

SBOA239-September 2017

TAS2557 and TAS2559 End-System Integration Guide

ABSTRACT

This document describes how to integrate the TAS2557 and TAS2559 devices into an end system.

Contents

1	End-System Integration	2
	Workflow	
3	Dump Binary File	3
4	Device Driver Integration	8
5	Tuning and Debugging in System	17

List of Figures

1	System Integration Page	2
2	End-System Integration Workflow	2
3	Dumping Binary File – Configuration Selection	3
4	Dumping Binary File – Snapshot Selection (1 of 2)	3
5	Dumping Binary File – Snapshot Selection (2 of 2)	4
6	Dumping Binary File – Take a New Snapshot	4
7	Dumping Binary File – Select Device in the Snapshot	4
8	Dumping Binary File – Choose Target Device	5
9	Dumping Binary File – A-SI Record Channel Selection	5
10	Dumping Binary File – Enter DDC Name	5
11	Factory Test and Calibration	
12	Summary Page	6
13	Dumped Files	7
14	Tuning and Debugging in System	18
15	DDC Selection for Tuning on End System	18
16	DDC Selection for Tuning on End System	19

List of Tables

Trademarks

Android is a trademark of Google, Inc. All other trademarks are the property of their respective owners.



End-System Integration

1 End-System Integration

The System Integration feature is used to dump the binary and coefficients required to program TAS2557 and TAS2559 device based on the Tuning Snapshots created in Audio Processing. This feature can also be used to debug/tune audio on Android[™] phone.

Figure 1 shows the System Integration page of the TAS2557 and TAS2559 application.

NOTE: The PPC3 screenshots and the information provided in this document are based on the PPC3 version 3.1.12.

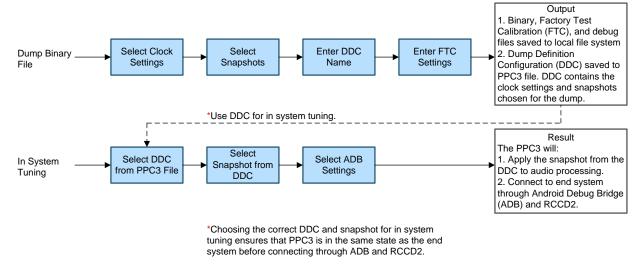
Select Your Option Audio Tuning and Debugging	
Dump Definition Configuration(DDC) Selection	•••
Choose the DDC that is loaded on the device	
 test-dump test-dump test-dump test-dump test-dump test-dump test-dump 	
Prev	Next

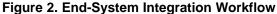


2 Workflow

2

Figure 2 shows the workflow of both options in the end-system integration.







3 Dump Binary File

This section details the dump binary file wizard, the files that will be dumped, and how to use them.

3.1 Dump Binary Wizard

Follow these steps to dump the binary file:

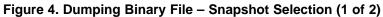
 Click Dump the binary file option and click Next. Configuration Selection page will appear as shown in Figure 3. In this page the user can choose the Sampling Frequency and Clocks that are available in the end system.

Configuration Se	lection						◎●_●
Choose the settings to cr	eate the binary f	ile					
pplication	-						
Tuning Mode	Application	Sampling Frequency	Clock Source	Clock Frequency	Base		
ampling Frequency		48 KHz	BCLK	1.536 MHz	۲	a	
Clock Source	Tuning Mode	16 KHz	MCLK	12.288 MHz	0		
Clock Frequency	~						
	~						
	bd						
Ad							
Burst Size (Bytes)	I2C slave address						

Figure 3. Dumping Binary File – Configuration Selection

2. Set the desired configuration values and click Next. Then, the Snapshot Selection wizard appears (see Figure 4 and Figure 5).







