SIGMASTUDIO FOR SHARC (ADSP-SC5XX/ADSP-215XX) - QUICK START GUIDE

Document Status	Approved
Approved by	ASH

ANALOG DEVICES, INC.

www.analog.com

Revision List

Table 1: Revision List

Revision	Date	Description
0.1	27.04.2015	Updated the document for v3.1.0 of the product.
0.2	05.05.2015	Updated for minor change in the template.
1.0	06.05.2015	Approved and base-lined for release 3.1.0.
1.1	21.05.2015	Updated with adding more details on SigmaStudio Host settings
1.2	07.06.2015	Incorporated review comments.
1.3	12.06.2015	Incorporated review comments
1.4	10.07.2015	Added annexure F
1.5	13.07.2015	Incorporated review comments from ASH
2.0	13.07.2015	Approved and Base-lined for 3.2.0
2.1	21.08.2015	Updated with Debug feature details
2.2	09.09.2015	Updated with Multi-rate processing details. Added appendix G
2.3	05.10.2015	Updated for review comments and added code/ buffer sharing details for release 3.4.0. Updated figures of EZ-Kit and SigmaStudio IC form
3.0	07.10.2015	Approved and Base-lined for 3.4.0
3.1	01.02.2016	Updated for M6 features (multi instancing, load balancing and ARM microcontroller mode)
3.2	02.02.2016	Updated for review comments.
4.0	02.02.2016	Approved and Baselined for 3.5.0
4.1	20.04.2016	Updated for Release 3.6.0
5.0	21.04.2016	Approved and Base-lined for 3.6.0
5.1	23.06.2016	Updated for Release 3.7.0 • Updated sections “Hardware Setup”, “Demo application”, “DemoUc application” to include instructions for ADSP-SC573 processor.
5.2	27.06.2016	Incorporated review comments.
6.0	29.06.2016	Approved and Base-lined for 3.7.0

6.1	04.10.2016	Updated for release 3.8.0
7.0	07.10.2016	Approved and Base-lined for 3.8.0
7.1	19.12.2016	Updated for release 3.9.0
8.0	23.12.2016	Approved and Base-lined for Release 3.9.0
8.1	25.05.2017	Updated for release 3.10.0 Beta
8.2	23.06.2017	Minor corrections
9.0	28.06.2017	Approved and Baselined for 3.10.0 Beta
9.1	22.09.2017	Updated for release 3.11.0
10.0	29.09.2017	Approved and Baselined for 3.11.0
10.1	16.03.2018	Updated for release 4.0.0
10.2	19.03.2018	Minor updates
11.0	21.03.2018	Approved and Baselined for 4.0.0
11.1	23.05.2018	Updated for 4.1.0 release
11.2	31.05.2018	Updated for review comments
12.0	04.06.2018	Approved and Baselined for 4.1.0
12.1	10.08.2018	Updated for 4.2.0 release
13.0	04.09.2018	Approved and Baselined for 4.2.0
13.1	19.06.2019	Updated for 4.4.0 release
13.2	28.06.2019	Addressed review comments for 4.4.0 release
14.0	02.07.2019	Approved and Baselined for 4.4.0
14.1	5.11.2019	Updated for 4.5.0 release
14.2	08.11.2019	Addressed review comments for 4.5.0 release
14.3	08.11.2019	Addressed further comments for release 4.5.0
15.0	08.11.2019	Approved and Baselined for 4.5.0
15.1	21.12.2020	Updated for 4.6.0 release
15.2	22.12.2020	Addressed review comments for release 4.6.0
15.3	22.12.2020	Addressed further comments for release 4.6.0
16.0	23.12.2020	Approved and baselined for release 4.6.0
16.1	08.04.2022	Updated for release 4.7.0
16.2	13.04.2022	Addressed review comments.
17.0	13.04.2022	Approved and baselined for release 4.7.0

Copyright, Disclaimer Statements

Copyright Information

Disclaimer

Analog Devices, Inc. reserves the right to change this product without prior notice. Information furnished by Analog Devices is believed to be accurate and reliable. However, no responsibility is assumed by Analog Devices for its use; nor for any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under the patent rights of Analog Devices, Inc.

Table of Contents

List of Figures

List of Tables

1 Introduction

SigmaStudio™ is a development environment from Analog Devices for graphically programming ADI's DSPs. SigmaStudio for SHARC includes an extensive set of algorithms to perform audio processing tasks such as filtering and mixing, as well as basic low-level DSP functions, optimized to run on the SHARC family of processors. The environment is integrated with CrossCore® Embedded Studio.

1.1 Scope

This document explains how to use SigmaStudio for SHARC (ADSP-SC5xx[i]/ADSP-215xx). This document is a Quick Start Guide to getting a SigmaStudio for SHARC demonstration working on ADSP-SC5xx and ADSP-215xx processors. Developers are assumed to be familiar with SigmaStudio. A basic understanding of the SHARC EZ-BOARD and SOM EZ-BOARD is recommended.

1.2 Organization of the Guide

This guide is organized into the following sections:

Section 1: this section contains the introduction.

Section 2: this section contains the hardware and software requirements for using SigmaStudio for SHARC (ADSP-SC5xx/ADSP-215xx).

Section 3: this section contains instructions to set up hardware.

Section 4: this section contains safety precautions to be taken while handling the hardware.

Section 5: this section contains the instructions to install required software and tools.

Section 6: this section contains steps to flash the default loader files, run the demo application.

Section 7: this section introduces the example microcontroller application.

Section 8: this section introduces the example library integration application.

Section 9: this section contains steps to debug a schematic.

Section 10: this section contains steps to enable S/PDIF Tx feature.

Section 11: this section introduces the FIR/IIR hardware accelerator multi-instancing support on ADSP-SC59x/ADSP-2159x.

2 Requirements Setup

The software and hardware components required for setting up this product are detailed in the sections 2.1.1 and 2.1.2. These tool chain components must be independently requested from ADI and installed before attempting the demo. The versions of the components are listed in reference Table 3 of **[1]**.

2.1.1 Software Requirements

The following are the list of software components required for setting up the demo application.

1. CrossCore Embedded Studio
2. ADSP-SC5xx EZ-Kit Lite board support package
3. SigmaStudio
4. SigmaStudio for SHARC (ADSP-SC5xx/ADSP-215xx)

Refer to section 2.2 of **[1]** for more details on software requirements.

2.1.2 Hardware Requirements

The following are the list of hardware components required for setting up the demo application.

1. ADSP-21569/ADSP-SC573/ADSP-SC584/ADSP-SC589 EZ-Board, ADSP-SC589 MINI BOARD, ADSP-21569 SOM EZ-Board (EV-21569-SOM), ADSP-21593 SOM EZ-Board (EV-21593-SOM) and ADSP-SC594 SOM EZ-Board (EV-SC594-SOM). ADSP-21573/ADSP-21584 processor mounted on ADSP-SC573/ADSP-SC584 EZ-BOARD.
2. PC/Laptop with a stereo audio out and USB port.
3. ADZS-ICE-2000/ICE-1000 emulator for downloading/debugging/flashing the framework to the Target.
4. EVAL-ADUSB2EBZ USB to SPI converter.
5. Aardvark I^{2}C/SPI Host Adapter
6. Audio cables and USB cables.

Refer to section 2.2 of **[1]** for more details on Hardware requirements.

3 Hardware Setup

The SigmaStudio for SHARC (ADSP-SC5xx/ADSP-215xx) demonstration setup includes a Host PC running SigmaStudio which is connected to the ADSP-SC573/ADSP-SC584/ADSP-SC589/ADSP-21584/ADSP-21573/ADSP-21569/ADSP-21593/ADSP-SC594 EZ-Board[1]. The connection is achieved using a USB-to-SPI converter. The EVAL-ADUSB2EBZ or Aardvark I²C/SPI acts as the USB-to-SPI converter, which is connected to the PC through a USB port and to the Target EZ-Board through SPI lines. The basic steps to set up the Target for SigmaStudio for SHARC (ADSP-SC5xx/ADSP-215xx) demonstration are given below. The setup required to run the demo application is illustrated in Figure 1 below.

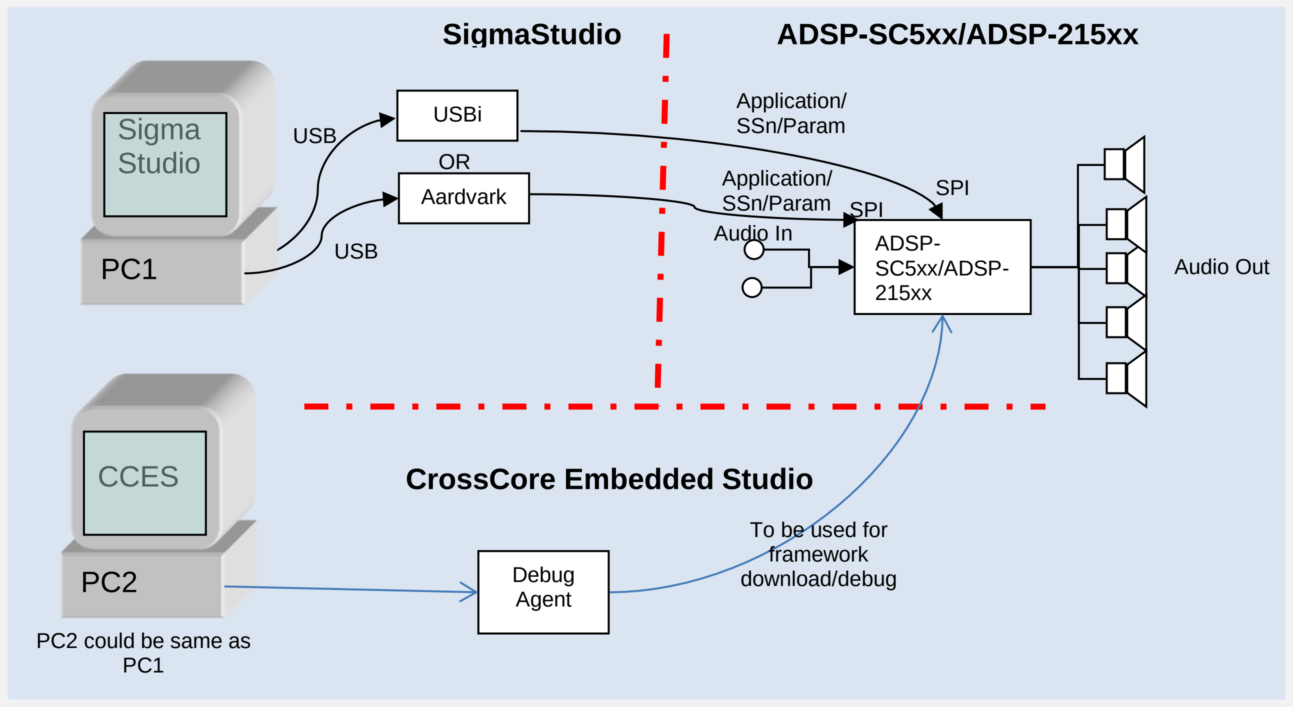


Figure 1: Demo Setup

3.1 Hardware Switches/Connections

[1] The ADSP-21573/ADSP-21584 processor is tested by mounting it on ADSP-SC573/ADSP-SC584 EZ-BOARD

