- Makefile in the package is for arm-linux-gnuebinf
  - Generate tool

    # make
- Makefile.x86 in the package is for x86
  - Generate tool

    # make –f Makefile.x86
- Assign permissions to tas2781_ftc

  # chmod 777 tas2781_ftc

FTC tool

- Run FTC tool with root permission
  - # sudo ./tas2781_ftc
- Put TAS2781-A.ftcfg file (Generated by ppc3) into direction: `/mnt/vendor/persist/audio/`
- After the calibration process finished, the results save into /home/audio/ti/tas2781_cal.txt

```
debian@beaglebone:~$ cat /home/audio/ti/tas2781_cal.txt
Ambient temperature = 20.00

Dev[0] Re = 13.41
Dev[0] rms_pow    = 0x00000000
Dev[0] t_limit    = 0x00000000
Dev[0] Result = 0x1
```

Calibration test

- It’s necessary to enable IV sense in PPC3.
- Write a shell script to test calibration. Select calibration_tuning_mode 1 in “configuration”.

Shell script with Alsa

```
#! /bin/bash
amixer set "Program" 0
amixer set "Configuration" 1
amixer set 'TASDEVICE Profile id' 4
aplay --device="hw:0,0" silence.wav &
sleep 15
./tas2781_ftc
```

Shell script with Tinyalsa

```
#! /bin/bash
tinymix set "Program" 0
tinymix set "Configuration" 1
tinymix set "TASDEVICE Profile id" 4
tinyplay silence.wav &
sleep 15
./tas2781_ftc
```

Appendix

Kernel 5.10-rt

- Download Kernel 5.10-rt for BBB from…
  - https://github.com/beagleboard/linux/tree/5.10-rt
- # sudo unzip kernel-5.10-rt

Porting Tinyalsa I

- Reference link:
  Tinyalsa - Tiny library to interface with ALSA in the Linux kernel - (tinyalsa) (opensourcelibs.com)
  Download link:
  GitHub - tinyalsa/tinyalsa: Tiny library to interface with ALSA in the Linux kernel
- copy driver code into /usr/local on BBB.

Porting Tinyalsa II | Steps of installation

1. Modify cross compile, add below statements into top of Makefile.
   export ARCH = arm
   export CROSS_COMPILE = arm-linux-gnueabihf-
   export CC = $(CROSS_COMPILE)gcc
2. Assign permissions to scripts in tinyalsa package
   #chmod 777 script
3. To build and install with Make, run the commands:
   # sudo make
   # sudo make install
   # sudo ldconfig
4. Once installed, the man pages are available via:
   # man tinyplay
   # man tinycap
   # man tinymix
   # man tinypcminfo

THANKS!