Dump	Binary	File
------	--------	------

Snapshot S	election		• • • • •
	hot with which the binary fil	e should be created.	
16 KHz BCLK 3.072 M	IHz (Base)		
🗹 🜖 Snapshot 1 🖠	N (B)	🗹 🕗 Snapshot 2 💦 🔢	
		Stereo Linking will be disabled when this snapshot is dumped as as Configuration	
Base snapshot	Snapshot 1 🗸		
Take Snapshot			
	onfiguration settings (Dump Defini a name for the DCC settings for wh	tion Configuration - DDC) and saving it along with .ppc3 file to enable you to t hich binary files are generated.	une in device later.
2557_dump			
Choose path to save g	enerated binary files		
TAS2557	F Browse		
Prev			Next

Figure 5. Dumping Binary File – Snapshot Selection (2 of 2)

3. Choose an existing snapshot from the options available (if any) or take a new snapshot by clicking the Take Snapshot button (shown in Figure 6).

Choose the shapshot with which the binary file should be created.

Application: Tuning Mode

No snapshots available for selection

O Take Snapshot

Figure 6. Dumping Binary File – Take a New Snapshot

4. Choose the devices that will be used in the snapshot for the TAS2557 Dual Mono and TAS2559 applications.

Choose the snapshot with which the binary file should be created.

Application: Tuning Mode

48 KHz BCLK 3.072 MHz (Base)





Stereo Linking will be disabled when this snapshot dumped as as Configuration

Figure 7. Dumping Binary File – Select Device in the Snapshot



5. Choose the Base snapshot that becomes the first Tuning Configuration in the End System. This configuration can be set as the boot configuration in the end system.

Base snapshot	Snapshot 1	~	
---------------	------------	---	--

Figure 8. Dumping Binary File – Choose Target Device

 Change the ASI Record Channel by clicking the settings icon in the top-right corner (see Figure 9). Choose the record channel from the popup. (This is applicable only for the TAS2557, not for the TAS2559.)

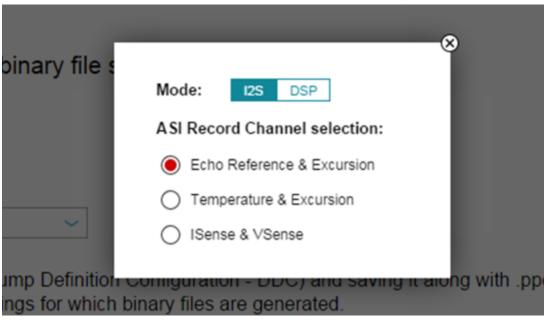


Figure 9. Dumping Binary File – A-SI Record Channel Selection

7. Enter the DDC name and path once a snapshot is chosen, then click Next (see Figure 10).



Figure 10. Dumping Binary File – Enter DDC Name

The data captured in the Factory Test and Calibration page generate a <DDC Name>.ftcfg file at the end of the wizard. A device driver on the end system uses the .ftcfg file to run the factory calibration.