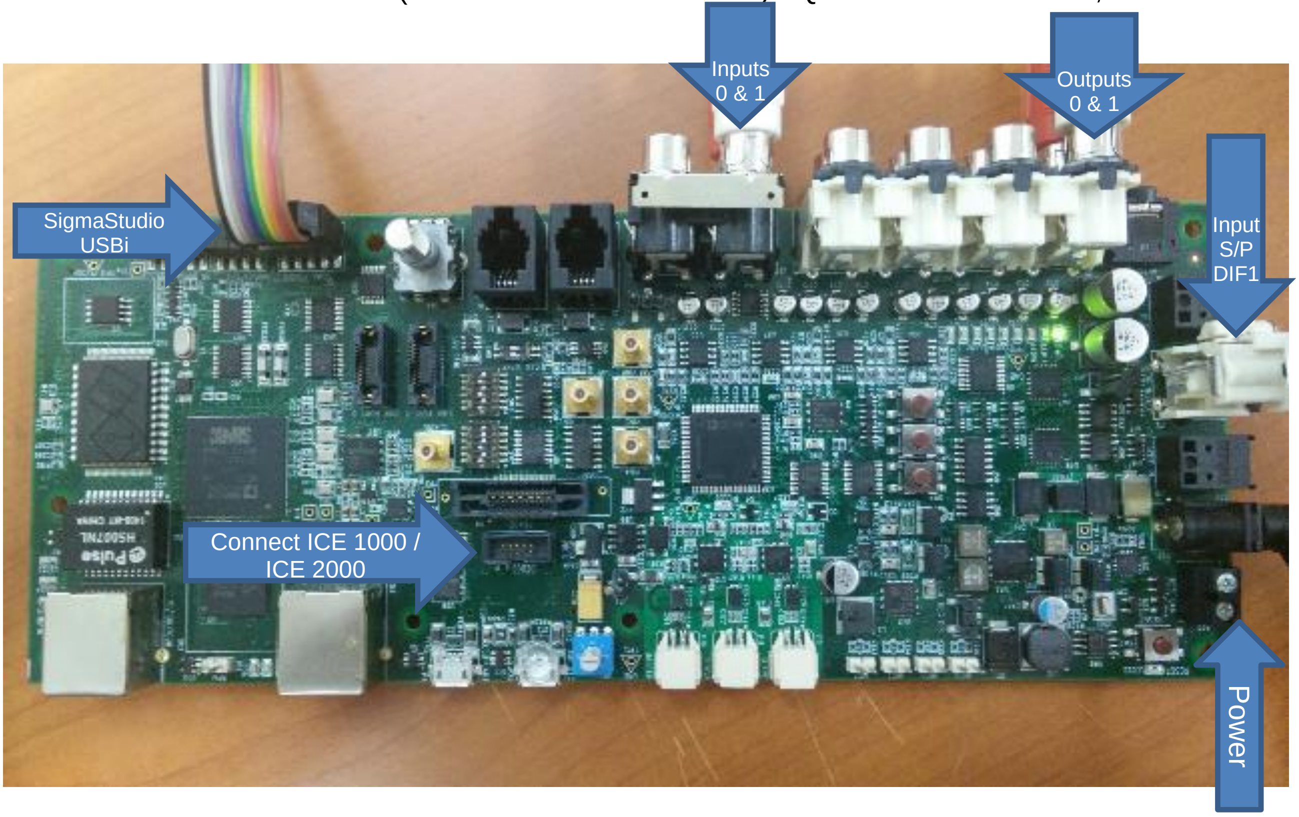


Figure 2: ADSP-SC584 Board Connections with USBi

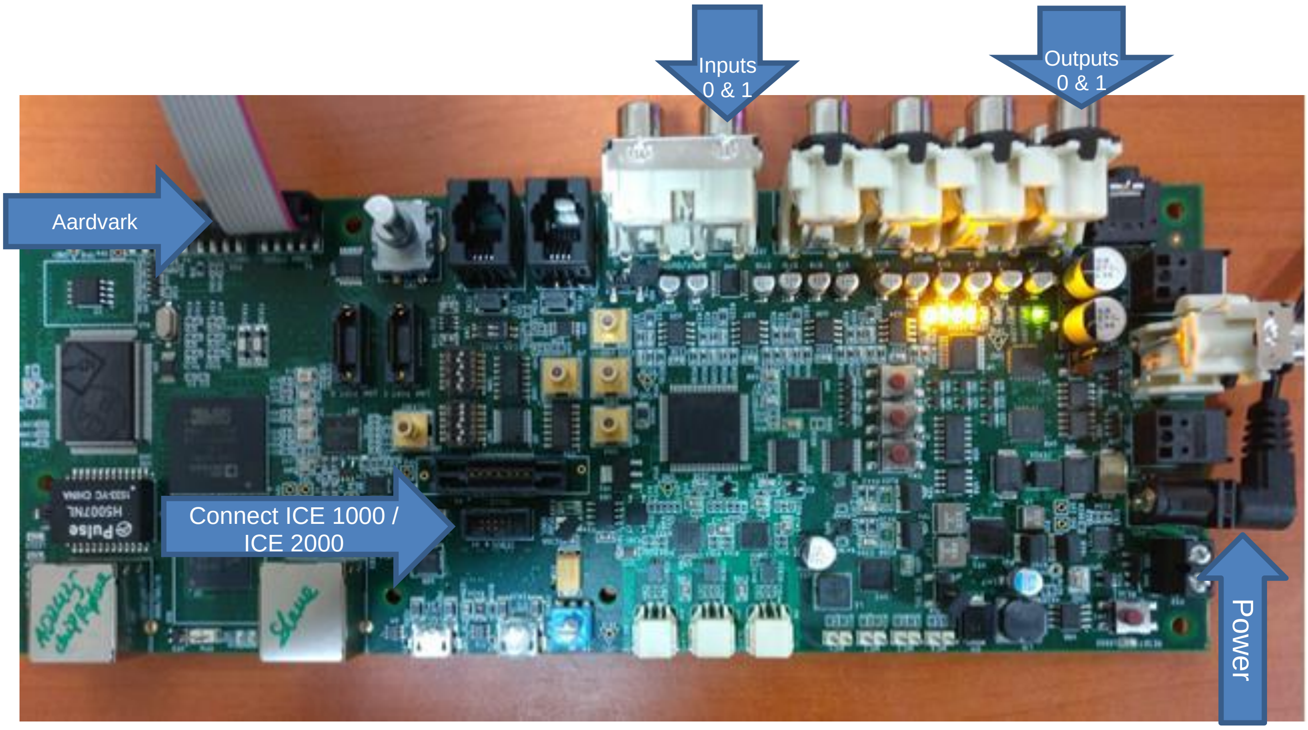


Figure 3: ADSP-SC584 Board Connections with Aardvark

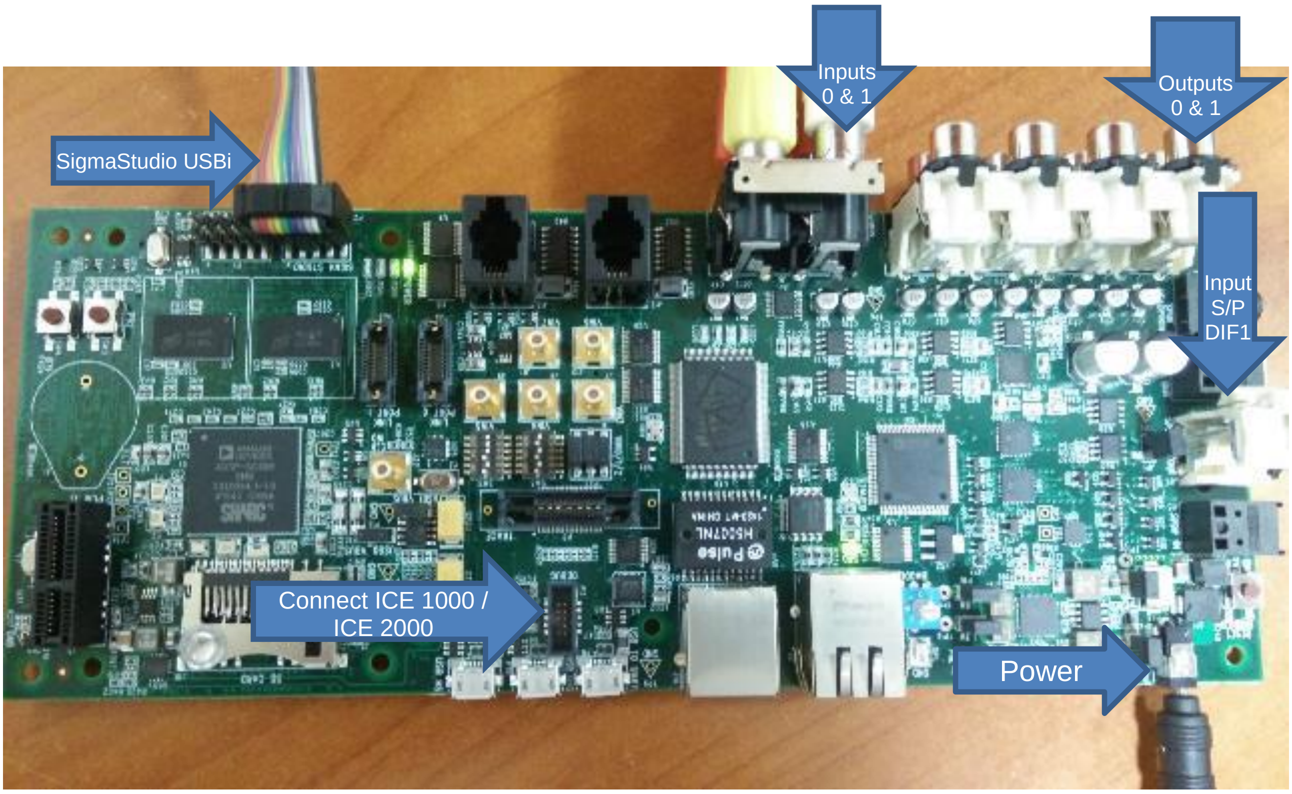


Figure 4: ADSP-SC589 Board Connections with USBi

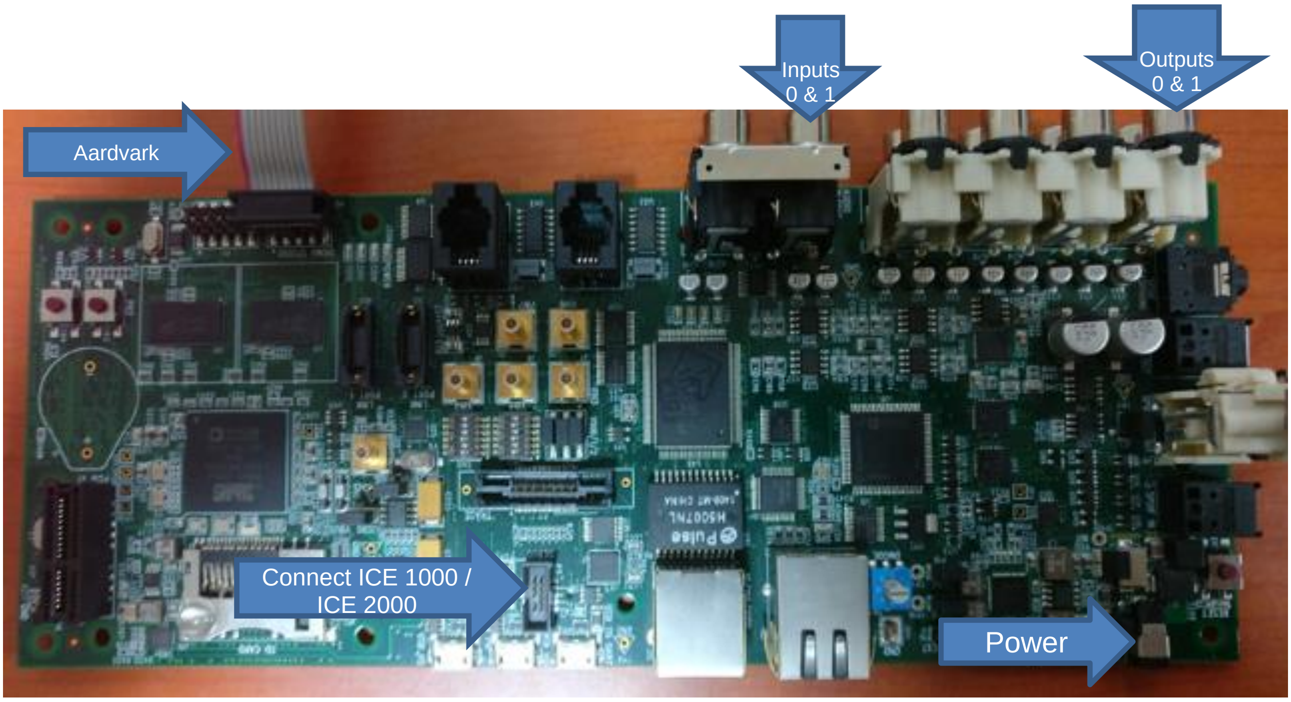


Figure 5: ADSP-SC589 Board Connections with Aardvark

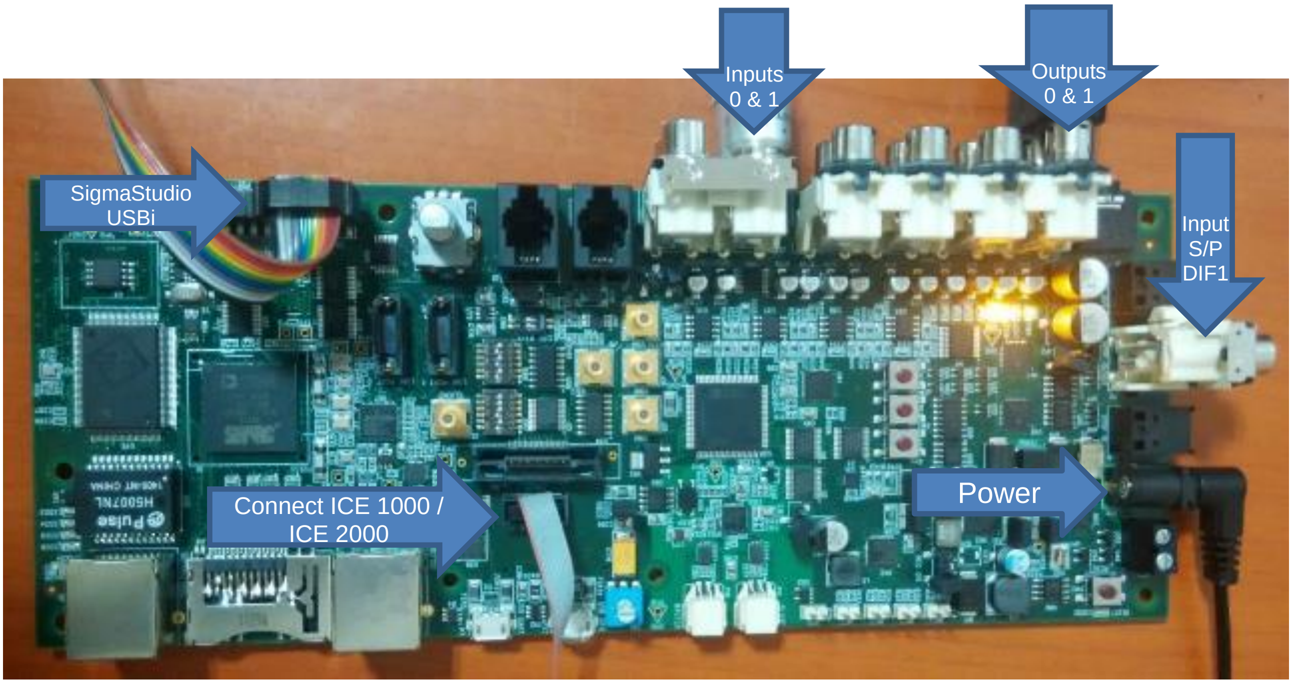


Figure 6: ADSP-SC573 Board Connections

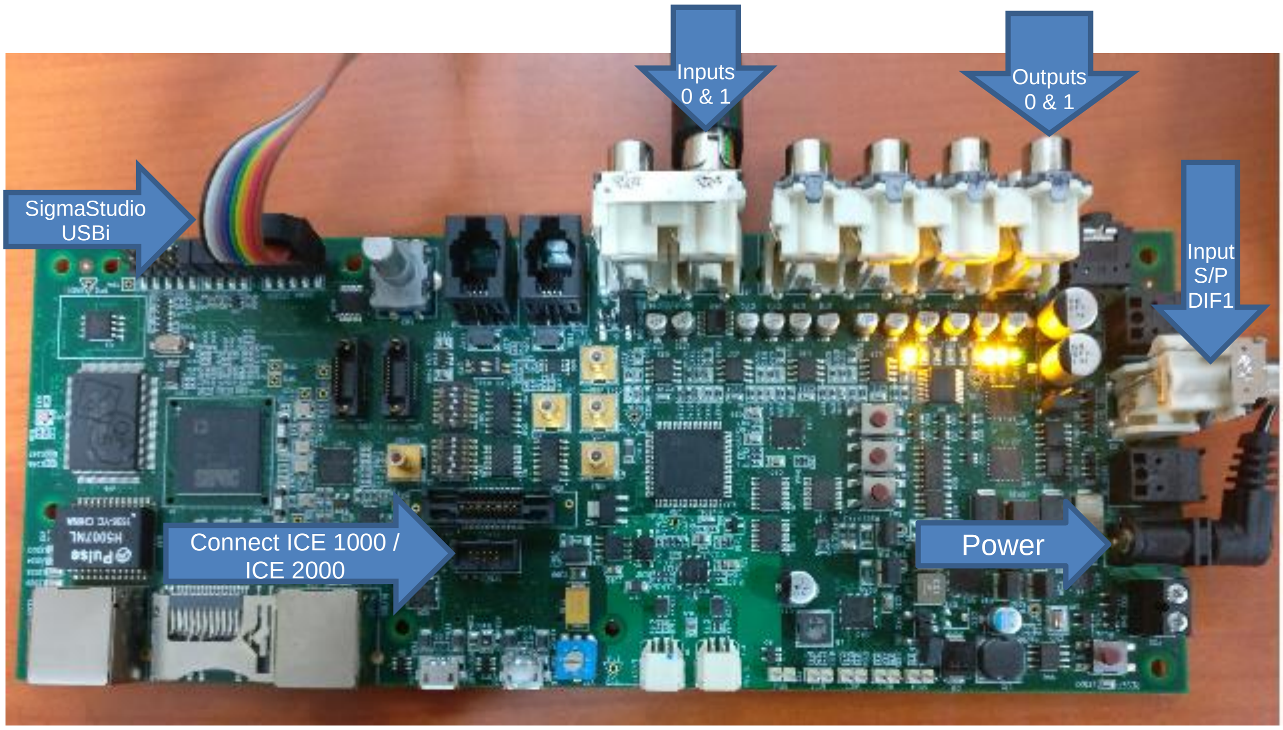


Figure 7: ADSP-21573 Board Connections

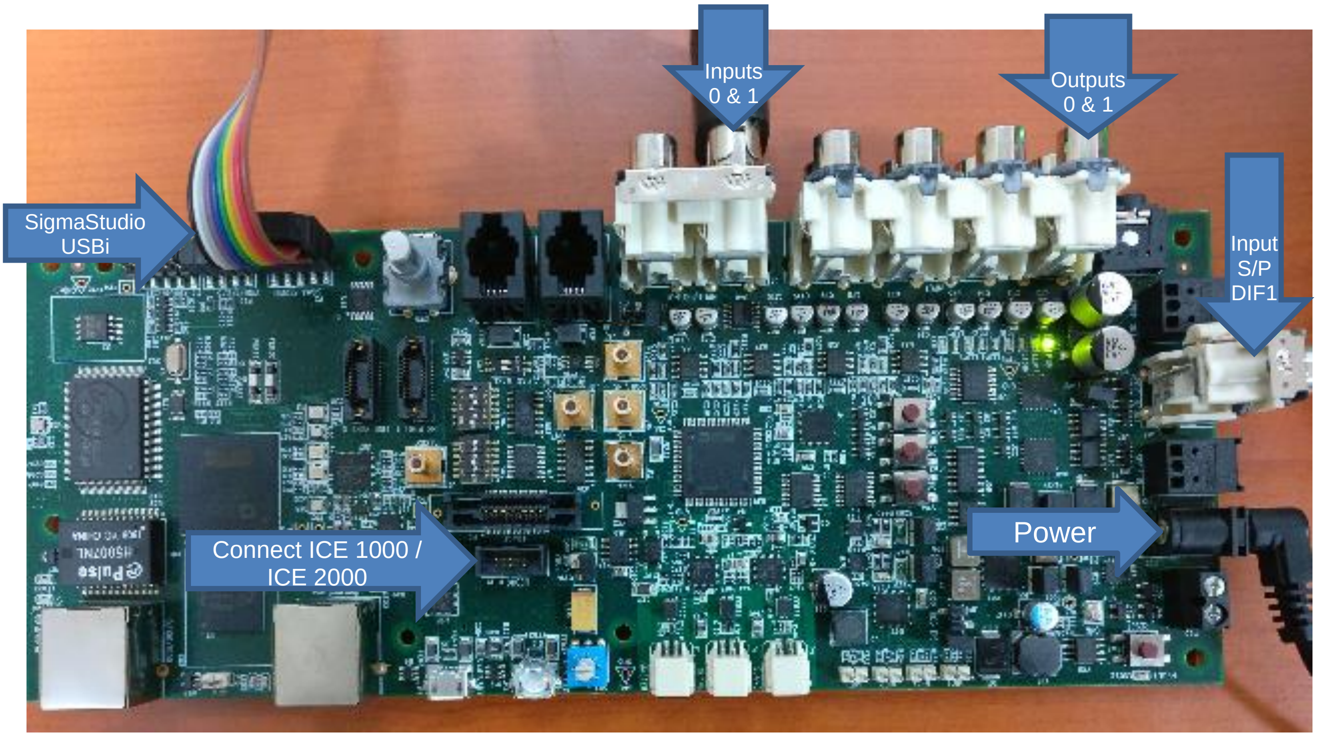


Figure 8: ADSP-21584 Board Connections

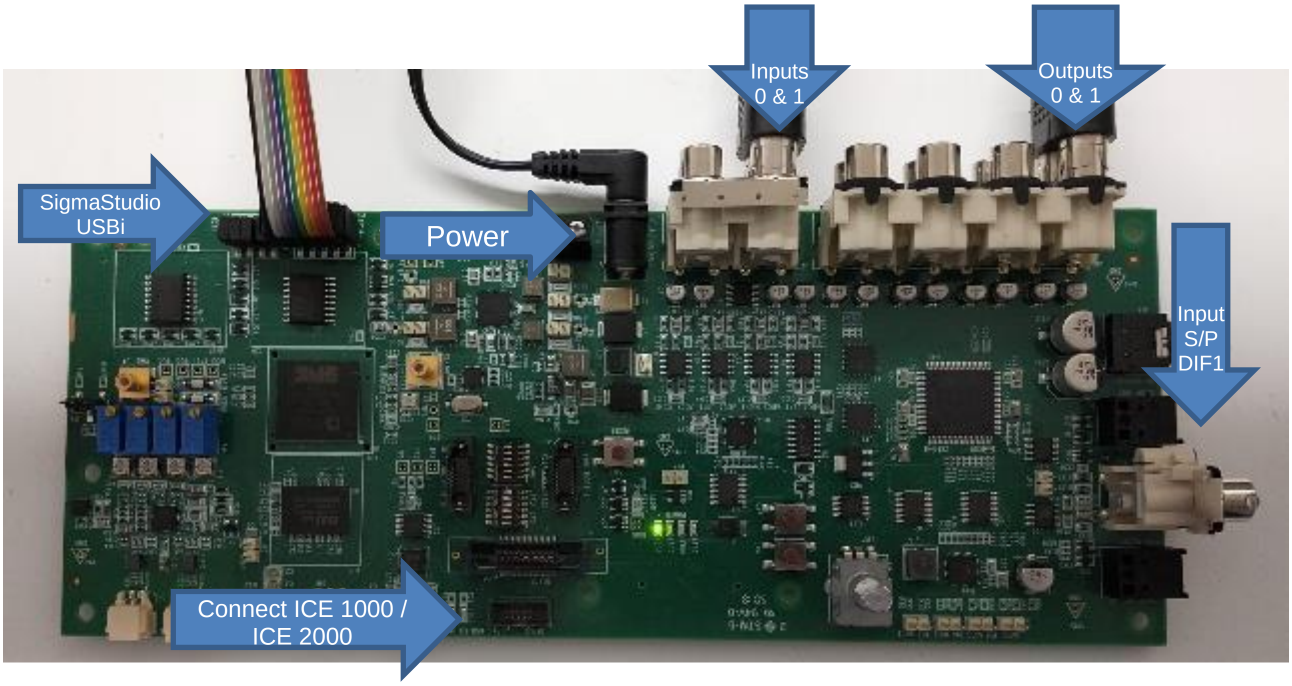


Figure 9: ADSP-21569 Board Connections

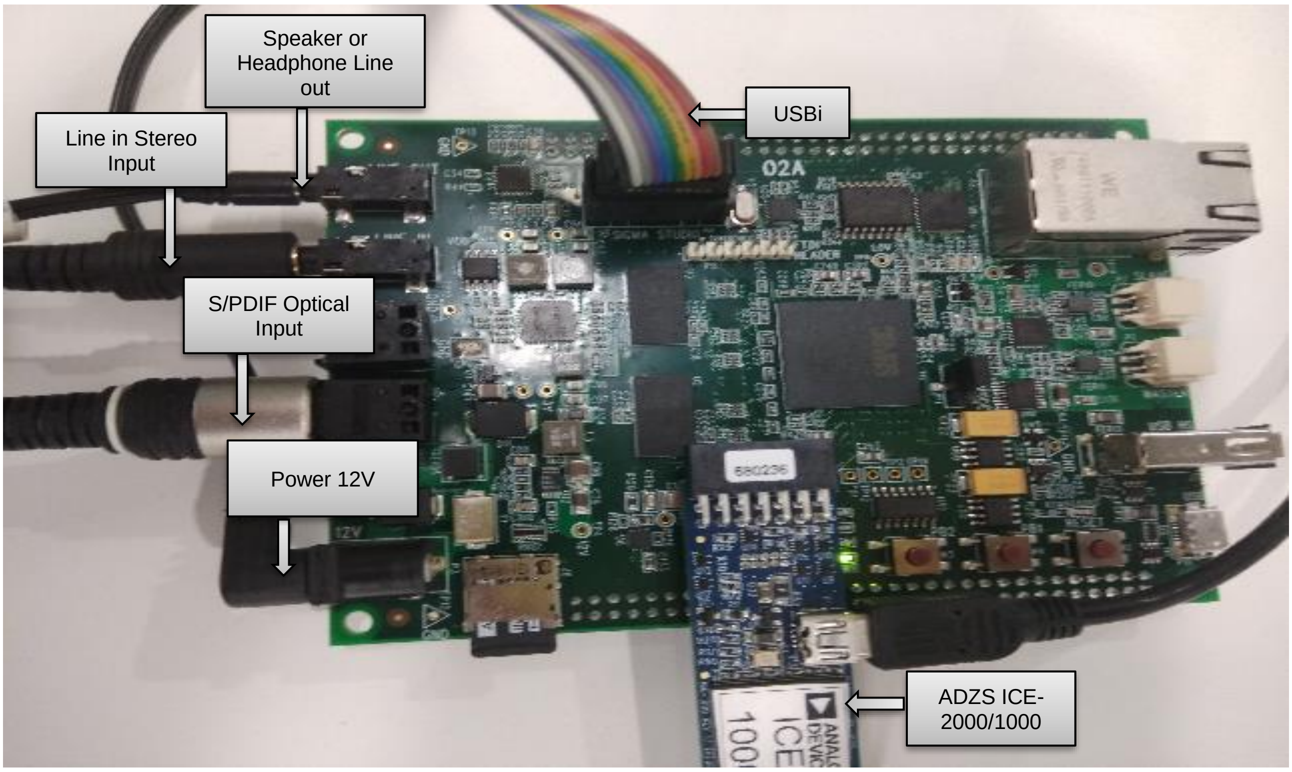


Figure 10: ADSP-SC589 MINI Board connections

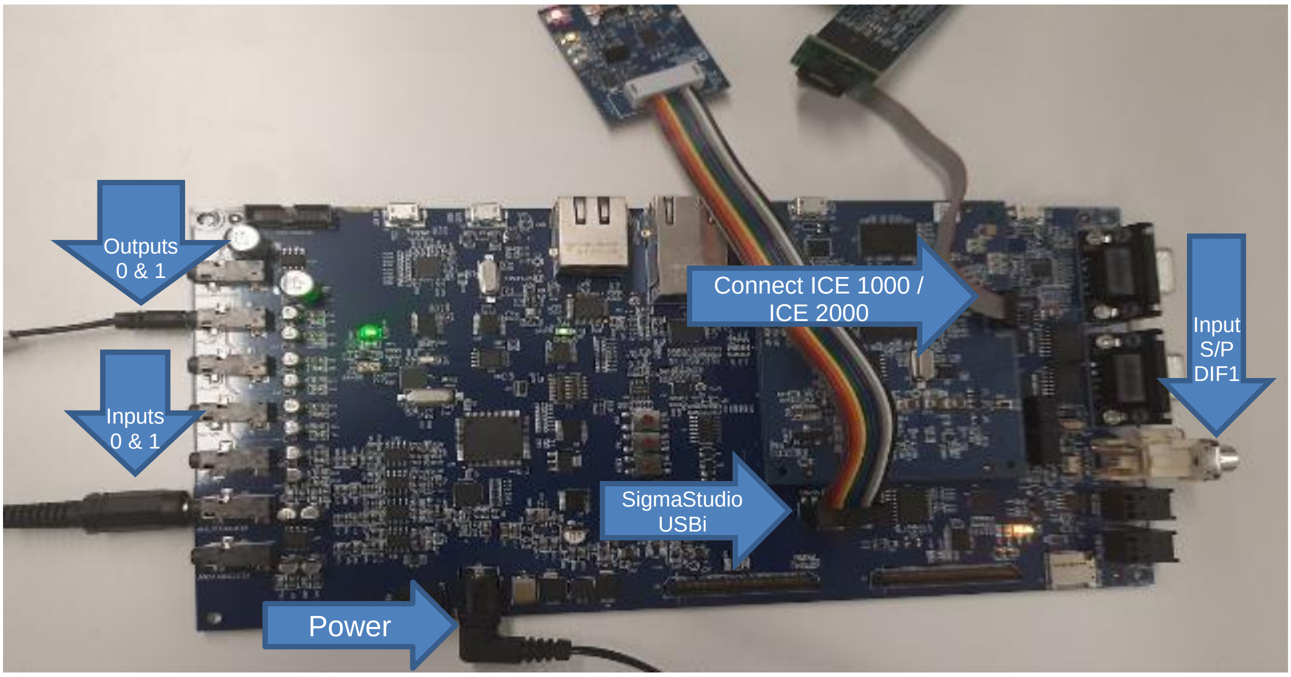


Figure 11: EV-21569-SOM with EV-SOM-CRR EZ-Kit Connections

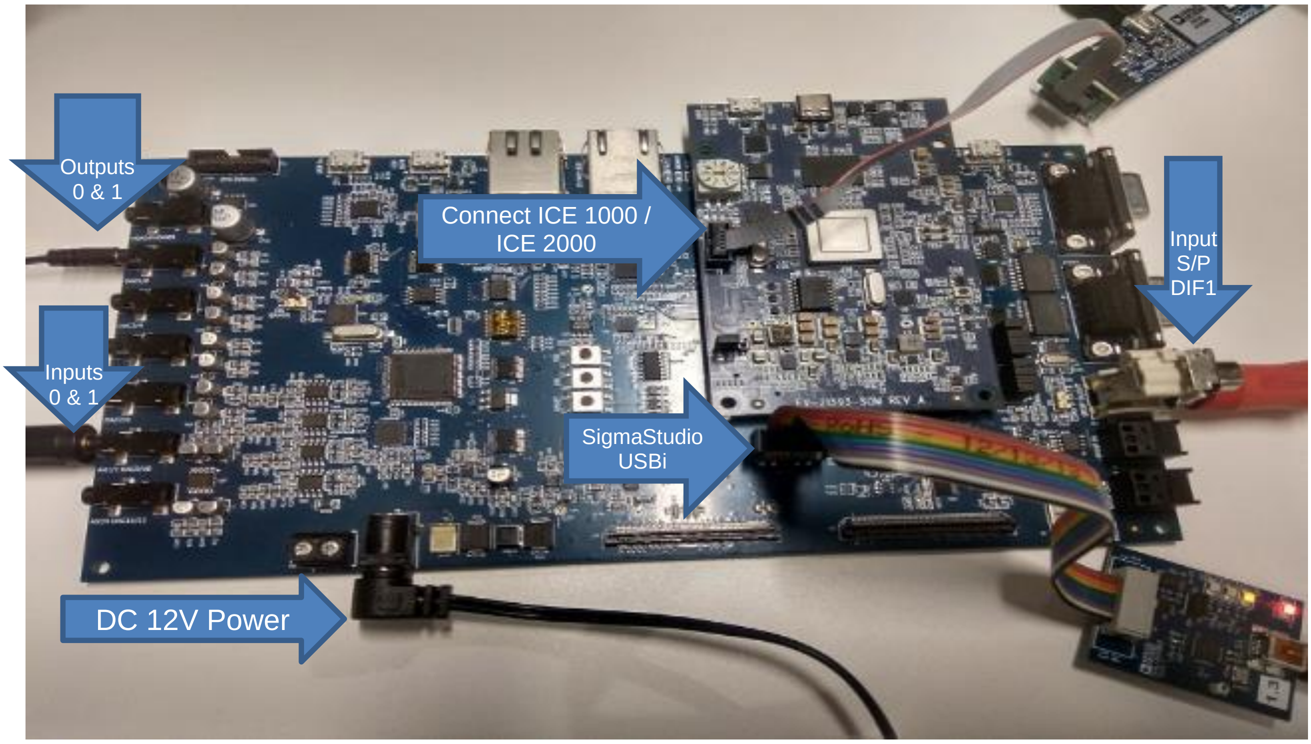


Figure 12: EV-21593-SOM with EV-SOM-CRR EZ-Kit Connections

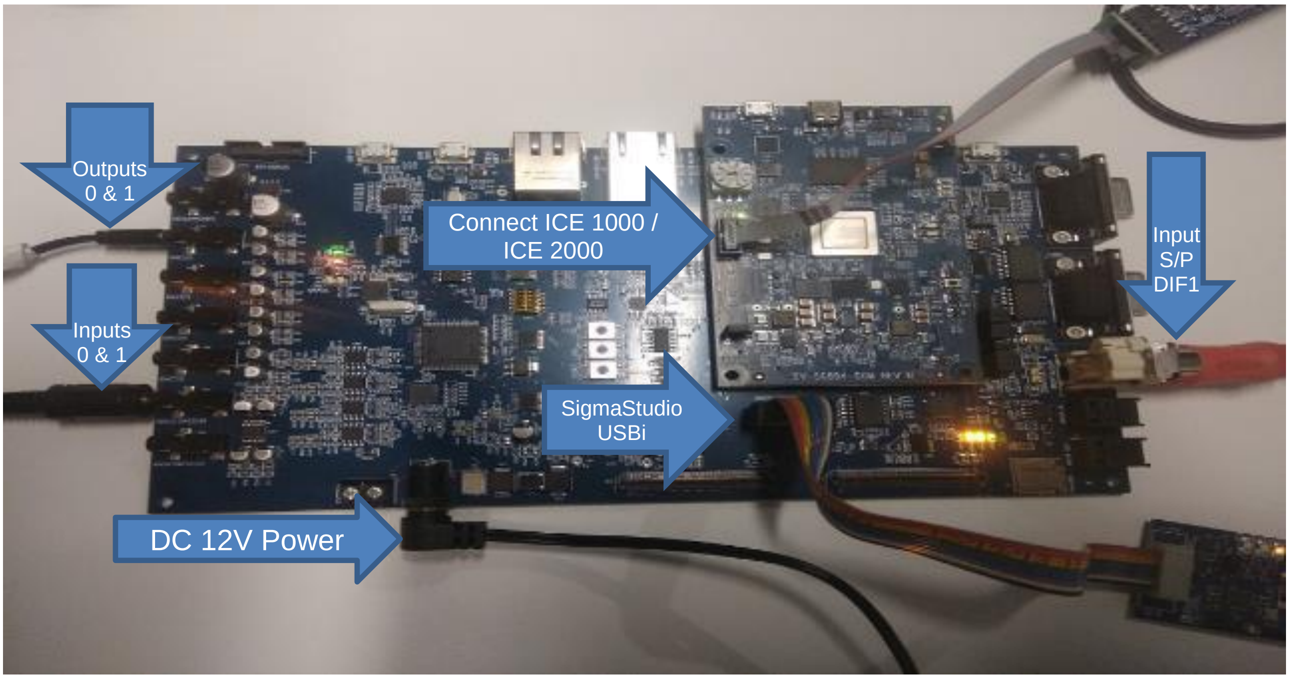


Figure 13: EV-SC594-SOM with EV-SOM-CRR EZ-Kit Connections

3.1.1 Connect EZ-BOARD to PC using USBi

Connect EVAL-ADUSB2EBZ to the PC running SigmaStudio using a USB cable. The EVAL-ADUSB2EBZ (hereafter referred to as the 'USBi') has a 10-pin socket. This socket should be connected to the header P2 (marked SIGMASTUDIO) on EZ-BOARD. When connected, Pin 1 of the USBi socket should match pin 1 of header P2 on the EZ-BOARD. This connection is used to download SigmaStudio packets from host PC to the EZ-BOARD.

The USBi to EZ-board connection (direction) is different across different EZ-Boards. Please refer to Figure 2, Figure 4, Figure 6, Figure 7, Figure 8 and Figure 9 above on how to connect to the specific EZ-Board.

3.1.2 Connect EZ-Board to PC using Aardvark I^{2}C/SPI Host Adapter

Connect Aardvark I2C/SPI Host Adapter to the PC running SigmaStudio using a USB cable. Aardvark I2C/SPI Host Adapter has a 10-pin socket. This socket should be connected to the header P2 (marked SIGMASTUDIO) on EZ-BOARD. When connected, Pin 1 of the USBi socket should match pin 1 of header P2 on the EZ-BOARD. This connection is used to download SigmaStudio packets from host PC to the EZ-BOARD.

The Aardvark to EZ-board connection (direction) is different across different EZ-Boards. Please refer to Figure 3, Figure 5 above on how to connect to the specific EZ-Board

3.1.3 Connect EV-SOMCRR-EZKIT to PC using USBi

Connect EVAL-ADUSB2EBZ to the PC running SigmaStudio using a USB cable. The EVAL-ADUSB2EBZ (hereafter referred to as the 'USBi') has a 10-pin socket. This socket should be connected to the header P3 (marked SIGMA STUDIO) on EV-SOMCRR-EZKIT. When connected, Pin 1 of the USBi socket should match pin 1 of header P3 on the EV-SOMCRR-EZKIT. This connection is used to download SigmaStudio packets from host PC to the EV-SOMCRR-EZKIT.

For the USBi to EV-SOMCRR-EZKIT connection, please refer to Figure 11, Figure 12Figure 13 and Figure 13 above on how to connect to the EV-SOMCRR-EZKIT.

The Aardvark to EV-SOMCRR-EZKIT connection is similar to USBi connections.

3.1.4 Connect to PC using ICE-2000/ICE-1000

Connect ADZS-ICE-2000/ICE-1000 to the PC running SigmaStudio using a USB cable. In case of ADZS-ICE-2000, the ribbon cable of ICE-2000 should be connected to header P1 (marked DEBUG) on EZ-BOARD. In case of ICE-1000, connect header J2 of ICE-1000 directly to header P1 (marked DEBUG) on EZ-BOARD. This connection is used to download/debug/flash the framework to the target.

Connect ADZS-ICE-2000/ICE-1000 to the PC running SigmaStudio using a USB cable. The ribbon cable of ICE-2000/ICE-1000 should be connected to header P1 (marked DEBUG) on EV-21569-SOM or EV-SC594-SOM or EV-21593-SOM boards.

3.1.5 Connect Audio Input and Output

Connect an analog audio source and headphones/speakers to the audio ports on the EZ-KIT or EV-SOMCRR-EZKIT.

3.1.6 Connect Power to the EZ-KIT

Connect the power and reset the board by pressing the ‘RESET’ button on the EZ-BOARD or EV-SOMCRR-EZKIT.

3.1.7 Switch/Jumper/Port Settings

3.1.8.1 ADSP-SC584 EZ-BOARD

The board settings for running SigmaStudio on ADSP-SC584 EZ-BOARD, using the Default Application is given in the table below.

Switch	Setting
SW1	Position 0
SW6	ON, OFF, ON, OFF, ON, OFF
SW7	ON, OFF, ON, OFF, OFF, OFF
SW8	OFF, OFF

Table 2: Switch Settings for ADSP-SC584 EZ-BOARD

Jumper	Setting
JP1	Not connected
JP2	Not connected
JP3	Not connected
JP4	Not connected

Table 3: Jumper Settings for ADSP-SC584 EZ-BOARD

Port	Setting
P2	Connected with SigmaStudio USBi
P3	Not connected