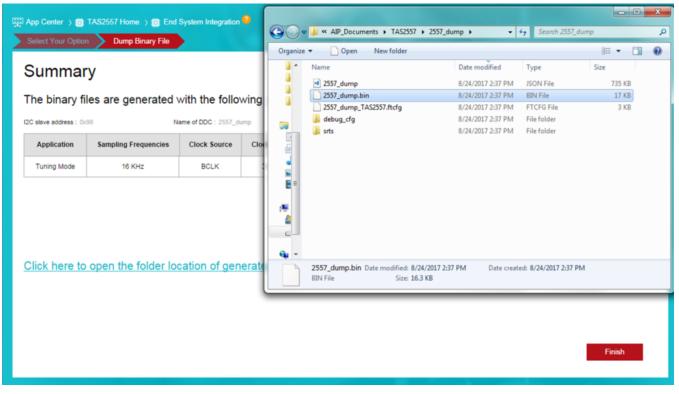
Dump Binary File

Your Option 💙 Dump Binary File

Characterization F	arameters										
Re	: 6.7 Ohr	n	CMWF	: 89	1 Hz	BI	: 0.8141	ſm	Mms	: 0.0667	g
Rtv	: 63.4 K/	N	Rtm	: 56	6.7 K/W	Rtva	: 136 K/	W	System Gain(DAC+Am	np) : 9.35 Vo	It/F.S
Device Non-linearity	: 1.5 %		Thermal Limit	: 80	∆°C	PIG	: 1 (dB)		Temperature Coefficier	nt : 0.0033	K-1
Sampling Frequency	: 16000	łz									
Pass / Fail Limits											4
Pass / Fail Limits	7.975	Ohm	Re Low	6.525	Ohm	F0 High	924	Hz	F0 Low	616	Hz
	7.975	Ohm	Re Low Q Low	6.525	Ohm	F0 High Temperature High	924	Hz Δ°C	F0 Low Temperature Low	616	
Re High	1.704	Ohm			Ohm						Hz

Figure 11. Factory Test and Calibration

Figure 12 shows the result of binary file and associated file generation.







3.2 Overview of Files Dumped by PPC3

Figure 13 shows the dumped files.

Organize	▼ Include in library ▼ Share with ▼	New folder)II 🕶 🔟 🔞
☆ Â	Name	Date modified Type	Size
Щ Маралан Сан Маралан Сан Сан Сан С С С С С С С С С С С С С	EVMDDC	8/24/2017 2:37 PM JSON File	735 KB
	EVMDDC.bin	8/24/2017 2:37 PM BIN File	17 KB
9	EVMDDC_TAS2557.ftcfg	8/24/2017 2:37 PM FTCFG File	3 KB
	퉬 debug_cfg	8/24/2017 2:37 PM File folder	
	鷆 srts	8/24/2017 2:37 PM File folder	
	5 items		

Figure 13. Dumped Files

3.2.1 Description of Files Dumped

3.2.1.1 EVMDDC.bin

The EVMDDC.bin file is recommended for use in platforms with file systems. The file contains all required information (except power and mute sequence) to program the TAS2557 and TAS2559 devices. Most Android phones with the TAS2557 and TAS2559 use this file format.

3.2.1.2 EVMDDC.ftcfg

Device driver uses this file to get the characterization data of the speaker to run the factory line calibration process on the end system.

3.2.1.3 Debug Files

The following debug files are dumped in addition to the EVMDDC.ftcfg and EVMDDC.bin files: EVMDDC.json, debug_cfg, debug.cfg, and more.



Device Driver Integration

4 Device Driver Integration

This section describes the sequence of programming by the device driver in the end system. The sequence of programming depends on the state in which the end system is currently in. This section lists the sequence of programming for each such state.

Example device driver code is available with TI. Users can use the example driver to port to their platform.

The dumped .bin file has various blocks (see the following code snippet).

Block ty	pes within the binary file are:	
#define	TAS2557_BLOCK_PLL	0x00
#define	TAS2557_BLOCK_PGM_ALL	0x0d
#define	TAS2557_BLOCK_PGM_DEV_A	0x01
#define	TAS2557_BLOCK_PGM_DEV_B	0x08
#define	TAS2557_BLOCK_CFG_COEFF_DEV_A	0x03
#define	TAS2557_BLOCK_CFG_COEFF_DEV_B	0x0a
#define	TAS2557_BLOCK_CFG_PRE_DEV_A	0x04
#define	TAS2557_BLOCK_CFG_PRE_DEV_B	0x0b
#define	TAS2557_BLOCK_CFG_POST	0x05
#define	TAS2557_BLOCK_CFG_POST_POWER	0x06
#define	TAS2559_BLOCK_PST_POWERUP_DEV_B	0x0e
#define	TAS2557_BLOCK_CFG_CAL_A	0x10
#define	TAS2557_BLOCK_CFG_CAL_B	0x20

Because of different applications (such as TAS2557, TAS2557 Dual-Mono, and TAS2559), it is possible that not all blocks are present in one application. If a block is not there, that step can be skipped.

Use the following references for Section 4.1 through Section 4.5.

[A]: applies to device A.

[B]: applies to device B.

[AB]: applies to device A and device B. For TAS2559 applications, device A refers to the TAS2559, and device B refers to the TAS2560.

4.1 Case One

Programming the TAS2557 and TAS2559 for the first time:

Use case: device initialization during system boot up.

In this case, the DSP program memory and coefficiency memory are empty. For better understanding, use configuration 0 as default tuning data.

Steps to playback:

1. [AB]Hardware reset

Hardware reset is strongly recommended for reliable operations afterward.

- 2. [AB]Software reset
- 3. Initialize

8

This is not part of the binary file. This initialization is about ASI format, IRQ configuration, and other configurations.

[A]TAS2557 mono applications:

See tas2557-android-driver.

int tas2557_load_default(struct tas2557_priv *pTAS2557)

[AB]TAS2557 stereo applications:

See tas2557-stereo-driver.

int yas2557_load_default(struct tas2557_priv *pTAS2557)

[AB]TAS2559 applications:

See tas2559-android-driver.

```
int tas2559_load_default(struct tas2559_priv *pTAS2559)
```



- 4. Download the program.
 - [AB]Load TAS2557_BLOCK_PGM_ALL of configuration 0 in broadcasting mode
 - [A]Load TAS2557_BLOCK_PGM_DEV_A of configuration 0
 - [B]Load TAS2557_BLOCK_PGM_DEV_B of configuration 0
- 5. Download the PLL.
 - For TAS2557 mono applications: [A] Load TAS2557_BLOCK_PLL of configuration 0
 - For TAS2557 stereo applications: [AB] Load TAS2557_BLOCK_PLL of configuration 0
 - For TAS2559 applications: [A] Load TAS2557_BLOCK_PLL of configuration 0
- 6. Download the predata of the configuration.
 - [A]Load TAS2557_BLOCK_CFG_PRE_DEV_A of configuration 0
 - [B]Load TAS2557_BLOCK_CFG_PRE_DEV_B of configuration 0
- 7. Download the coefficient.
 - [A]Load TAS2557_BLOCK_CFG_COEFF_DEV_A of configuration 0
 - [B]Load TAS2557_BLOCK_CFG_COEFF_DEV_B of configuration 0
- 8. Download calibration data if present.
 - [A]Load TAS2557_BLOCK_CFG_CAL_A of configuration 0
 - [B]Load TAS2557_BLOCK_CFG_CAL_B of configuration 0
 - Stop here unless playback is required. If playback is required, proceed to Step 10.
- 9. Feed the PLL clock. Audio stream may start any time after this step.
- 10. [AB]Power on TAS devices.
 - (a) Load the start-up sequence; this is not part of the binary file.

For TAS2559 applications, the TAS2559 must be turned on first, then followed by a TAS2560 0.2-ms delay required in between.

TAS2557 mono applications:

See tas2557-android-driver.

static unsigned int p_tas2557_startup_data[] =

TAS2557 stereo applications:

See tas2557-stereo-driver.

static unsigned int p_tas2557_startup_data[]

TAS2559 applications:

See tas2559-android-driver.

static unsigned int p_tas2559_startup_data[]

- 11. Download the power power up block.
 - [B]Load TAS2559_BLOCK_PST_POWERUP_DEV_B of configuration 0 This block is only present for TAS2559 applications.
- 12. [AB]Unmute TAS devices.
 - (a) Load the unmute sequence; this is not part of the binary file.
 - TAS2557 mono applications:
 - See tas2557-android-driver.

static unsigned int p_tas2557_unmute_data[] =

TAS2557 stereo applications:
 See tas2557-stereo-driver.

static unsigned int p_tas2557_unmute_data[]

• TAS2559 applications:

See tas2559-android-driver.

static unsigned int p_tas2559_unmute_data[]



Device Driver Integration

www.ti.com

Steps to sleep:

- 1. [AB]Mute and shutdown TAS devices.
 - (a) Load the shutdown sequence; this is not part of the binary file.
 - TAS2557 mono applications:

See tas2557-android-driver.

static unsigned int p_tas2557_shutdown_data[] =

• TAS2557 stereo applications: See tas2557-stereo-driver.

static unsigned int p_tas2557_shutdown_data[]

TAS2559 applications:

See tas2559-android-driver.

```
static unsigned int p_tas2559_shutdown_data[]
```

2. Turn off the PLL clock.

4.2 Case Two

TAS2557 and TAS2559 has been programmed and there is no need to change the configuration.