P12	Not connected
P17	Connected
P18	Connected
P20	Connected
P21	Connected
P24	Not connected
P25	Pins (3,4) (5,6) (7,8) Connected

Table 4: Port Settings for ADSP-SC584 EZ-BOARD

3.1.8.2 ADSP-SC589 EZ-BOARD

The board settings for running SigmaStudio on ADSP-SC589 EZ-BOARD, using the Default Application is given in the table below.

Switch	Setting
SW1	Position 0
SW5	ON, OFF, ON, OFF, ON, OFF
SW6	ON, OFF, ON, OFF, OFF, OFF

Table 5: Switch Settings for ADSP-SC589 EZ-BOARD

Jumper	Setting
JP1	Connected

Table 6: Jumper Settings for ADSP-SC589 EZ-BOARD

Port	Setting
P2	Connected with SigmaStudio USBi
P14	Connected
P15	Connected
P16	Connected
P17	Connected
P18	Connected
P19	Connected
P21	Connected
P23	All 3 pairs of jumpers are connected

Table 7: Port Settings for ADSP-SC589 EZ-BOARD

3.1.8.3 ADSP-SC573 EZ-BOARD

The board settings for running SigmaStudio on ADSP-SC573 EZ-BOARD, using the Default Application is given in the table below.

Switch	Setting
SW1	Position 0
SW6	ON, OFF, ON, OFF, ON, OFF
SW7	ON, OFF, ON, OFF, OFF, OFF
SW8	OFF, OFF

Table 8: Switch Settings for ADSP-SC573 EZ-BOARD

Jumper	Setting
JP1	Not connected
JP2	Not connected
JP3	Not connected
JP4	Not connected

Table 9: Jumper Settings for ADSP-SC573 EZ-BOARD

Port	Setting
P2	Connected with SigmaStudio USBi
P17	Connected
P18	Connected
P20	Connected
P21	Connected
P24	Connected
P25	All pairs of jumpers are connected

Table 10: Port Settings for ADSP-SC573 EZ-BOARD

3.1.8.4 ADSP-SC589 MINI BOARD

The board settings for running SigmaStudio on ADSP-SC589 MINI BOARD, using the Default Application is given in the table below.

Switch	Setting
SW2	Reset
SW3	Not Used
SW4	Not Used

Table 11: Switch Settings for ADSP-SC589 MINI Board

Jumper	Setting
JP1	Boot Mode (1 and 2 connected for SPI boot) else disconnected

Table 12: Jumper Settings for ADSP-SC589 MINI Board

Port	Setting
P1	Debug port connected to ADZS-ICE-2000/ICE-1000
P2	Connected with SigmaStudio USBi
P3	Connected to Power

Table 13: Port Settings for ADSP-SC589 MINI Board

3.1.8.5 ADSP-21569 EZ-BOARD

The board settings for running SigmaStudio on ADSP-21569 EZ-BOARD, using the Default Application is given in the table below.

Switch	Setting
SW5	ON, OFF, ON, OFF, ON, OFF
SW6	ON, OFF, ON, OFF, OFF, OFF

Table 14: Switch Settings for ADSP-21569 EZ-BOARD

Jumper	Setting
JP1	Not connected
JP7	Not connected
JP8	Not connected

Table 15: Jumper Settings for ADSP-21569 EZ-BOARD

Port	Setting
P5	Connected with SigmaStudio USBi
P4	Debug port connected to ADZS-ICE-2000/ICE-1000
P10	Connected to Power

Table 16: Port Settings for ADSP-21569 EZ-BOARD

3.1.8.6 EV-21569-SOM with EV-SOM-CRR EZ-Kit

The board settings for running SigmaStudio on EV-21569-SOM with EV-SOM-CRR EZ-Kit, using the Default Application is given in the table below.

Switch	Setting
SW1	BOOT Mode Switch Default Position 0

Table 17: Switch Settings for EV-21569-SOM BOARD

Jumper	Setting
JP1	2-3 connected

Table 18: Jumper Settings for EV-21569-SOM BOARD

Port	Setting
P1	Debug port connected to ADZS-ICE-2000/ICE-1000

Table 19: Port Settings for EV-21569-SOM BOARD

Switch	Setting
SW1 (CRR)	OFF, OFF, OFF, OFF, OFF, OFF

Table 20: Switch Settings for EV-SOM-CRR EZ-Kit

Jumper	Setting
JP1	Not connected
JP2	Not connected

Table 21: Jumper Settings for EV-SOM-CRR EZ-Kit

Port	Setting
P3	Connected with SigmaStudio USBi
P8	Connected to Power
J3(SOMCRR) – J2 (SOM)	Connected
J6(SOMCRR) – J1 (SOM)	Connected
J5(SOMCRR) – J3 (SOM)	Connected

Table 22: Port Settings for EV-SOM-CRR EZ-Kit

3.1.8.7 EV-21593-SOM with EV-SOM-CRR EZ-Kit

The board settings for running SigmaStudio on EV-21593-SOM with EV-SOM-CRR EZ-Kit, using the Default Application is given in the table below.

Switch	Setting
SW1	BOOT Mode Switch Default Position 0

Table 23: Switch Settings for EV-21593-SOM BOARD

Jumper	Setting
JP1	2-3 connected

Table 24: Jumper Settings for EV-21593-SOM BOARD

Port	Setting
P1	Debug port connected to ADZS-ICE-2000/ICE-1000

Table 25: Port Settings for EV-21593-SOM BOARD

Switch	Setting
SW1 (CRR)	OFF, OFF, OFF, OFF, OFF, OFF

Table 26: Switch Settings for EV-SOMCRR EZ-Kit

Jumper	Setting
JP1	Not connected
JP2	Not connected

Table 27: Jumper Settings for EV-SOMCRR EZ-Kit

Port	Setting
P3	Connected with SigmaStudio USBi
P8	Connected to Power
J3(SOMCRR) – J2 (SOM)	Connected
J6(SOMCRR) – J1 (SOM)	Connected

J5(SOMCRR) – J3 (SOM)	Connected

Table 28: Port Settings for EV-SOMCRR EZ-Kit

3.1.8.8 EV-SC594-SOM with EV-SOM-CRR EZ-Kit

The board settings for running SigmaStudio on EV-SC594-SOM with EV-SOM-CRR EZ-Kit, using the Default Application is given in the table below.