Use case: device power up to play music.

The DSP program memory has been programmed and the coefficient memory has been programmed.

Steps to playback:

- 1. Feed the PLL clock. Audio stream may start any time after this step.
- 2. [AB]Power on TAS devices.
 - (a) Load the start-up sequence; this is not part of the binary file.

For TAS2559 applications, the TAS2559 must be turned on first, then followed by a TAS2560 0.2-ms delay required in between.

• TAS2557 mono applications:

See tas2557-android-driver.

static unsigned int p_tas2557_startup_data[] =

 TAS2557 stereo applications: See tas2557-stereo-driver.

static unsigned int p_tas2557_startup_data[]

TAS2559 applications:

See tas2559-android-driver.

static unsigned int p_tas2559_startup_data[]

- 3. Download the post power up block.
 - [B]Load TAS2559_BLOCK_PST_POWERUP_DEV_B of current configuration This block is only present for TAS2559 applications.

- 4. [AB]Unmute TAS devices.
 - (a) Load the unmute sequence; this is not part of the binary file.
 - TAS2557 mono applications:

See tas2557-android-driver.

static unsigned int p_tas2557_unmute_data[] =

TAS2557 stereo applications: See tas2557-stereo-driver.

static unsigned int p_tas2557_unmute_data[]

• TAS2559 applications: See tas2559-android-driver.

static unsigned int p_tas2559_unmute_data[]

Steps to sleep:

- 1. [AB]Mute and shut down TAS devices.
 - (a) Load the shutdown sequence; this is not part of the binary file.
 - TAS2557 mono applications:
 - See tas2557-android-driver.

static unsigned int p_tas2557_shutdown_data[] =

• TAS2557 stereo applications: See tas2557-stereo-driver.

static unsigned int p_tas2557_shutdown_data[]

TAS2559 applications:

See tas2559-android-driver.

static unsigned int p_tas2559_shutdown_data[]

2. Turn off the PLL clock.

4.3 Case Three

Need to change to a new configuration with the same program and same PLL.

Use case: switch to another tuning (PPC3 snapshot) during music playback. Assume the new configuration is x.

Steps to playback:

NOTE: If the music is playing and TAS devices are running, skips Steps 3, 4, 5, and 6. If the music is not playing, stop after completing Step 2.

- 1. Download the coefficient.
 - [A]Load TAS2557_BLOCK_CFG_COEFF_DEV_A of configuration x
 - [B]Load TAS2557_BLOCK_CFG_COEFF_DEV_B of configuration x
- 2. Download the calibration data if present.
 - [A]Load TAS2557_BLOCK_CFG_CAL_A of configuration x
 - [B]Load TAS2557_BLOCK_CFG_CAL_B of configuration x
- 3. Feed the PLL clock. The audio stream may start any time after this step.



- 4. [AB]Power on TAS devices.
 - (a) Load the start up sequence; this is not part of the binary file.

For TAS2559 applications, the TAS2559 must be turned on first, then followed by a TAS2560 0.2-ms delay required in between.

• TAS2557 mono applications: See tas2557-android-driver.

static unsigned int p_tas2557_startup_data[] =

TAS2557 stereo applications: See tas2557-stereo-driver.

static unsigned int p_tas2557_startup_data[]

• TAS2559 applications:

See tas2559-android-driver.

```
static unsigned int p_tas2559_startup_data[]
```

- 5. Download the post power up block.
 - [B]Load TAS2559_BLOCK_PST_POWERUP_DEV_B of current configuration This block is only present for TAS2559 applications.
- 6. Load the unmute sequence; this is not part of the binary file.
 - TAS2557 mono applications:
 - See tas2557-android-driver.

static unsigned int p_tas2557_unmute_data[] =

 TAS2557 stereo applications: See tas2557-stereo-driver.

static unsigned int p_tas2557_unmute_data[]

TAS2559 applications:

See tas2559-android-driver.

static unsigned int p_tas2559_unmute_data[]

Steps to sleep:

1. [AB]Mute and shutdown TAS devices.

(a) Load the shutdown sequence; this is not part of the binary file.

TAS2557 mono applications:

See tas2557-android-driver.

static unsigned int p_tas2557_shutdown_data[] =

• TAS2557 stereo applications:

See tas2557-stereo-driver.

static unsigned int p_tas2557_shutdown_data[]

TAS2559 applications:

See tas2559-android-driver.

static unsigned int p_tas2559_shutdown_data[]

2. Turn off the PLL clock.



4.4 Case Four

Need to change to a ne configuration with the same program and different PLL.

Use case: device is playing with a 48-kHz sample rate and switches to music playback with a 44.1-kHz sample rate.

Device Driver Integration

Assume the new configuration index is x.

NOTE: [AB]If the music is playing and TAS devices are running, load the shutdown sequence. If the music is not playing, skip steps 5, 6, 7, and 8.

The shutdown sequence is not part of the binary file.

TAS2557 mono applications:

See tas2557-android-driver.

static unsigned int p_tas2557_shutdown_data[] =

TAS2557 stereo applications:

See tas2557-stereo-driver.

static unsigned int p_tas2557_shutdown_data[]

TAS2559 applications:

See tas2559-android-driver.

static unsigned int p_tas2559_shutdown_data[]

- 1. Download the PLL.
 - For TAS2557 mono applications: [A]Load TAS2557_BLOCK_PLL of configuration x
 - For TAS2557 stereo applications: [AB]Load TAS2557_BLOCK_PLL of configuration x
 - For TAS2559 applications: [A]Load TAS2557_BLOCK_PLL of configuration x
- 2. Download the configuration predata.
 - [A]Load TAS2557_BLOCK_CFG_PRE_DEV_A of configuration x
 - [B]Load TAS2557_BLOCK_CFG_PRE_DEV_B of configuration x
- 3. Download the coefficient.
 - [A]Load TAS2557_BLOCK_CFG_COEFF_DEV_A of configuration x
 - [B]Load TAS2557_BLOCK_CFG_COEFF_DEV_B of configuration x
- 4. Download the calibration data if present.
 - [A]Load TAS2557_BLOCK_CFG_CAL_A of configuration x
 - [B]Load TAS2557_BLOCK_CFG_CAL_B of configuration x
- 5. Feed the PLL clock. The audio stream may start any time after this step.



- 6. [AB]Power on the TAS devices.
 - (a) Load the startup sequence; this is not part of the binary file.

For TAS2559 applications, the TAS2559 must be turned on first, then followed by a TAS2560 0.2-ms delay required in between.

 TAS2557 mono applications: See tas2557-android-driver.

static unsigned int p_tas2557_startup_data[] =

TAS2557 stereo applications: See tas2557-stereo-driver.

static unsigned int p_tas2557_startup_data[]

• TAS2559 applications:

See tas2559-android-driver.

```
static unsigned int p_tas2559_startup_data[]
```

- 7. Download the post power up block.
 - [B]Load TAS2559_BLOCK_PST_POWERUP_DEV_B of configuration x This block is only present for TAS2559 applications.
- 8. [AB]Unmute TAS devices.

Load the unmute sequence; this is not part of the binary file.

TAS2557 mono applications:

See tas2557-android-driver.

static unsigned int p_tas2557_unmute_data[] =

- TAS2557 stereo applications:
 - See tas2557-stereo-driver.

static unsigned int p_tas2557_unmute_data[]

• TAS2559 applications:

See tas2559-android-driver.

static unsigned int p_tas2559_unmute_data[]

Steps to sleep:

1. [AB]Mute and shutdown TAS devices.