Switch	Setting
SW1	BOOT Mode Switch Default Position 0

Table 29: Switch Settings for EV-SC594-SOM BOARD

Jumper	Setting
JP1	2-3 connected

Table 30: Jumper Settings for EV-SC594-SOM BOARD

Port	Setting
P1	Debug port connected to ADZS-ICE-2000/ICE-1000

Table 31: Port Settings for EV-SC594-SOM BOARD

Switch	Setting
SW1 (CRR)	OFF, OFF, OFF, OFF, OFF, OFF

Table 32: Switch Settings for EV-SOMCRR EZ-Kit

Jumper	Setting
JP1	Not connected
JP2	Not connected

Table 33: Jumper Settings for EV-SOMCRR EZ-Kit

Port	Setting
P3	Connected with SigmaStudio USBi
P8	Connected to Power
J3(SOMCRR) – J2 (SOM)	Connected
J6(SOMCRR) – J1 (SOM)	Connected
J5(SOMCRR) – J3 (SOM)	Connected

Table 34: Port Settings for EV-SOMCRR EZ-Kit

4 Safety Precaution and Recommendations

Refer to the *AE_42_SS4G_ReleaseNotes.pdf* **[1]** for safety precaution and recommendations.

5 PC Software Setup

Install the following software/tools in the listed order.

1. Install CrossCore Embedded Studio.
   SigmaStudio uses the tools that are included with CrossCore Embedded Studio. Install CrossCore Embedded Studio if it is not already installed on the SigmaStudio Host PC.

2. Install ADSP-SC5xx EZ-Kit Lite Board Support package.

3. Install SigmaStudio for SHARC (ADSP-SC5xx) bundle.
   This bundle contains the SigmaStudio Installer as well. There is no need to install SigmaStudio separately. SigmaStudio for SHARC (ADSP-SC5xx/ADSP-215xx) support is automatically added when SigmaStudio is re-launched. SigmaStudio automatically detects the installation directory and uses it.

Note that if more than one version of the CrossCore Embedded Studio tool-chain is detected, the ‘SHARC’ tab in the ‘Settings’ window allows for tool-chain selection.

Refer section 5 of **[1]** for detailed information on the dependent software/tools.

6 Demo application

This section explains the following

- Loader File Flashing and Generation
- How to run the Example Schematics which are provided with this package using the pre-built executables of the Demo application.
- Rebuilding the demo Application.
- Creating and running new schematics.

6.1 Loader file Flashing and Generation

6.1.1 Loader file generation

The following command is used to generate a loader file for ADSP-SC584 processor.

```
"C:\Analog Devices\CrossCore Embedded Studio 2.10.1\elfloader.exe" -proc ADSP-SC584 -core0=SS_App_Core0.exe -init ezkitSC584_initcode_core0_v10.exe -core1=SS_App_Core1.dxe -core2=SS_App_Core2.dxe -NoFinalTag=SS_App_Core0.exe -NoFinalTag=SS_App_Core1.dxe -b SPI -f BINARY -Width 8 -bcode 0x1 -verbose -o SS_App_SC584.ldr
```

The above command will generate a loader file by the name SS_App_SC584.ldr. Copy the application DXEs of all the three cores of ADSP-SC584 processor to the *“< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples \LDR*” folder. In case, ARM Core executable is missing .exe at the end, add the .exe to the name of ARM executable. The file “ezkitSC584_initcode_core0_v10.exe” is a part of CCES. It can be found at “C:\Analog Devices\CrossCore Embedded Studio 2.10.1\SHARC\ldr”. Copy this file to *“< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*” folder.

The following command is used to generate a loader file for ADSP-SC589 processor.

```
"C:\Analog Devices\CrossCore Embedded Studio 2.10.1\elfloader.exe" -proc ADSP-SC589 -core0=SS_App_Core0.exe -init ezkitSC589_initcode_core0_v10.exe -core1=SS_App_Core1.dxe -core2=SS_App_Core2.dxe -NoFinalTag=SS_App_Core0.exe -NoFinalTag=SS_App_Core1.dxe -b SPI -f BINARY -Width 8 -bcode 0x1 -verbose -o SS_App_SC589.ldr
```

The above command will generate a loader file by the name SS_App_SC589.ldr. Copy the application DXEs of all the three cores of ADSP-SC589 processor to the *“< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*” folder. In case, ARM Core executable is missing .exe at the end, add the .exe to the name of ARM executable. The file “ezkitSC589_initcode_core0_v10.exe” is a part of CCES. It can be found at “C:\Analog Devices\CrossCore Embedded Studio 2.10.1\SHARC\ldr”. Copy this file to *“< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*” folder.

The following command is used to generate a loader file for ADSP-SC573 processor.

```
"C:\Analog Devices\CrossCore Embedded Studio 2.10.1\elfloader.exe" -proc ADSP-
SC573 -core0=SS_App_Core0.exe -init ezkitSC573_initcode_core0.exe -
core1=SS_App_Core1.dxe -core2=SS_App_Core2.dxe -NoFinalTag=SS_App_Core0.exe -
NoFinalTag=SS_App_Core1.dxe -b SPI -f BINARY -Width 8 -bcode 0x1 -verbose -o
SS_App_SC573.ldr
```

The above command will generate a loader file by the name SS_App_SC573.ldr. Copy the application DXEs of all the three cores of ADSP-SC573 processor to the “*< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*” folder. In case, ARM Core executable is missing .exe at the end, add the .exe to the name of ARM executable. The file “ezkitSC573_initcode_core0.exe” is a part of CCES. It can be found at “C:\Analog Devices\CrossCore Embedded Studio 2.10.1\SHARC\ldr”. Copy this file to “*< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*” folder.

The following command is used to generate a loader file for ADSP-21569 processor.

```
"C:\Analog Devices\CrossCore Embedded Studio 2.10.1\elfloader.exe" -proc ADSP-
21569 SS_App_Core1 -init ezkit21569_initcode.dxe -b SPI -f BINARY -Width 8 -
bcode 0x1 -verbose -o SS_App_21569.ldr
```

The above command will generate a loader file by the name SS_App_21569.ldr. Copy the application DXE of ADSP-21569 processor to the “*< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*” folder. The file “ezkit21569_initcode.dxe” is a part of CCES. It can be found at “C:\Analog Devices\CrossCore Embedded Studio 2.10.1\SHARC\ldr”. Copy this file to “*< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*” folder.

The following command is used to generate a loader file for ADSP-SC594 processor.

```
"C:\Analog Devices\CrossCore Embedded Studio 2.10.1\elfloader.exe" -proc ADSP-
SC594 -core0=SS_App_Core0.exe -init ezkitSC594W_initcode_core0.exe -
core1=SS_App_Core1.dxe -core2=SS_App_Core2.dxe -NoFinalTag=SS_App_Core0.exe -
NoFinalTag=SS_App_Core1.dxe -b SPI -f BINARY -Width 8 -bcode 0x1 -verbose -o
SS_App_SC594.ldr
```

The above command will generate a loader file by the name SS_App_SC594.ldr. Copy the application DXEs of all the three cores of ADSP-SC594 processor to the “*< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*” folder. In case, ARM Core executable is missing .exe at the end, add the .exe to the name of ARM executable. The file “ezkitSC594W_initcode_core0.exe” is a part of CCES. It can be found at “C:\Analog Devices\CrossCore Embedded Studio 2.10.1\SHARC\ldr”. Copy this file to “*< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*” folder.

The following command is used to generate a loader file for ADSP-21593 processor.

```
"C:\Analog Devices\CrossCore Embedded Studio 2.10.1\elfloader.exe" -proc ADSP-21593 -core1=SS_App_Core1.dxe -init ezkit21593_initcode_core1.dxe -core2=SS_App_Core2.dxe -NoFinalTag=SS_App_Core1.exe -b SPI -f BINARY -Width 8 -bcode 0x1 -verbose -o SS_App_21593.ldr
```

The above command will generate a loader file by the name SS_App_21593.ldr. Copy the application DXEs of all the two cores of ADSP-21593 processor to the *"< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*" folder. The file "ezkit21593_initcode_core1.exe" is a part of CCES. It can be found at "C:\Analog Devices\CrossCore Embedded Studio 2.10.1\SHARC\ldr". Copy this file to *"< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*" folder.

Note: *The "< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR" folder contains the loader files for EV-21569-SOM with EV-SOM-CRR EZ-Kit platform by default for ADSP-21569 processor.*

6.1.2 Programming the flash

Run the Flash_ADSPSC5xx.bat batch file present in *"< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\utilities\FlashProgrammer\*" folder to program the flash on ADSP-SC5xx EZ-Board using the pre-built SS_App_SC5xx.ldr loader file present in *"< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*" folder.

Use the following command to program the flash on ADSP-SC584 EZ-Board using the pre-built SS_App_SC584.ldr loader file present in *"< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*" folder.

```
Flash_ADSPSC5xx.bat 584 XXXX "<Software Modules folder>\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR\SS_App_SC584.ldr"
```

Use the following command to program the flash on ADSP-SC589 EZ-Board using the pre-built SS_App_SC589.ldr loader file present in *"< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*" folder.

```
Flash_ADSPSC5xx.bat 589 XXXX "<Software Modules folder>\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR\SS_App_SC589.ldr"
```

Use the following command to program the flash on ADSP-SC573 EZ-Board using the pre-built SS_App_SC573.ldr loader file present in *"< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*" folder.

```
Flash_ADSPSC5xx.bat 573 XXXX "<Software Modules folder>\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR\SS_App_SC573.ldr"
```

The ADSP-21569 example demo and demoUc application loader files in `"<Software Modules folder>\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR\"` are generated for EV-

21569-SOM with EV-SOM-CRR EZ-Kit platform. The executable for ADSP-21569 EZ-Kit platform can be generated by removing the *"ADSP_21569_SOM"* pre-processor macro from project compiler options and rebuilding the application. The loader file can be generated from the executable by following the instructions in section 6.1.1. Use the following command to program the flash on ADSP-21569 EZ-Board using the re-built SS_App_21569.ldr loader file.

```
Flash_2156x.bat 21569 XXXX "<Software Modules folder>\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR\SS_App_21569.ldr"
```

Use the following command to program the flash on EV-21569-SOM with EV-SOM-CRR EZ-Kit using the pre-built SS_App_21569.ldr loader file present in *"< Software Modules folder >* *\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*" folder.

```
Flash_2156x_SOM.bat 21569 XXXX "<Software Modules folder>\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR\SS_App_21569.ldr"
```

Use the following command to program the flash on EV-SC594-SOM with EV-SOM-CRR EZ-Kit using the pre-built SS_App_SC594.ldr loader file present in *"< Software Modules folder >* *\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*" folder.

```
Flash_ADSPSC59x.bat 594 XXXX "<Software Modules folder>\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR\SS_App_SC594.ldr"
```

Use the following command to program the flash on EV-21593-SOM with EV-SOM-CRR EZ-Kit using the pre-built SS_App_21593.ldr loader file present in *"< Software Modules folder >* *\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR*" folder.

```
Flash_ADSP2159x.bat 21593 XXXX "<Software Modules folder>\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LDR\SS_App_21593.ldr"
```

XXXX stands for 1000 or 2000 based on the emulator used for flashing.

The boot switch should be set to '1' and a hard reset of the EZ-Board is required to run the demo.

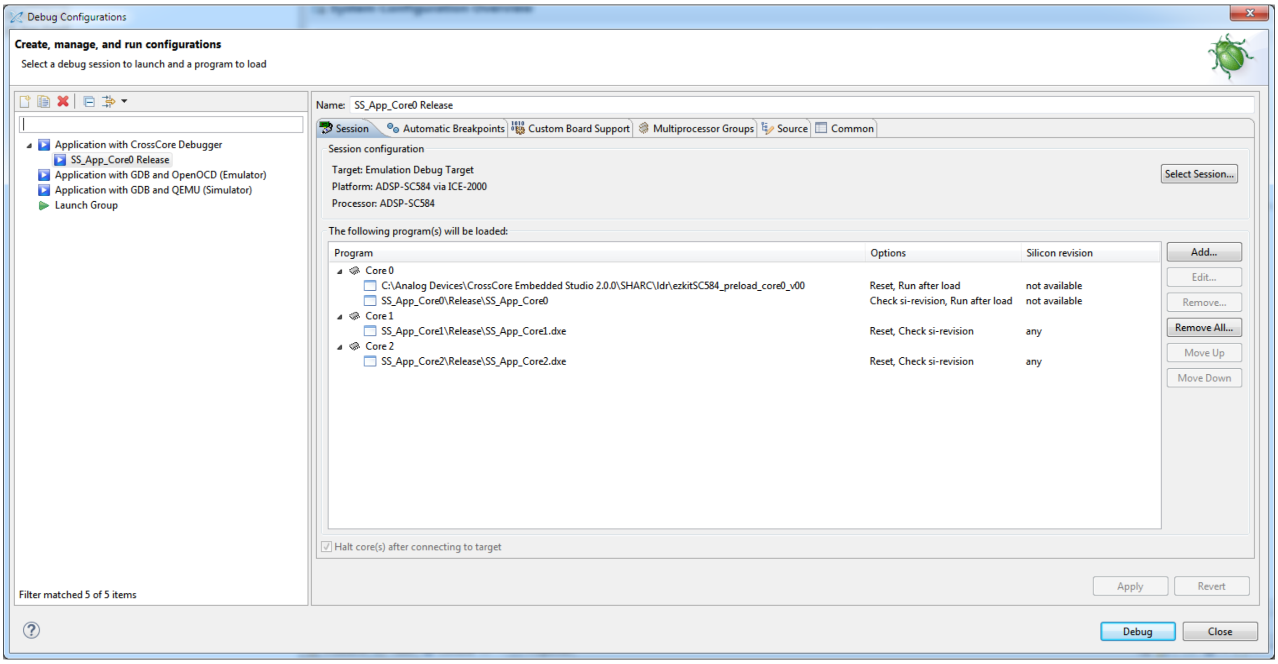
6.2 Getting Started with Example Schematics

Example Schematics for ADSP-SC5xx are provided in *"< Software Modules folder >* *\SigmaStudioForSHARC-SH-Rel4.7.0\Host\Examples\Sample Schematics*" folder. Follow the instructions given below to execute the example schematics for specific targets using the Demo application.

6.2.1 Running the example schematics on ADSP-SC573/ADSP-SC584/ADSP-SC589/ADSP-SC589-SAM/ADSP-SC594 target using the Demo application

1. Establish the hardware setup as described in Section 3.1.

2. Install all software and tools mentioned in Section 2.
3. Launch CrossCore Embedded Studio.
4. Uncheck “Build (if required) before launching” checkbox in Window->Preferences->Run/Debug->Launching.
5. Select Run→Debug Configurations. Create a new Debug configuration under *‘Application with CrossCore Debugger’*. Select appropriate processor, connection type and platform. Click finish.
6. Load the prebuilt DXE’s for appropriate target ‘SS_App_Core0’ on Core 0, ‘*SS_App_Core1.dxe*’ on Core 1 and ‘*SS_App_Core2.dxe*’ on Core 2 as shown in Figure 14. Prebuilt DXEs of each project can be found inside the respective ‘Release’ folders of each of the projects.



Debug Configurations

Create, manage, and run configurations

Select a debug session to launch and a program to load

Name: SS_App_Core0 Release

Session configuration

Target: Emulation Debug Target

Platform: ADSP-SC584 via ICE-2000

Processor: ADSP-SC584

The following program(s) will be loaded:

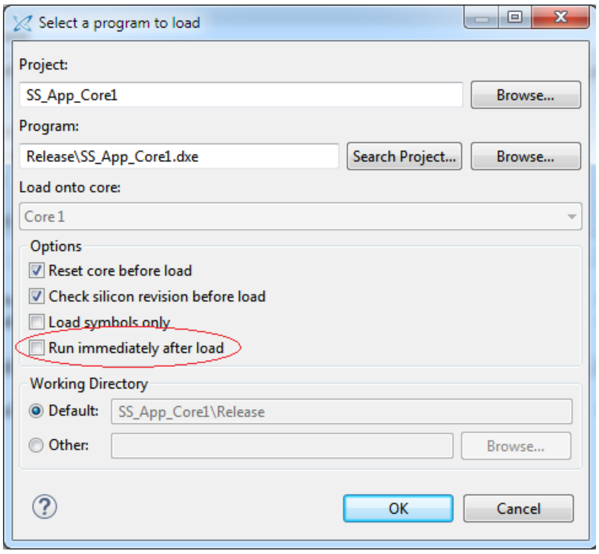
Program	Options	Silicon revision
Core 0		
C:\Analog Devices\CrossCore Embedded Studio 2.0.0\SHARC\ldr\ezkitSC584_preload_core0_v00	Reset, Run after load	not available
SS_App_Core0\Release\SS_App_Core0	Check si-revision, Run after load	not available
Core 1		
SS_App_Core1\Release\SS_App_Core1.dxe	Reset, Check si-revision	any
Core 2		
SS_App_Core2\Release\SS_App_Core2.dxe	Reset, Check si-revision	any

☑ Halt core(s) after connecting to target

Filter matched 5 of 5 items

Figure 14: Debug Configuration

7. Double click on ‘*SS_App_Core1*.dxe’ of Core 1 and uncheck *‘Run immediately after load’* option as shown in the Figure 15. Repeat the same for ‘*SS_App_Core2.dxe*’ for Core 2.