Load the shutdown sequence; this is not part of the binary file.

 TAS2557 mono applications: See tas2557-android-driver.

static unsigned int p_tas2557_shutdown_data[] =

• TAS2557 stereo applications:

See tas2557-stereo-driver.

static unsigned int p_tas2557_shutdown_data[]

TAS2559 applications:

See tas2559-android-driver.

static unsigned int p_tas2559_shutdown_data[]

2. Turn off the PLL clock.



4.5 Case Five

Need to change to a new configuration with a different program.

Use case: from speaker protection mode (tuning mode), change to ROM1 mode for factory test. Assume the new configuration index is x.

Device Driver Integration

Steps to playback:

[AB]If music is playing and TAS devices are running, load the shutdown sequence. The shutdown sequence is not part of the binary file.

• TAS2557 mono applications:

See tas2557-android-driver.

static unsigned int p_tas2557_shutdown_data[] =

TAS2557 stereo applications:

See tas2557-stereo-driver.

static unsigned int p_tas2557_shutdown_data[]

TAS2559 applications:

See tas2559-android-driver.

static unsigned int p_tas2559_shutdown_data[]

1. [AB]Hardware reset.

Hardware reset is strongly recommended for reliable operations afterward.

- 2. [AB]Software reset.
- 3. [AB]Initialize.

This is not part of the binary file. This initialization is about ASI format, IRQ configuration, and some other configurations.

TAS2557 mono applications:

See tas2557-android-driver.

int tas2557_load_default(struct tas2557_priv *pTAS2557)

TAS2557 stereo applications:

See tas2557-stereo-driver.

int tas2557_load_default(struct tas2557_priv *pTAS2557)

• TAS2559 applications:

See tas2559-android-driver.

int tas2559_load_default(struct tas2559_priv *pTAS2559)

- 4. Download the program.
 - [AB]Load TAS2557_BLOCK_PGM_ALL of configuration x in broadcasting mode
 - [A]Load TAS2557_BLOCK_PGM_DEV_A of configuration x
 - [B]Load TAS2557_BLOCK_PGM_DEV_B of configuration x

Device Driver Integration



www.ti.com

- 5. Download the PLL.
 - For TAS2557 mono applications: [A]Load TAS2557_BLOCK_PLL of configuration x
 - For TAS2557 stereo applications: [AB]Load TAS2557_BLOCK_PLL of configuration x
 - For TAS2559 applications: [A]Load TAS2557_BLOCK_PLL of configuration x
- 6. Download the configuration predata.
 - [A]Load TAS2557_BLOCK_CFG_PRE_DEV_A of configuration x
 - [B]Load TAS2557_BLOCK_CFG_PRE_DEV_B of configuration x
- 7. Download the coefficient.
 - [A]Load TAS2557_BLOCK_CFG_COEFF_DEV_A of configuration x
 - [B]Load TAS2557_BLOCK_CFG_COEFF_DEV_B of configuration x
- 8. Download the calibration data if present.
 - [A]Load TAS2557_BLOCK_CFG_CAL_A of configuration x
 - [A]Load TAS2557_BLOCK_CFG_CAL_A of configuration x

Continue to Step 11 if music is playing. If music is not playing, stop at Step 10. Refer to Step 1.

9. Feed the PLL clock.

The PLL clock should run at this time.

10. [AB]Power on TAS devices.

Load the startup sequence; this is not part of the binary file.

For TAS2559 applications, the TAS2559 must be turned on first, then followed by a TAS2560 0.2-ms delay required in between.

• TAS2557 mono applications:

See tas2557-android-driver.

```
static unsigned int p_tas2557_startup_data[] =
```

• TAS2557 stereo applications:

See tas2557-stereo-driver.

static unsigned int p_tas2557_startup_data[]

- TAS2559 applications:
 - See tas2559-android-driver.

static unsigned int p_tas2559_startup_data[]

- 11. Download the post power up block.
 - [B]Load TAS2559_BLOCK_PST_POWERUP_DEV_B of configuration x This block is only present for TAS2559 applications.

12. [AB]Unmute TAS devices.

- Load the unmute sequence; this is not part of the binary file.
- TAS2557 mono applications:

See tas2557-android-driver.

static unsigned int p_tas2557_unmute_data[] =

 TAS2557 stereo applications: See tas2557-stereo-driver.

static unsigned int p_tas2557_unmute_data[]

 TAS2559 applications: See tas2559-android-driver.

static unsigned int p_tas2559_unmute_data[]

Steps to sleep:

1. [AB]Mute and shutdown TAS devices.

Load the shutdown sequence; this is not part of the binary file.

TAS2557 mono applications:

See tas2557-android-driver.

static unsigned int p_tas2557_shutdown_data[] =

 TAS2557 stereo applications: See tas2557-stereo-driver.

static unsigned int p_tas2557_shutdown_data[]

• TAS2559 applications:

See tas2559-android-driver.

static unsigned int p_tas2559_shutdown_data[]

2. Turn off the PLL clock.

5 Tuning and Debugging in System

With the In System Tuning feature available in PPC3, the user can connect the end system to the PPC3 through an Android Debug Bridge (ADB). Once connected, the user can start tuning audio directly on the end system.



Tuning and Debugging in System

5.1 Download Tuning Data to End System

Use the PPC3 file that contains the Dump Definition Configuration (DDC) used for generating the binary file in Section 3.1. Select Tuning and debugging in System and select the required snapshot in the DDC to tune on the end system. See Figure 14 and Figure 15.

Select Your Option	
What would you like to do?	
O Dump the Binary file	
Choose this option to generate a binary file for the end system driver.	
Tuning and debugging in System	
Choose this option if you would like to make fine adjustments in the end system. You will have to load the .ppc3 file that was created when you generated the binary fi	iles.
ASI Record Channel selection (Choose between Echo Reference, Temperature/Excursion and I// Data on the channels)	
This setting will be applied once you are connected to System Tuning.	
O Temperature on Left Channel and Excursion on Right Channel for device A	
Echo Reference on Left Channel and Excursion on Right Channel for device A	
O Isense on Left Channel and Vsense on Right Channel for device A	
Ne	xt

Figure 14. Tuning and Debugging in System

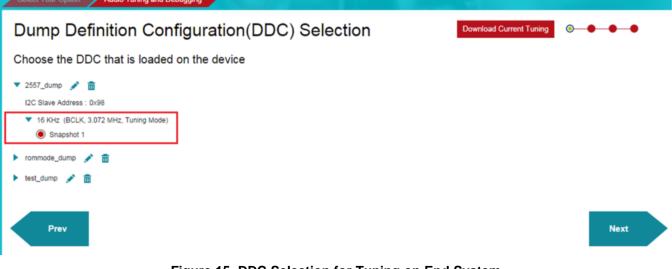


Figure 15. DDC Selection for Tuning on End System



Alternatively, users can download the current GUI settings to the end system instead of the DDC by clicking on the Download Current Tuning (see Figure 16).

Select Your Option Audio Tuning and Debugging	
Dump Definition Configuration(DDC) Selection	
Choose the DDC that is loaded on the device	
 test-dump I2C Slave Address : 0x98 48 KHz (BCLK, 3.072 MHz, Tuning Mode) Music Tuning 16 KHz (BCLK, 3.072 MHz, Tuning Mode) 	
Prev	Next
Figure 16. DDC Selection for Tuning on End System	

After selecting one of the options in Figure 16, users can click the Tuning and Audio Processing to tune the end system.



Revision History

www.ti.com

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Date	Version	Description
September 2017	*	Initial release

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ('TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your noncompliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/stdterms.htm), evaluation

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265 Copyright © 2017, Texas Instruments Incorporated