Select a program to load

Project: SS_App_Core1

Program: Release\SS_App_Core1.dxe

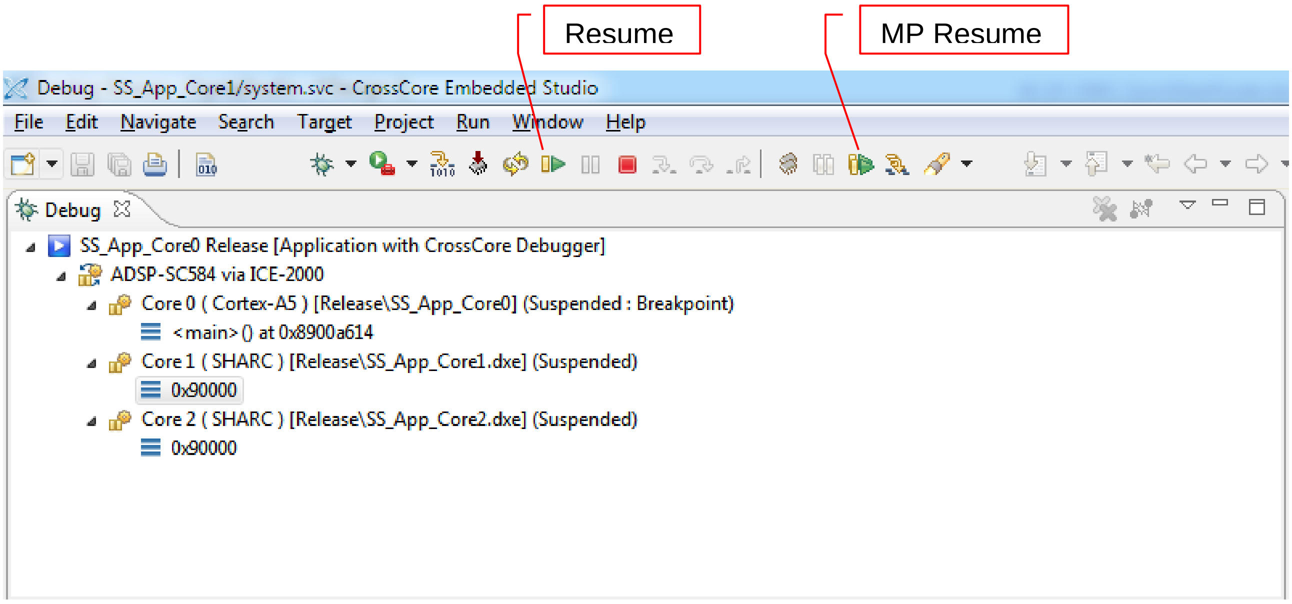
Load onto core: Core 1

Options
- ☑ Reset core before load
- ☑ Check silicon revision before load
- □ Load symbols only
- □ Run immediately after load

Working Directory
- ● Default: SS_App_Core1\Release
- ○ Other:

Figure 15: Select a program to load window

8. Reset the board and press ‘Debug’ to launch the debug session. Click on “MP Resume” to run Core 0 (ARM). This enables the Core 1 and 2 (SHARC). Then individually select and run Core 1 and Core 2 by clicking on “Resume” as shown in Figure 16



Debug - SS_App_Core1/system.svc - CrossCore Embedded Studio

```
SS_App_Core0 Release [Application with CrossCore Debugger]
  ADSP-SC584 via ICE-2000
    Core 0 ( Cortex-A5 ) [Release\SS_App_Core0] (Suspended : Breakpoint)
      <main> () at 0x8900a614
    Core 1 ( SHARC ) [Release\SS_App_Core1.dxe] (Suspended)
      0x90000
    Core 2 ( SHARC ) [Release\SS_App_Core2.dxe] (Suspended)
      0x90000
```

Figure 16: Debug Configuration (Resume and MP Resume)

9. Launch SigmaStudio.
10. Launch the AddIns browser by clicking *Add-Ins browser* option in *Tools* menu.

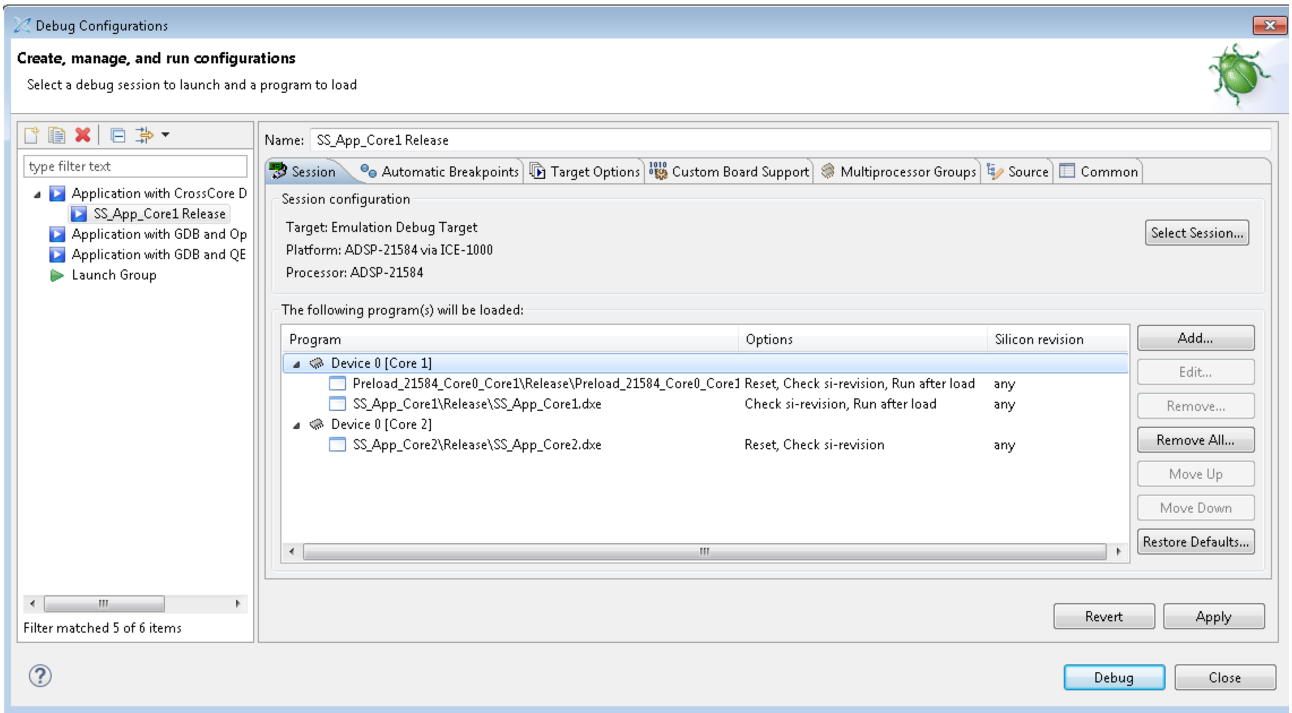
11. Check if the latest version of *SigmaStudioForSHARC (ADSP-SC5xx/ADSP-215xx) DLLs*, namely, SharcPubLib.dll, SharcModules.dll, SharcDesigner.dll are selected. If not selected, select the latest version and disable the older versions. Save and close the Add-ins browser and re-launch SigmaStudio.
12. Open *Volume_Mute_Block_SC5xx.dspproj* from "< *Software Modules install folder* > \ *SigmaStudioForSHARC-SH-Rel4.7.0\Host\Examples\Sample Schematics\ADSP-SC5xx*" folder.
13. Select the prebuilt Application DXE's for both the SHARC's by clicking on *'Select Application DXE'* in SHARC0/SHARC1 tab of SigmaStudio IC control window.
14. Connect the audio inputs to the EZ-BOARD and play audio on PC.
15. Press Link-Compile-Download.
16. The output audio will be heard through speakers/headphones connected to the EZ-BOARD.
17. The checkbox inside the mute cells in the schematic can be checked/unchecked to mute/unmute any channel.

Note: In the case of ADSP-SC589-SAM, the application DXE's are named: *'SS_App_SAM_Core0'* for Core 0, '*SS_App_SAM_Core1.dxe*' for Core 1 and '*SS_App_SAM_Core2.dxe*' for Core 2 and the example schematic can be found in "< *Software Modules install folder* > \ *SigmaStudioForSHARC-SH-Rel4.7.0\Host\Examples\Sample Schematics\ADSP-SC589-Mini*" folder.

6.2.2 Running the example schematics on ADSP-21573/ADSP-21584/ADSP-21593 target using the Demo application

1. Establish the hardware setup as described in Section 3.1.
2. Install all software and tools mentioned in Section 2.
3. Launch CrossCore Embedded Studio.
4. Uncheck "Build (if required) before launching" checkbox in Window->Preferences->Run/Debug->Launching.
5. Select Run→Debug Configurations. Create a new Debug configuration under *'Application with CrossCore Debugger'*. Select appropriate processor, connection type and platform. Click finish.

6. Load the prebuilt DXE’s for appropriate target ‘21584_preload_Core1.dxe” or “21573_preload_Core1.dxe” for 21584 and 21573 respectively, and “SS_App_Core1.dxe” on Core 1 and ‘SS_*App_Core2.dxe*’ on Core 2 as shown in Figure 17. Prebuilt DXEs of each project can be found inside the respective ‘Release’ folders of each of the projects. Prebuilt preload DXEs ADSP-21593 target will get automatically loaded from “C:\Analog Devices\CrossCore Embedded Studio 2.10.1\SHARC\ldr” path.



Name: SS_App_Core1 Release

Session configuration

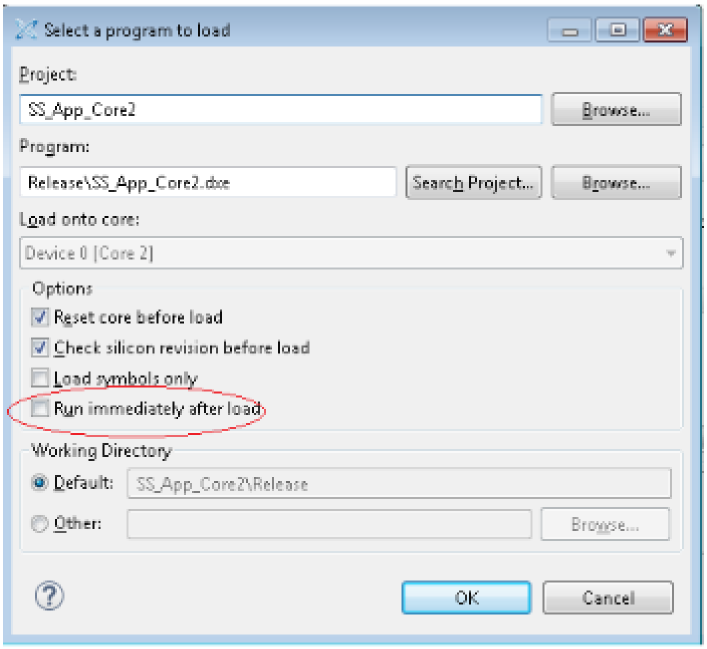
Target: Emulation Debug Target
Platform: ADSP-21584 via ICE-1000
Processor: ADSP-21584

The following program(s) will be loaded:

Program	Options	Silicon revision
Device 0 [Core 1]		
Preload_21584_Core0_Core1\Release\Preload_21584_Core0_Core1	Reset, Check si-revision, Run after load	any
SS_App_Core1\Release\SS_App_Core1.dxe	Check si-revision, Run after load	any
Device 0 [Core 2]		
SS_App_Core2\Release\SS_App_Core2.dxe	Reset, Check si-revision	any

Figure 17: Debug Configuration

7. Double click on ‘*SS_App_Core2.*dxe’ of Core 2 and uncheck *‘Run immediately after load’* option as shown in Figure 18.



Select a program to load

Project: SS_App_Core2

Program: Release\SS_App_Core2.dxe

Load onto core: Device 0 [Core 2]

Options

☑ Reset core before load

☑ Check silicon revision before load

□ Load symbols only

□ Run immediately after load

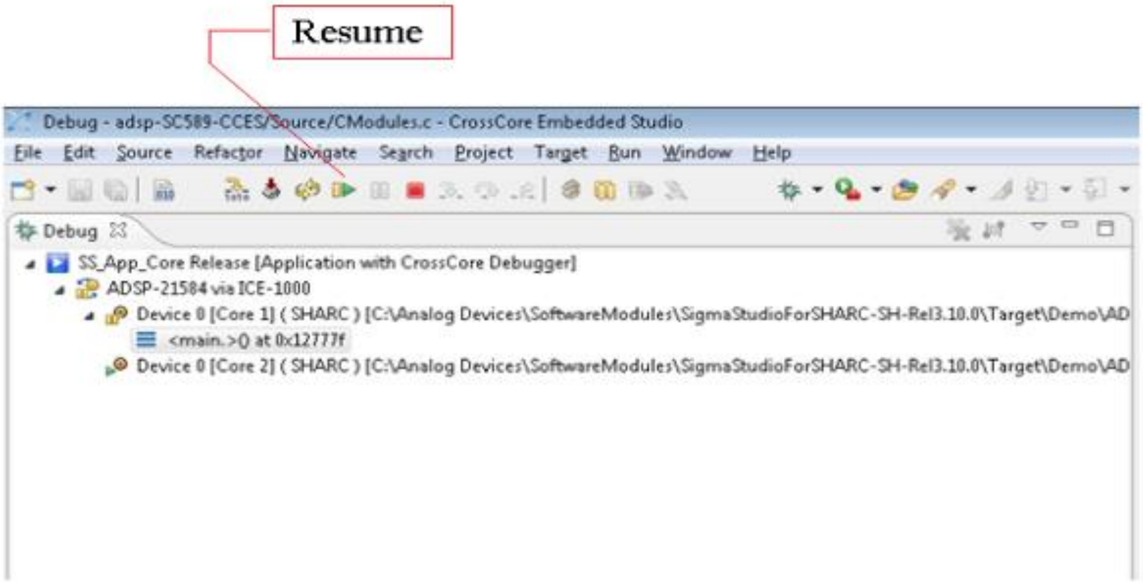
Working Directory

● Default: SS_App_Core2\Release

○ Other:

Figure 18: Select a program to load window

8. Reset the board and press ‘Debug’ to launch the debug session. Click on “Resume” to run Core 1(SHARC). This enables the Core 2 (SHARC). Then individually select and run Core Core 2 by clicking on “Resume” as shown in Figure 19.



Debug - adsp-SC589-CCES/Source/CModules.c - CrossCore Embedded Studio

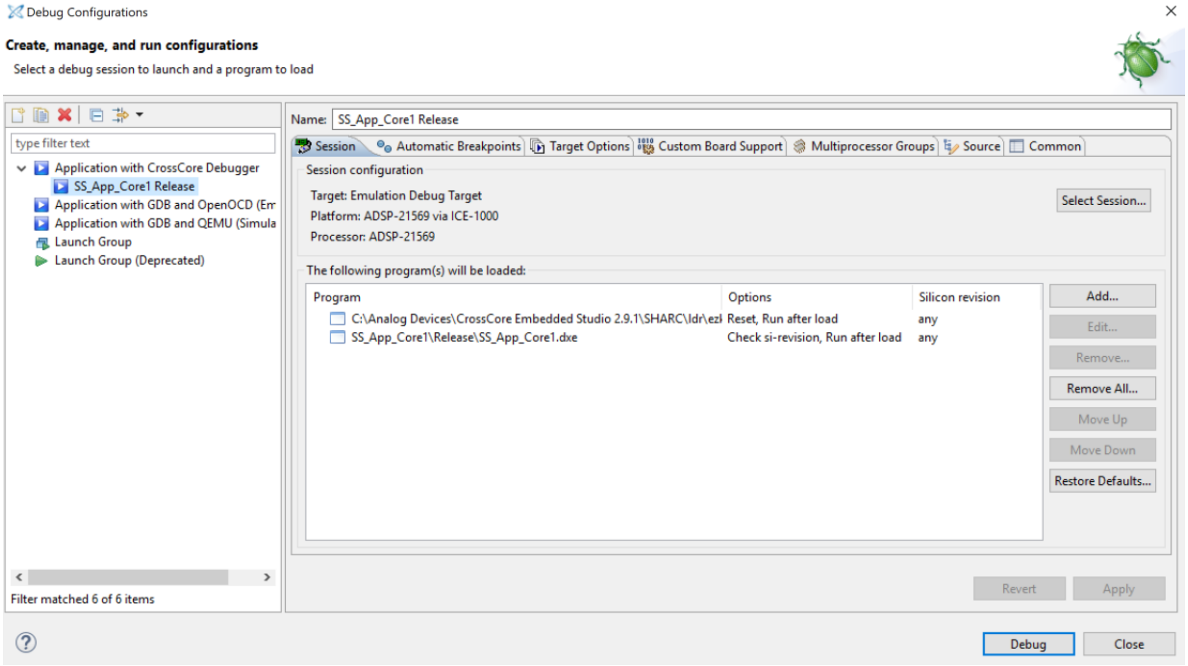
SS_App_Core Release [Application with CrossCore Debugger]
ADSP-21584 via ICE-1000
Device 0 [Core 1] (SHARC) [C:\Analog Devices\SoftwareModules\SigmaStudioForSHARC-SH-Rel3.10.0\Target\Demo\AD
<main.>() at 0x12777f
Device 0 [Core 2] (SHARC) [C:\Analog Devices\SoftwareModules\SigmaStudioForSHARC-SH-Rel3.10.0\Target\Demo\AD

Figure 19 Debug Configuration (Resume)

9. Launch SigmaStudio.
10. Launch the AddIns browser by clicking *Add-Ins browser* option in *Tools* menu.
11. Check if the latest version of *SigmaStudioForSHARC(ADSP-SC5xx/ADSP-215xx)* DLLs, namely, SharcPubLib.dll, SharcModules.dll, SharcDesigner.dll are selected. If not selected, select the latest version and disable the older versions. Save and close the Add-ins browser and re-launch SigmaStudio.
12. Open *Volume_Mute_Block_SC5xx.dspproj* from "< *Software Modules install folder* > \ *SigmaStudioForSHARC-SH-Rel4.7.0\Host\Examples\Sample Schematics\ADSP-SC5xx*" folder.
13. Select the prebuilt Application DXE's for both the SHARC's by clicking on *'Select Application DXE'* in SHARC0/SHARC1 tab of SigmaStudio IC control window.
14. Connect the audio inputs to the EZ-BOARD and play audio on PC.
15. Press Link-Compile-Download.
16. The output audio will be heard through speakers/headphones connected to the EZ-BOARD.
17. The checkbox inside the mute cells in the schematic can be checked/unchecked to mute/unmute any channel.

6.2.3 Running the example schematics on ADSP-21569 target using the Demo application

1. Establish the hardware setup as described in Section 3.1.
2. Install all software and tools mentioned in Section 2.
3. Launch CrossCore Embedded Studio.
4. Uncheck "Build (if required) before launching" checkbox in Window->Preferences->Run/Debug->Launching.
5. Select Run→Debug Configurations. Create a new Debug configuration under *'Application with CrossCore Debugger'*. Select appropriate processor, connection type and platform. Click finish.
6. The prebuilt DXE's for the target 'ezkit21569_preload.dxe" is automatically loaded on Core1. Load the prebuilt "SS_App_Core1.dxe" on Core 1. Prebuilt DXEs of each project can be found inside the respective 'Release' folders of each of the projects.



Debug Configurations

Create, manage, and run configurations

Select a debug session to launch and a program to load

Name: SS_App_Core1 Release

Session configuration

Target: Emulation Debug Target
Platform: ADSP-21569 via ICE-1000
Processor: ADSP-21569

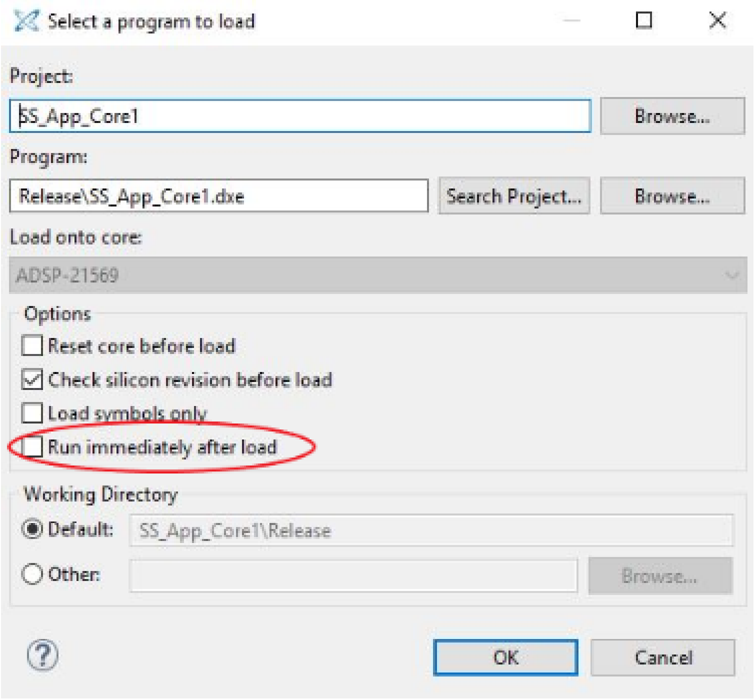
The following program(s) will be loaded:

Program	Options	Silicon revision
□ C:\Analog Devices\CrossCore Embedded Studio 2.9.1\SHARC\ldr\ezl	Reset, Run after load	any
□ SS_App_Core1\Release\SS_App_Core1.dxe	Check si-revision, Run after load	any

Filter matched 6 of 6 items

Figure 20: Debug Configuration

7. Double click on ‘*SS_App_Core1.*dxe’ of Core 1 and uncheck *‘Run immediately after load’* option as shown in Figure 21.



Select a program to load

Project: SS_App_Core1

Program: Release\SS_App_Core1.dxe

Load onto core: ADSP-21569

Options

☐ Reset core before load

☑ Check silicon revision before load

☐ Load symbols only

☐ Run immediately after load

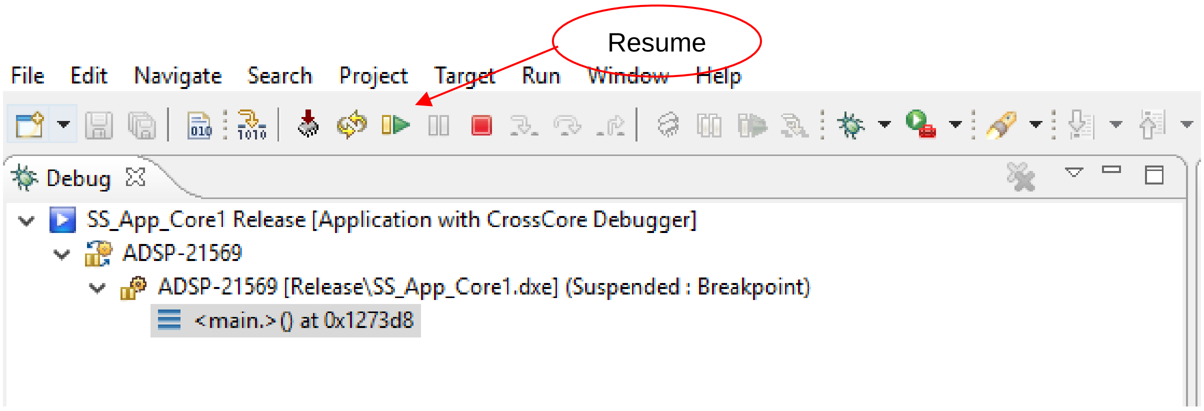
Working Directory

● Default: SS_App_Core1\Release

○ Other:

Figure 21: Select a program to load window

8. Reset the board and press ‘Debug’ to launch the debug session. Click on “Resume” to run Core 1(SHARC) as shown in Figure 22.



SS_App_Core1 Release [Application with CrossCore Debugger]
ADSP-21569
ADSP-21569 [Release\SS_App_Core1.dxe] (Suspended : Breakpoint)
<main.>() at 0x1273d8

Figure 22 Debug Configuration (Resume)

9. Launch SigmaStudio.
10. Launch the AddIns browser by clicking *Add-Ins browser* option in *Tools* menu.
11. Check if the latest version of *SigmaStudioForSHARC(ADSP-SC5xx/ADSP-215xx)* DLLs, namely, SharcPubLib.dll, SharcModules.dll, SharcDesigner.dll are selected. If not selected,

select the latest version and disable the older versions. Save and close the Add-ins browser and re-launch SigmaStudio.

12. Open *Volume_Mute_Block_2156x.dspproj* from "< *Software Modules install folder* > \ *SigmaStudioForSHARC-SH-Rel4.7.0\Host\Examples\Sample Schematics\ADSP-2156x*" folder.
13. Select the prebuilt Application DXE for SHARC Core 0 by clicking on *'Select Application DXE'* in SHARC0 tab of SigmaStudio IC control window.
14. Connect the audio inputs to the EZ-BOARD and play audio on PC.
15. Press Link-Compile-Download.
16. The output audio will be heard through speakers/headphones connected to the EZ-BOARD.
17. The checkbox inside the mute cells in the schematic can be checked/unchecked to mute/unmute any channel.

Note: *The ADSP-21569 example demo application has support for EV-21569-SOM with EV-SOM-CRR EZ-Kit platform by default. For running the application on ADSP-21569 EZ-Kit platform, the "ADSP_21569_SOM" pre-processor macro has to be removed from project compiler options and the application needs to be rebuilt.*

6.3 Rebuilding the Demo Application

The Default Application for ADSP-SC573, ADSP-SC584, ADSP-SC589, ADSP-SC594, ADSP-21584, ADSP-21573, ADSP-21593 and ADSP-21569 are supplied with the package. Follow the steps given below to rebuild the Application. The projects can be found under target specific subfolders inside *"< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\Demo*\".

1. Launch CrossCore Embedded Studio on the SigmaStudio Host PC. Specify the workspace when prompted.
2. Select File→Import. Import *SS_App_Core0*, *SS_App_Core1* and *SS_App_Core2* projects into CrossCore Embedded Studio for ADSP-SC573, ADSP-SC584, ADSP-SC589, ADSP-SC589 SAM and ADSP-SC594.
3. Select File→Import. Import *SS_App_Core1* and *SS_App_Core2* projects into CrossCore Embedded Studio for ADSP-21584, ADSP-21573 and ADSP-21593.
4. Select File→Import. Import *SS_App_Core1* project into CrossCore Embedded Studio for ADSP-21569.

5. Select all the three projects (*SS_App_Core0*, *SS_App_Core1* and *SS_App_Core2)*. Clean and rebuild the projects in the required configuration (Release/Debug) for ADSP-SC573, ADSP-SC584, ADSP-SC589 and ADSP-SC594.
6. Select the two projects (*SS_App_Core1* and *SS_App_Core2)*. Clean and rebuild the projects in the required configuration (Release/Debug) for ADSP-21584, ADSP-21573 and ADSP-21593.
7. Select the project (*SS_App_Core1)*. Clean and rebuild the projects in the required configuration (Release/Debug) for ADSP-21569.
8. The newly generated executables will be present in “Debug” or “Release” folder based on the chosen configuration.

Note: *There is a linker warning related to “RESERVE_EXPAND” used in .ldf file which is expected and ignore it.*

6.3.1 Launching Debug Configuration and Booting the Target

Follow steps 5 to 8 from section 6.2 to launch debug configuration and to boot the Target.

Note: Select the rebuilt DXE’s in step 6 of section 6.3 instead of prebuilt DXE’s

6.4 Creating and Running a New Simple Schematic using the Demo application

6.4.1 Creating a New Schematic: ADSP-SC5xx and ADSP-2158x/ADSP-2157x/ADSP-2159x

Follow the steps below to create a SigmaStudio for SHARC Schematic for ADSP-SC5xx and ADSP-215xx.

1. Launch SigmaStudio

2. Launch the AddIns browser by clicking Add-in’s browser option in Tools menu.

3. Check if the latest version of SigmaStudioForSHARC (ADSP-SC5xx/ADSP-215xx) DLLs, namely, SharcPubLib.dll, SharcModules.dll, SharcDesigner.dll are selected. If not selected, select the latest version and disable the older versions. Save and close the Add-ins browser and re-launch SigmaStudio.

4. Open a new project. The *‘Tree ToolBox’* on the left side displays all the supported processors and communication channels.

5. Drag two ‘ADSP-SC5xx’ from the *‘Processor’* tree and drop it into the *‘Hardware Configuration’* design window. This block is referred to in this document, as the ‘IC’.

6. Drag ‘USBi’ from the *‘Communication Channels’* tree and drop it into the *‘Hardware Configuration’* design window. If the USBi driver is installed properly and the USBi is connected to the PC, the word “USB” is in a green box. If there is a problem with driver installation or connection, the box with text “USB” is shown in red. In such a case, remove the USBi from the *‘Hardware Configuration’* design window, verify the USB connection, and repeat this step.

7. Connect the first blue (output) pin of the ‘USBi’ to the green (input) pin of the ‘IC 1’ block. Connect the second blue (output) pin of the ‘USBi’ to the green (input) pin of the ‘IC 2’ block as shown in Figure 23

8. A drop-down menu is enabled adjacent to the connected pin of ‘USBi’. Select ‘SPI 0x1 ADR0’ from the drop-down menu for both the ICs.

9. If “Aardvark Host I^{2}C/SPI Host Adapter” is being used, drag ‘Aardvark’ from the *‘Communication Channels’* tree and drop it into the *‘Hardware Configuration’* design window. If the Aardvark driver is installed properly and the Aardvark is connected to the PC, the word “USB” is in a green box. If there is a problem with driver installation or connection, the box with text “USB” is shown in red. In such a case, remove the ‘Aardvark’ from the *‘Hardware Configuration’* design window, verify the USB connection, and repeat this step.

10. Connect the first blue (output) pin of the ‘Aardvark to the green (input) pin of the ‘IC 1’ block. Connect the second blue (output) pin of the ‘Aardvark to the green (input) pin of the ‘IC 2’ block as shown in Figure 24.

11. A drop-down menu is enabled adjacent to the connected pin of ‘Aardvark. Select ‘SPI 0x1 (1)’ from the drop-down menu for both the ICs.

12. An IC control window is created for every IC dragged as part of step 0. Select “Core 1” in the “Default SHARC Core” section of the "Main” tab of IC control window for “IC 1”. Select “Core 2” in the “Default SHARC Core” section of the "Main” tab of IC control window for “IC 2”. Leave the other settings unchanged.

13. Go to the *‘Block Schematic’* tab. The toolbox on the left side lists the Modules available for the SHARC DSP.

14. Drag and drop the desired Modules in the Schematic workspace and interconnect the Modules to form a DSP design.

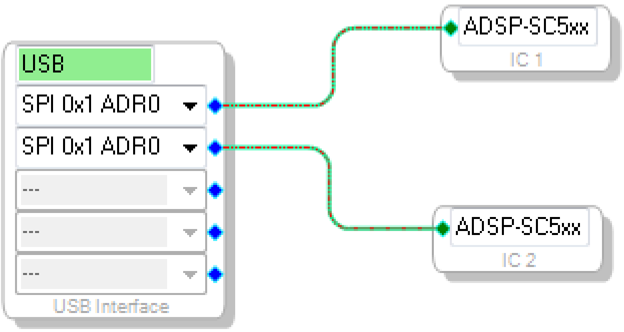


Figure 23: USBi to ADSP-SC5xx connection

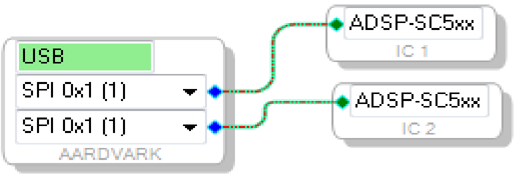


Figure 24: Aardvark to ADSP-SC5xx connection

6.4.2 Creating a New Schematic: ADSP-2156x

Follow the steps below to create a SigmaStudio for SHARC Schematic for ADSP-2156x.

1. Launch SigmaStudio

2. Launch the AddIns browser by clicking Add-Ins browser option in Tools menu.

3. Check if the latest version of SigmaStudioForSHARC(ADSP-SC5xx/ADSP-215xx) DLLs, namely, SharcPubLib.dll, SharcModules.dll, SharcDesigner.dll are selected. If not selected, select the latest version and disable the older versions. Save and close the Add-ins browser and re-launch SigmaStudio.

4. Open a new project. The *'Tree ToolBox'* on the left side displays all the supported processors and communication channels.

5. Drag a 'ADSP-SC5xx' from the *'Processor'* tree and drop it into the *'Hardware Configuration'* design window. This block is referred to in this document, as the 'IC'.

6. Drag 'USBi' from the *'Communication Channels'* tree and drop it into the *'Hardware Configuration'* design window. If the USBi driver is installed properly and the USBi is connected to the PC, the word "USB" is in a green box. If there is a problem with driver installation or connection, the box with text "USB" is shown in red. In such a case, remove

the USBi from the *'Hardware Configuration'* design window, verify the USB connection, and repeat this step.

7. Connect the first blue (output) pin of the 'USBi' to the green (input) pin of the 'IC 1' block as shown in Figure 25.

8. A drop-down menu is enabled adjacent to the connected pin of 'USBi'. Select 'SPI 0x1 ADR0' from the drop-down menu for both the ICs.

9. If "Aardvark Host I^{2}C/SPI Host Adapter" is being used, drag 'Aardvark' from the *'Communication Channels'* tree and drop it into the *'Hardware Configuration'* design window. If the Aardvark driver is installed properly and the Aardvark is connected to the PC, the word "USB" is in a green box. If there is a problem with driver installation or connection, the box with text "USB" is shown in red. In such a case, remove the 'Aardvark' from the *'Hardware Configuration'* design window, verify the USB connection, and repeat this step.

10. Connect the first blue (output) pin of the 'Aardvark to the green (input) pin of the 'IC 1' block as shown in Figure 26.

11. A drop-down menu is enabled adjacent to the connected pin of 'Aardvark. Select 'SPI 0x1 (1)' from the drop-down menu for both the ICs.

12. An IC control window is created for the IC1 dragged as part of step 0. Select "Core 1" in the "Default SHARC Core" section of the "Main" tab of IC control window for "IC 1". Leave the other settings unchanged.

13. Go to the *'Block Schematic'* tab. The toolbox on the left side lists the Modules available for the SHARC DSP.

14. Drag and drop the desired Modules in the Schematic workspace and interconnect the Modules to form a DSP design.

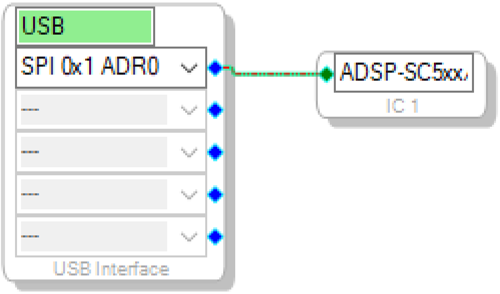


Figure 25: USBi to ADSP-2156x connection

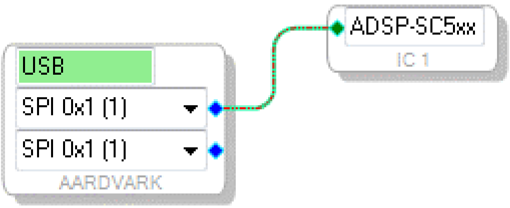


Figure 26: Aardvark to 2156x connection

6.4.3 Running the New Schematic

1. Refer to section 6.4.1 and 6.4.2 for creating a schematic for ADSP-SC5xx and ADSP-215xx.
2. Refer to section 6.3 for rebuilding the Application (if required) and booting the Target.
3. Select the rebuilt/prebuilt Application DXE's for both the SHARC's by clicking on *'Select Application DXE'* in SHARC0/SHARC1 tab of SigmaStudio IC control window.
4. Connect the audio inputs to the EZ-BOARD and play audio on PC.
5. Press Link-Compile-Download.
6. The output audio will be heard through speakers/headphones connected to the EZ-BOARD.

7 DemoUc application

The DemoUc application demonstrates the usage of ARM core of ADSP-SC5xx as microcontroller in the final deployment mode of the application developed using SigmaStudio. The DemoUc applications for ADSP-SC573 can be found in *“< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\DemoUc\ADSP-SC57x*[ii]\”, those of ADSP-SC584 and ADSP-SC589 can be found in *“< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\ Examples\DemoUc\ADSP-SC58x*[iii]\”, for ADSP-SC594 can be found in “< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\DemoUc\ADSP-SC59x[iv]\”, for ADSP-21593 can be found in “< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\DemoUc\ADSP-2159x[v]\”, and the example for ADSP-21569 can be found in in *“< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\ Examples\DemoUc\ADSP-2156x*[vi]*”*

The DemoUc application for ADSP-SC5xx provided with the package uses the code and parameters exported for the “Volume_Mute_Block_SC5xx.dspproj” schematic located in *“< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Host\Examples\Sample Schematics\ADSP-SC5xx\”* .This application configures push buttons PB1 and PB2 on the EZ-Board which can be used to alter certain parameters of the “Volume_Mute_Block_SC5xx.dspproj” schematic. PB1 is used to change the “Enable/Disable” parameter of the “Master Mute” cell of “Volume_Mute_Block_SC5xx.dspproj” schematic running on SHARC Core 1. PB2 is used to change the “Enable/Disable” parameter of the “General Filter” cell of “Volume_Mute_Block_SC5xx.dspproj” schematic running on SHARC Core 2. Similar mechanisms may be used to tune the parameters of a custom schematic in the final deployment mode using ARM core as microcontroller.

The DemoUc application for ADSP-2156x uses the code and parameters exported for “Volume_Mute_Block_2156x.dspproj” schematic located in *“< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Host\Examples\Sample Schematics\ADSP-2156x\”.*

The sections below describe the steps for running this example application and updating this application for a custom schematic.

7.1 Running the default DemoUc application

7.1.1 DemoUc application for ADSP-SC5xx

Follow the steps below for running the default DemoUc application with code and parameters from “Volume_Mute_Block_SC5xx.dspproj”.

1. Launch CrossCore Embedded Studio on the SigmaStudio Host PC. Specify the workspace when prompted.

2. Select File→Import. Import *SS_uC_App_Core0*, *SS_uC_App_Core1* and *SS_uC_App_Core2* projects of appropriate target located in subfolders inside “< *Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\Target\ Examples\DemoUc\”* into CrossCore Embedded Studio.

3. Uncheck “Build (if required) before launching” checkbox in Window->Preferences->Run/Debug->Launching.

4. Select Run→Debug Configurations. Create a new Debug configuration under ‘Application with CrossCore Debugger’. Select appropriate processor, connection type and platform. Click finish.

5. Load the prebuilt DXE’s SS_uC_App_Core0‘on Core 0, ‘SS_uC_App_Core1.dxe’ on Core 1 and ‘SS_uC_App_Core2.dxe’ on Core 2. Prebuilt DXEs of each project can be found inside the respective ‘Release’ folders of each of the projects.

6. Connect the audio inputs to the EZ-BOARD and play audio on PC.

7. Follow steps 7 and 8 of section 6.1 for setting the options of the “Debug Configurations” and running the application.

8. The output audio will be heard through speakers/headphones connected to the EZ-BOARD.

7.1.2 DemoUc application for ADSP-2156x

Follow the steps below for running the default DemoUc application with code and parameters from “Volume_Mute_Block_2156x.dspproj”.

1. Launch CrossCore Embedded Studio on the SigmaStudio Host PC. Specify the workspace when prompted.

2. Select File→Import. Import *SS_uC_App_Core1* project located in folder “*< Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\DemoUc\ ADSP-2156x”* into CrossCore Embedded Studio.

3. Uncheck “Build (if required) before launching” checkbox in Window->Preferences->Run/Debug->Launching.

4. Select Run→Debug Configurations. Create a new Debug configuration under ‘Application with CrossCore Debugger’. Select appropriate processor, connection type and platform. Click finish.

5. Load the prebuilt ‘SS_uC_App_Core1.dxe’ on Core 1. The Prebuilt DXE of the project can be found inside the respective ‘Release’ folder of ADSP-2156x.

6. Connect the audio inputs to the EZ-BOARD and play audio on PC.

7. Follow steps 7 and 8 of section 6.1 for setting the options of the “Debug Configurations” and running the application.

8. The output audio will be heard through speakers/headphones connected to the EZ-BOARD.

Note: *The ADSP-21569 example demoUc application has support for EV-21569-SOM with EV-SOM-CRR EZ-Kit platform by default. For running the application on ADSP-21569 EZ-Kit platform, the "ADSP_21569_SOM" pre-processor macro has to be removed from project compiler options and the application needs to be rebuilt.*

PB2 is used to change the "Enable/Disable" parameter of the "General Filter" cell on ADSP-21569 EZ-Kit won't work since the same GPIO pin used for LED.

The schematic source files need to be regenerated for using DemoUc application in "debug" build configuration. Use the built "debug" version of generated DXE's in SigmaStudio schematic. Refer section A for generating schematic source files using "export system files" of SigmaStudio schematic.

The SigmaStudio schematic tuning support added for example DemoUc applications. The SigmaStudio schematic tuning not supported for ADSP-SC573 DemoUc application since push button and SPI1 shares same GPIO line.

7.2 Running the DemoUc application with a custom schematic

Follow the steps below for running the DemoUc application with code and parameters exported from a custom schematic.

7.2.1 Custom schematic on ADSP-SC5xx/ADSP-2157x/ADSP-2158x/ADSP-2159x

1. Open SigmaStudio

2. Launch the Add-Ins browser by clicking Add-Ins browser option in Tools menu.

3. Check if the latest version of SigmaStudioForSHARC (ADSP-SC5xx/ADSP-215xx) DLLs, namely, SharcPubLib.dll, SharcModules.dll, SharcDesigner.dll are selected. If not selected, select the latest version and disable the older versions. Save and close the Add-Ins browser and re-launch SigmaStudio.

4. Open the schematic which should be executed in the deployment mode using ARM core as Microcontroller.

5. Select the rebuilt/prebuilt Application DXE's of appropriate target from "< *Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\DemoUc\"* for the SHARC cores by clicking on 'Select Application DXE' in SHARC0/SHARC1 tab of SigmaStudio IC control window.

6. Click on “Link Compile Download” and do export the system files as detailed in section “System Files Export” of **[4]**. Ensure that the “Export XML only” option is unchecked.

7. Copy all the exported files into “< *Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\ Target\utilities\exportCodeParam”* folder.

8. Run the “*exportCodeParam”* utility as detailed in Annexure A for all the ICs in the schematic.

9. Copy the generated ‘.c’ and ‘.h’ files into “< *Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\ Target\Examples\DemoUc\ADSP-SC58x\Source”* folder or “< *Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\ Target\ Examples\DemoUc\ADSP-SC57x\Source” or “< Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\ Target\ Examples\DemoUc\ADSP-SC59x\Source”* folder*.*

10. Launch CrossCore Embedded Studio on the SigmaStudio Host PC. Specify the workspace when prompted.

11. Select File→Import. Import *SS_uC_App_Core0*, *SS_uC_App_Core1* and *SS_uC_App_Core2* projects of appropriate target located in subfolders inside “< *Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\ Target\ Examples\DemoUc\”* into CrossCore Embedded Studio.

12. Remove the ‘.c’ files related to default .dspproj schematic from the ARM core application and add the new ‘.c’ files generated for the custom schematic.

13. Open file ‘adi_ss_uc_app_arm.c’ and include the generated ‘.h’ file in place of the existing ‘.h’ files in line 25 and 26.

14. Ensure that the names of the SMAP, Code and parameter arrays in function ‘PopulateSsnConfig()’ are in accordance with the arrays within the newly created ‘.c’ file.

15. Rebuild the application for ARM core

16. Select Run→Debug Configurations. Create a new Debug configuration under ‘Application with CrossCore Debugger’. Select appropriate processor, connection type and platform. Click finish.

17. Load the DXEs SS_uC_App_Core0‘on Core 0, ‘SS_uC_App_Core1.dxe’ on Core 1 and ‘SS_uC_App_Core2.dxe’ on Core 2. Application DXEs can be found inside the respective ‘Release’ or ‘Debug’ folders of each of the projects based on the chosen active configuration in CCES.

18. Connect the audio inputs to the EZ-BOARD and play audio on PC.

19. Follow steps 7 and 8 of section 6.1for setting the options of the “Debug Configurations” and running the application.

20. The output audio will be heard through speakers/headphones connected to the EZ-BOARD as per the connections in the custom schematic.

Note: The example microcontroller application provided with the package only supports single instance running on each of the SHARC cores. The application source files are required to be modified for executing multi-instance schematics.

7.2.2 Custom schematic for ADSP-2156x

1. Open SigmaStudio

2. Launch the Add-Ins browser by clicking Add-Ins browser option in Tools menu.

3. Check if the latest version of SigmaStudioForSHARC(ADSP-SC5xx/ADSP-215xx) DLLs, namely, SharcPubLib.dll, SharcModules.dll, SharcDesigner.dll are selected. If not selected, select the latest version and disable the older versions. Save and close the Add-Ins browser and re-launch SigmaStudio.

4. Open the schematic which should be executed in the deployment mode using SHARC core as Microcontroller.

5. Select the rebuilt/prebuilt Application DXE’s of ADSP-2156x target from “< *Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\DemoUc\ADSP-2156x”* for the SHARC core by clicking on ‘Select Application DXE’ in SHARC0 tab of SigmaStudio IC control window.

6. Export the system files as detailed in section “System Files Export” of **[4]**. Ensure that the “Export XML only” option is unchecked.

7. Copy all the exported files into “< *Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\ Target\utilities\exportCodeParam”* folder.

8. Run the “*exportCodeParam”* utility as detailed in Annexure A for all the ICs in the schematic.

9. Copy the generated ‘.c’ and ‘.h’ files into < *Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\ Target\ Examples\DemoUc\ADSP-2156x\Source”*.

10. Launch CrossCore Embedded Studio on the SigmaStudio Host PC. Specify the workspace when prompted.

11. Select File→Import. Import *SS_uC_App_Core1* project located inside "< *Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\ Target\ Examples\DemoUc\ADSP-2156x"* into CrossCore Embedded Studio.

12. Remove the '.c' files related to default dspproj schematic from the SHARC core application and add the new '.c' files generated for the custom schematic.

13. Open file 'adi_ss_uc_app_sh0.c' and include the generated '.h' file in place of the existing '.h' files in line 24.

14. Ensure that the names of the SMAP, Code and parameter arrays in function 'PopulateSsnConfig()' are in accordance with the arrays within the newly created '.c' file.

15. Rebuild the application for SHARC core

16. Select Run→Debug Configurations. Create a new Debug configuration under 'Application with CrossCore Debugger'. Select appropriate processor, connection type and platform. Click finish.

17. Load the DXEs 'SS_uC_App_Core1.dxe' on Core 1. The application DXE can be found inside the respective 'Release' or 'Debug' folders of each of the projects based on the chosen active configuration in CCES.

18. Connect the audio inputs to the EZ-BOARD and play audio on PC.

19. Follow steps 7 and 8 of section 6.1for setting the options of the "Debug Configurations" and running the application.

20. The output audio will be heard through speakers/headphones connected to the EZ-BOARD as per the connections in the custom schematic.

***Note:** The ADSP-21569 example demoUc application has support for EV-21569-SOM with EV-SOM-CRR EZ-Kit platform by default. For running the application on ADSP-21569 EZ-Kit platform, the "ADSP_21569_SOM" pre-processor macro has to be removed from project compiler options and the application needs to be rebuilt.*

8 Library Integration with Audio Talk-Through Example

This section demonstrates the use of SigmaStudio Communication and SSN libraries with the Audio talk-through example present in the EZ-BOARD BSP.

8.1 Running the library integration example for ADSP-SC584/ADPS-SC594

Follow the below steps to run the default ADSP-SC584/ADSP-SC594 example application:

1. Launch CrossCore Embedded Studio on the SigmaStudio Host PC. Specify the workspace when prompted.

2. Select File→Import. Import LibIntegrationExample_*Core0*, *LibIntegrationExample_Core1* and *LibIntegrationExample_Core2* projects of appropriate targets located in subfolders inside “*< Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\Target\ Examples\LibraryIntegration\ ADSP-SC584\”* or “*< Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\Target\ Examples\LibraryIntegration\ ADSP-SC594\”* into CrossCore Embedded Studio.

3. Uncheck “Build (if required) before launching” checkbox in Window->Preferences->Run/Debug->Launching.

4. Select Run→Debug Configurations. Create a new Debug configuration under ‘Application with CrossCore Debugger’. Select appropriate processor, connection type and platform. Click finish.

5. Load the prebuilt DXE’- LibIntegrationExample_Core0 ‘on Core 0, ‘LibIntegrationExample_Core1.dxe’ on Core 1 and ‘LibIntegrationExample_Core2.dxe’ on Core 2. The DXEs of each project can be found inside the respective ‘Release’ folders of each of the projects.

6. Follow steps 7 and 8 of section 6.1 for setting the options of the “Debug Configurations” and running the application.

7. Create a new SigmaStudio schematic.

8. Drag and drop the ADSP-SC5xx/215xx IC from the Processors into the hardware configuration tab. Drag the USBi from the Communication Channels and connect it to the ADSP-SC5xx/215xx IC.

9. From under IC1 in the Tree Toolbox, drag and drop the Signal Input and Output modules into the Block schematic tab and connect the first two channels of the Signal Input to the output.

10. In the Hardware Configuration Tab -> ADSP-SC5xx/215xx Control click on Select Application DXE and load the LibIntegrationExample_Core1.dxe.

11. Click on the Link, Compile and Download button in the SigmaStudio menu bar.

12. Connect the audio inputs to the EZ-BOARD audio input channels 0 and 1 and play audio on PC.

13. The output audio will be heard through speakers/headphones connected to the EZ-BOARD audio outputs channels 0 and 1.

8.2 Running the library integration example for ADSP-21573/ADSP-21593

Follow the below steps to run the default ADSP-21573/ADSP-21593 example application:

1. Launch CrossCore Embedded Studio on the SigmaStudio Host PC. Specify the workspace when prompted.

2. Select File→Import *LibIntegrationExample_Core1* and *LibIntegrationExample_Core2* projects of appropriate targets located in subfolders inside “< *Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LibraryIntegration\ADSP-21573\”* or “< *Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LibraryIntegration\ADSP-21593\”* into CrossCore Embedded Studio.

3. Uncheck “Build (if required) before launching” checkbox in Window->Preferences->Run/Debug->Launching.

4. Select Run→Debug Configurations. Create a new Debug configuration under ‘Application with CrossCore Debugger’. Select appropriate processor, connection type and platform. Click finish.

5. Load the prebuilt DXE’- ‘LibIntegrationExample_Core1.dxe’ on Core 1 and ‘LibIntegrationExample_Core2.dxe’ on Core 2. The DXEs of each project can be found inside the respective ‘Release’ folders of each of the projects.

6. Follow steps 7 and 8 of section 6.1 for setting the options of the “Debug Configurations” and running the application.

7. Create a new SigmaStudio schematic.

8. Drag and drop the ADSP-SC5xx/215xx IC from the Processors into the hardware configuration tab. Drag the USBi from the Communication Channels and connect it to the ADSP-SC5xx/215xx IC.

9. From under IC1 in the Tree Toolbox, drag and drop the Signal Input and Output modules into the Block schematic tab and connect the first two channels of the Signal Input to the output.

10. For ADSP-21573, In the Hardware Configuration tab, click on Tab->IC2-ADSP-SC5xx control, set the core as ‘Core2’ in the main window and in the SHARC1 tab control click on Select Application DXE and load the LibIntegrationExample_Core2.dxe.

11. For ADSP-21593, In the Hardware Configuration Tab -> ADSP-SC5xx/215xx Control click on Select Application DXE and load the LibIntegrationExample_Core1.dxe.

12. Click on the Link, Compile and Download button in the SigmaStudio menu bar.

13. Connect the audio inputs to the EZ-BOARD audio input channels 0 and 1 and play audio on PC.

14. The output audio will be heard through speakers/headphones connected to the EZ-BOARD audio outputs channels 0 and 1.

8.3 Running the library integration example for ADSP-21569

Follow the below steps to run the default ADSP-21569 example application:

1. Launch CrossCore Embedded Studio on the SigmaStudio Host PC. Specify the workspace when prompted.

2. Select File→Import *LibIntegration* project present in “< *Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\Target\Examples\LibraryIntegration\ADSP-21569\”* into CrossCore Embedded Studio.

3. Uncheck “Build (if required) before launching” checkbox in Window->Preferences->Run/Debug->Launching.

4. Select Run→Debug Configurations. Create a new Debug configuration under ‘Application with CrossCore Debugger’. Select appropriate processor, connection type and platform. Click finish.

5. Load the prebuilt DXE’- ‘LibIntegration.dxe’ on Core 1. The DXEs of the project can be found inside the respective ‘Release’ folders of each of the projects.

6. Follow steps 7 and 8 of section 6.1 for setting the options of the “Debug Configurations” and running the application.

7. Create a new SigmaStudio schematic.

8. Drag and drop the ADSP-SC5xx/215xx IC from the Processors into the hardware configuration tab. Drag the USBi from the Communication Channels and connect it to the ADSP-SC5xx/215xx IC.

9. From under IC1 in the Tree Toolbox, drag and drop the Signal Input and Output modules into the Block schematic tab and connect the first two channels of the Signal Input to the output.

10. In the Hardware Configuration Tab -> IC1- ADSP-SC5xx/215xx Control click on Select Application DXE and load the LibIntegration.dxe.

11. Click on the Link, Compile and Download button in the SigmaStudio menu bar.

12. Connect the audio inputs to the EZ-BOARD audio input channels 0 and 1 and play audio on PC.

13. The output audio will be heard through speakers/headphones connected to the EZ-BOARD audio outputs channels 0 and 1.

Note: *The ADSP-21569 example* library integration *application has support for EV-21569-SOM with EV-SOM-CRR EZ-Kit platform by default. For running the application on ADSP-21569 EZ-Kit platform, the “ADSP_21569_SOM” pre-processor macro has to be removed from project compiler options and the application needs to be rebuilt.*

9 Debugging Schematics using CCES

This section explains how to debug a schematic using CrossCore Embedded Studio.

9.1 Steps to be followed to Debug a function in the schematic

Follow the instructions given below to set up the CCES for debugging the schematic.

1. Establish the hardware setup as described in Section 3.1.
2. Install all software and tools mentioned in Section 2.
3. The Default Application for ADSP-SC573, ADSP-SC584, ADSP-SC589 and ADSP-SC594 are supplied with the package. The projects can be found inside *"< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\ Examples\Demo\*". Launch CrossCore Embedded Studio on the SigmaStudio Host PC. Specify the workspace when prompted. Select File→Import. Import *SS_App_Core0*, *SS_App_Core1* and *SS_App_Core2* projects of appropriate target into CrossCore Embedded Studio.
4. Select all the 3 core applications and build in Debug Mode.

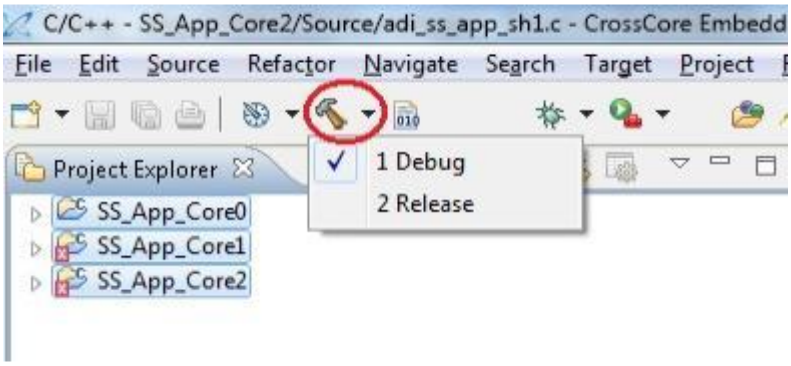
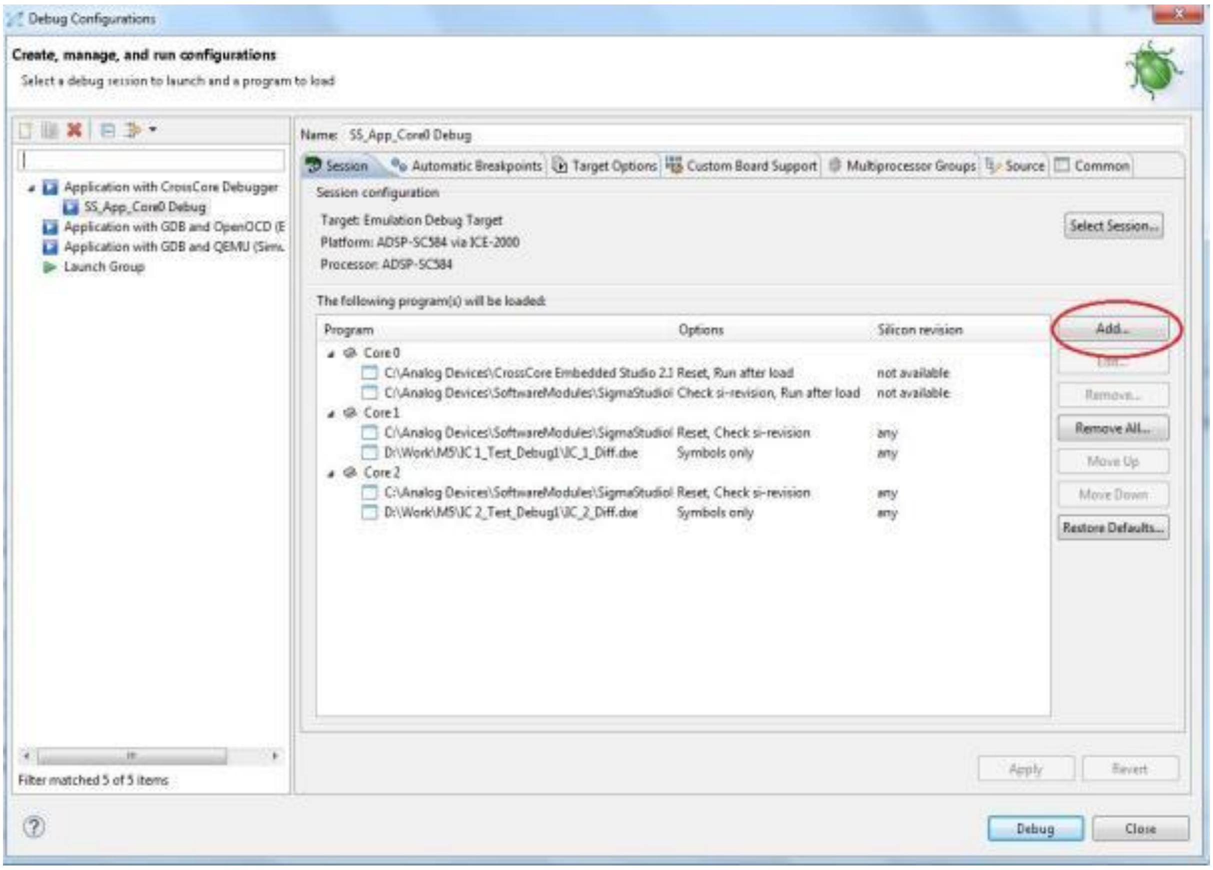

C/C++ - SS_App_Core2/Source/adi_ss_app_sh1.c - CrossCore Embedd
File Edit Source Refactor Navigate Search Target Project
Project Explorer
SS_App_Core0
SS_App_Core1
SS_App_Core2
1 Debug
2 Release

Figure 27: Build the application

5. Uncheck "Build (if required) before launching" checkbox in Window->Preferences->Run/Debug->Launching.
6. Launch SigmaStudio.
7. Open any desired schematic that is required to be debugged.
8. Make sure that the custom module going to be debugged was created with library file built in debug configuration.
9. Select the rebuilt Application DXE's for both the SHARC's by clicking on *'Select Application DXE'* in SHARC0/SHARC1 tab of SigmaStudio IC control window.

10. Press Link-Compile-Download. This is a dummy download of the schematic for generating the schematic DXEs.
11. Open the IC 1_<Schematic Name> and IC 2_<Schematic Name> folders to find the IC_1_Diff.dxe and IC_2_Diff.dxe files. The symbols present in these DXEs are loaded on top of the target application DXEs as explained in further steps.
12. Select Run→Debug Configurations in CCES. Create a new Debug configuration under *'Application with CrossCore Debugger'*. Select appropriate processor, connection type and platform. Click finish.
13. Load the DXE's 'SS_App_Core0' on Core 0, '*SS_App_Core1.dxe*' on Core 1 and '*SS_App_Core2.dxe*' on Core 2. Now select the '*SS_App_Core1.dxe*' in Core 1, press 'Add' and select the schematic DXE corresponding to Core 1 (IC_1_Diff.dxe), select *SS_App_Core2.dxe*' in Core 2, press 'Add' and select the schematic DXE corresponding to Core 2 (IC_2_Diff.dxe) as shown in Figure 28.



Debug Configurations

Create, manage, and run configurations

Select a debug session to launch and a program to load

Name: SS_App_Core0 Debug

Session configuration

Target: Emulation Debug Target

Platform: ADSP-SC584 via ICE-2000

Processor: ADSP-SC584

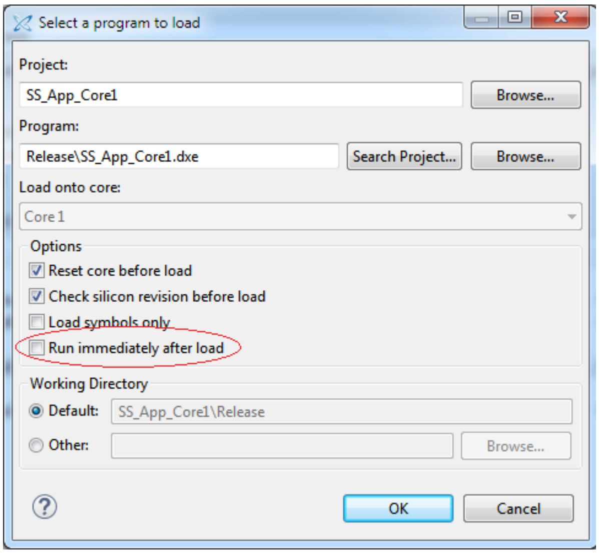
The following program(s) will be loaded:

Program	Options	Silicon revision
Core 0		
C:\Analog Devices\CrossCore Embedded Studio 2.1	Reset, Run after load	not available
C:\Analog Devices\SoftwareModules\SigmaStudioI	Check si-revision, Run after load	not available
Core 1		
C:\Analog Devices\SoftwareModules\SigmaStudioI	Reset, Check si-revision	any
D:\Work\M5\IC 1_Test_Debug1\IC_1_Diff.dxe	Symbols only	any
Core 2		
C:\Analog Devices\SoftwareModules\SigmaStudioI	Reset, Check si-revision	any
D:\Work\M5\IC 2_Test_Debug1\IC_2_Diff.dxe	Symbols only	any

Filter matched 5 of 5 items

Figure 28: Debug Configuration

14. Double click on ‘*SS_App_Core1*.dxe’ of Core 1 and uncheck *‘Run immediately after load’* option as shown in the Figure 29. Repeat the same for ‘*SS_App_Core2.dxe*’ for Core 2.



Select a program to load

Project:
SS_App_Core1

Program:
Release\SS_App_Core1.dxe

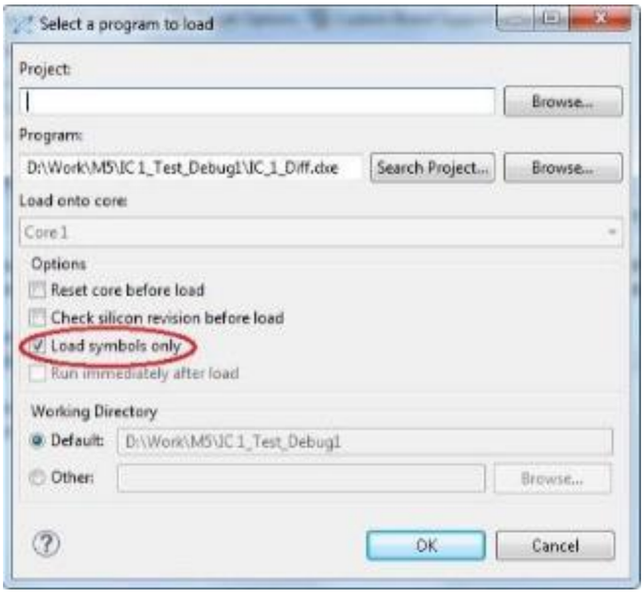
Load onto core:
Core 1

Options
☑ Reset core before load
☑ Check silicon revision before load
□ Load symbols only
□ Run immediately after load

Working Directory
● Default: SS_App_Core1\Release
○ Other:

Figure 29: Select a program to load window

15. Double click on ‘*IC_1_Diff.dxe*’ of Core 1 and uncheck everything except *‘Load Symbols only’ after load’* option as shown in the Figure 30. Repeat the same for ‘*IC_2_Diff.dxe*’ for Core 2.



Select a program to load

Project:

Program:
D:\Work\M5\IC1_Test_Debug1\IC_1_Diff.dxe

Load onto core:
Core 1

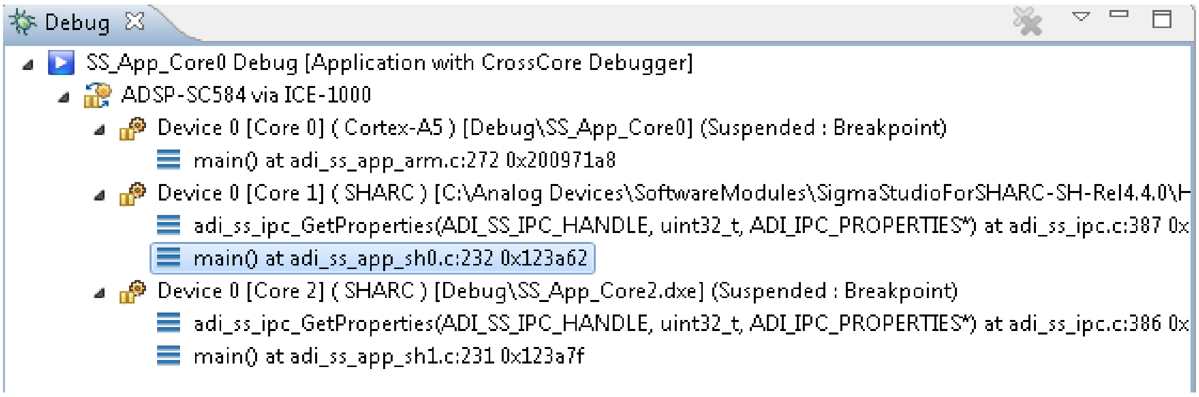
Options
□ Reset core before load
□ Check silicon revision before load
☑ Load symbols only
□ Run immediately after load

Working Directory
● Default: D:\Work\M5\IC1_Test_Debug1
○ Other:

Figure 30: Load the schematic DXE

16. Click on “MP Resume” to run Core 0 (ARM). This enables the Core 1 and 2 (SHARC).

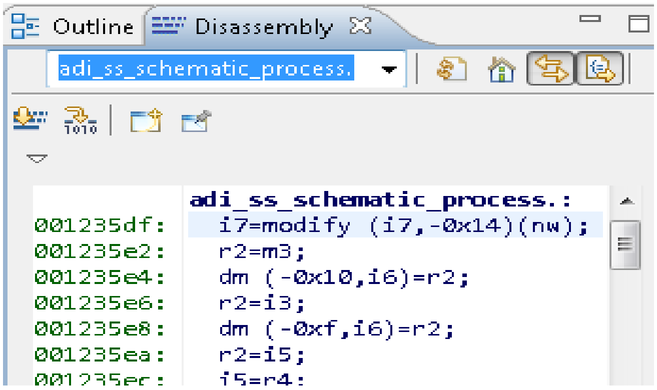
17. Select main () function in Debugger window of corresponding SHARC core, the SigmaStudio schematic module to be debugged.



```
Debug
  SS_App_Core0 Debug [Application with CrossCore Debugger]
    ADSP-SC584 via ICE-1000
      Device 0 [Core 0] ( Cortex-A5 ) [Debug\SS_App_Core0] (Suspended : Breakpoint)
        main() at adi_ss_app_arm.c:272 0x200971a8
      Device 0 [Core 1] ( SHARC ) [C:\Analog Devices\SoftwareModules\SigmaStudioForSHARC-SH-Rel4.4.0\H
        adi_ss_ipc_GetProperties(ADI_SS_IPC_HANDLE, uint32_t, ADI_IPC_PROPERTIES*) at adi_ss_ipc.c:387 0x
        main() at adi_ss_app_sh0.c:232 0x123a62
      Device 0 [Core 2] ( SHARC ) [Debug\SS_App_Core2.dxe] (Suspended : Breakpoint)
        adi_ss_ipc_GetProperties(ADI_SS_IPC_HANDLE, uint32_t, ADI_IPC_PROPERTIES*) at adi_ss_ipc.c:386 0x
        main() at adi_ss_app_sh1.c:231 0x123a7f
```

Figure 31: Select main() function of SHARC Core

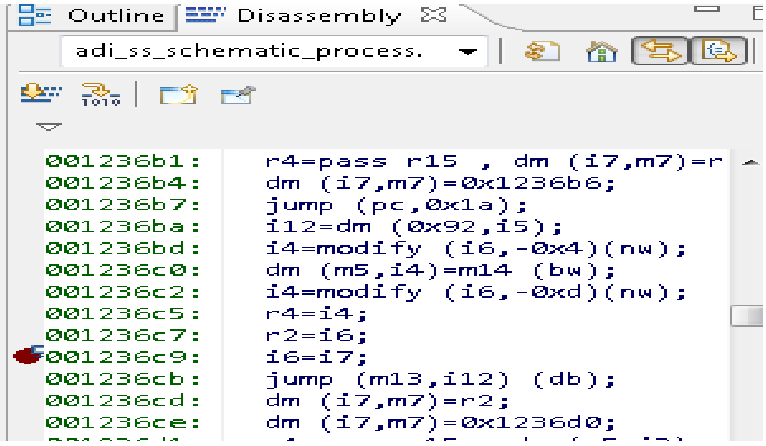
18. Enter **“*adi_ss_schematic_process.*”** symbol name in the Disassembly window and press enter to get Disassembly function.



```
              adi_ss_schematic_process.:
001235df:       i7=modify (i7,-0x14)(nw);
001235e2:       r2=m3;
001235e4:       dm (-0x10,i6)=r2;
001235e6:       r2=i3;
001235e8:       dm (-0xf,i6)=r2;
001235ea:       r2=i5;
001235ec:       i5=r4;
```

Figure 32: Disassembly Find Symbol “adi_ss_schematic_process.”

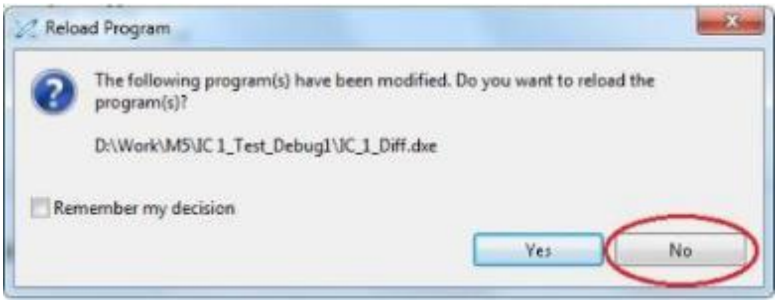
19. Scroll down in the Disassembly function and set breakpoint in the instruction shown in Figure 33. There will be a pop up when breakpoint set, just select “Don’t show me again” and press “OK”.



```
001236b1:   r4=pass r15 , dm (i7,m7)=r
001236b4:   dm (i7,m7)=0x1236b6;
001236b7:   jump (pc,0x1a);
001236ba:   i12=dm (0x92,i5);
001236bd:   i4=modify (i6,-0x4)(nw);
001236c0:   dm (m5,i4)=m14 (bw);
001236c2:   i4=modify (i6,-0xd)(nw);
001236c5:   r4=i4;
001236c7:   r2=i6;
001236c9:   i6=i7;
001236cb:   jump (m13,i12) (db);
001236cd:   dm (i7,m7)=r2;
001236ce:   dm (i7,m7)=0x1236d0;
```

Figure 33: Set Breakpoint in Disassembly Function

20. The steps 17 to 19 need to be followed for SHARC core2 if any module to be debugged on SHARC core2.
21. Then individually select and run Core 1 and Core 2 by clicking on “Resume” as shown in Figure 16.
22. Link-Compile-Download the required schematic again in SigmaStudio.
23. A pop-up appears in CCES prompting the user to re-load the schematic DXEs. Select No.


Reload Program

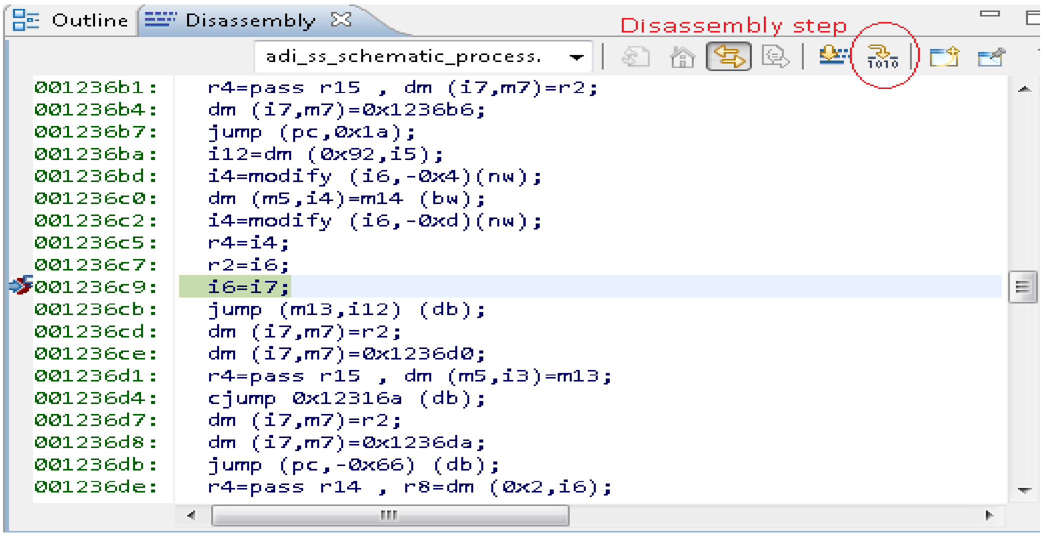
The following program(s) have been modified. Do you want to reload the program(s)?

D:\Work\M5\IC 1_Test_Debug1\IC_1_Diff.dxe

□ Remember my decision

Figure 34: Schematic DXE re-load prompt window

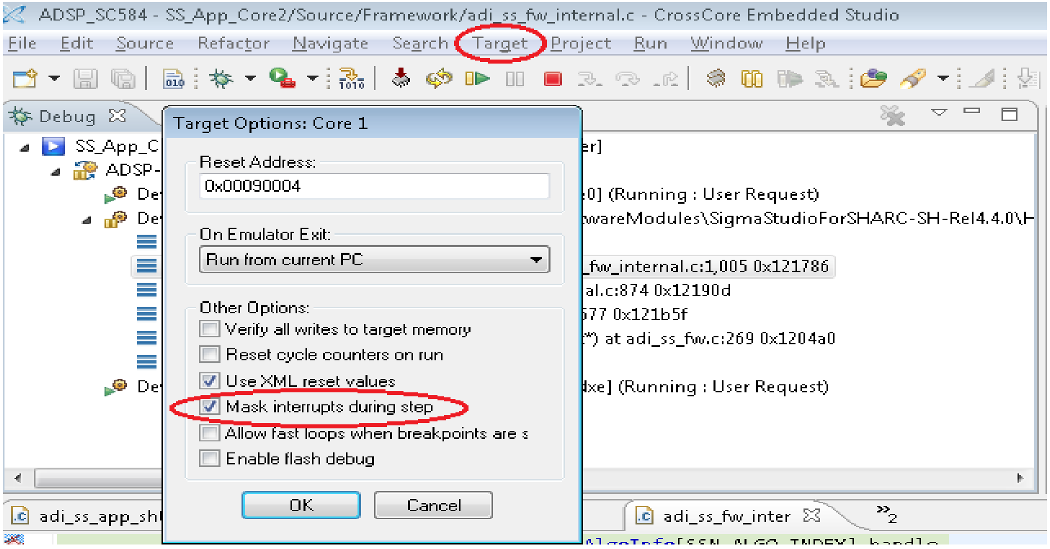
24. Observe that the target halts at the breakpoint set as shown in Figure 35.



```
001236b1:   r4=pass r15 , dm (i7,m7)=r2;
001236b4:   dm (i7,m7)=0x1236b6;
001236b7:   jump (pc,0x1a);
001236ba:   i12=dm (0x92,i5);
001236bd:   i4=modify (i6,-0x4)(nw);
001236c0:   dm (m5,i4)=m14 (bw);
001236c2:   i4=modify (i6,-0xd)(nw);
001236c5:   r4=i4;
001236c7:   r2=i6;
001236c9:   i6=i7;
001236cb:   jump (m13,i12) (db);
001236cd:   dm (i7,m7)=r2;
001236ce:   dm (i7,m7)=0x1236d0;
001236d1:   r4=pass r15 , dm (m5,i3)=m13;
001236d4:   cjump 0x12316a (db);
001236d7:   dm (i7,m7)=r2;
001236d8:   dm (i7,m7)=0x1236da;
001236db:   jump (pc,-0x66) (db);
001236de:   r4=pass r14 , r8=dm (0x2,i6);
```

Figure 35: Core Halts at Breakpoint

25. Select “Mask interrupts during step” in target option and press “OK” as shown inFigure 36.


ADSP_SC584 - SS_App_Core2/Source/Framework/adi_ss_fw_internal.c - CrossCore Embedded Studio
File
Edit
Source
Refactor
Navigate
Search
Target
Project
Run
Window
Help
Debug
SS_App_C
ADSP-
De
De
De
er]
0] (Running : User Request)
wareModules\SigmaStudioForSHARC-SH-Rel4.4.0\
_fw_internal.c:1,005 0x121786
al.c:874 0x12190d
577 0x121b5f
*) at adi_ss_fw.c:269 0x1204a0
xe] (Running : User Request)
adi_ss_app_sh
adi_ss_fw_inter

Target Options: Core 1

Reset Address:
0x00090004

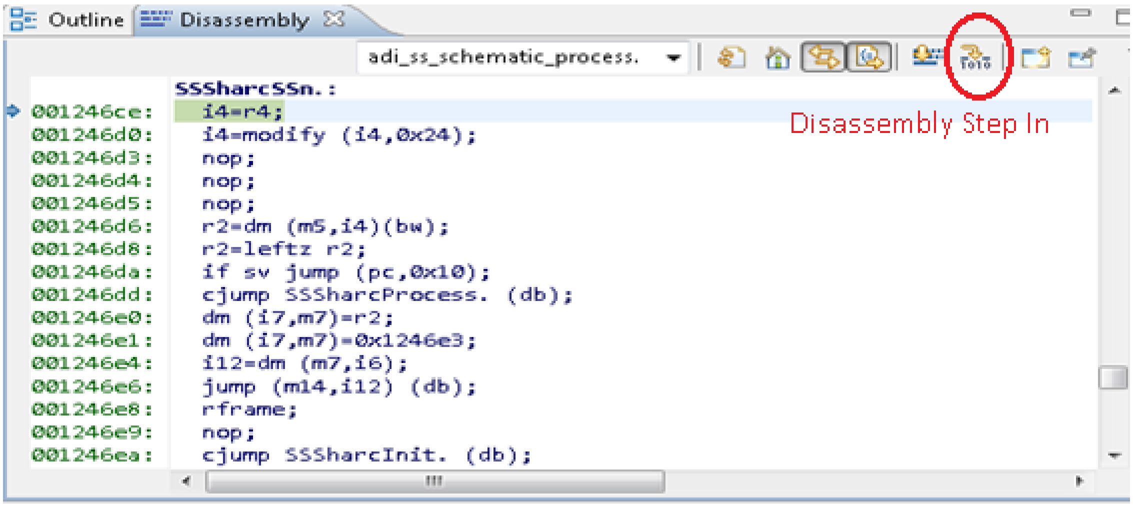
On Emulator Exit:
Run from current PC

Other Options:
☐ Verify all writes to target memory
☐ Reset cycle counters on run
☑ Use XML reset values
☑ Mask interrupts during step
☐ Allow fast loops when breakpoints are s
☐ Enable flash debug

OK Cancel

Figure 36: Mask interrupt during step selection

26. Step-in 2 times using disassembly step in (shown in Figure 37) to enter ***“SSSharcSSn.”*** disassembly function.

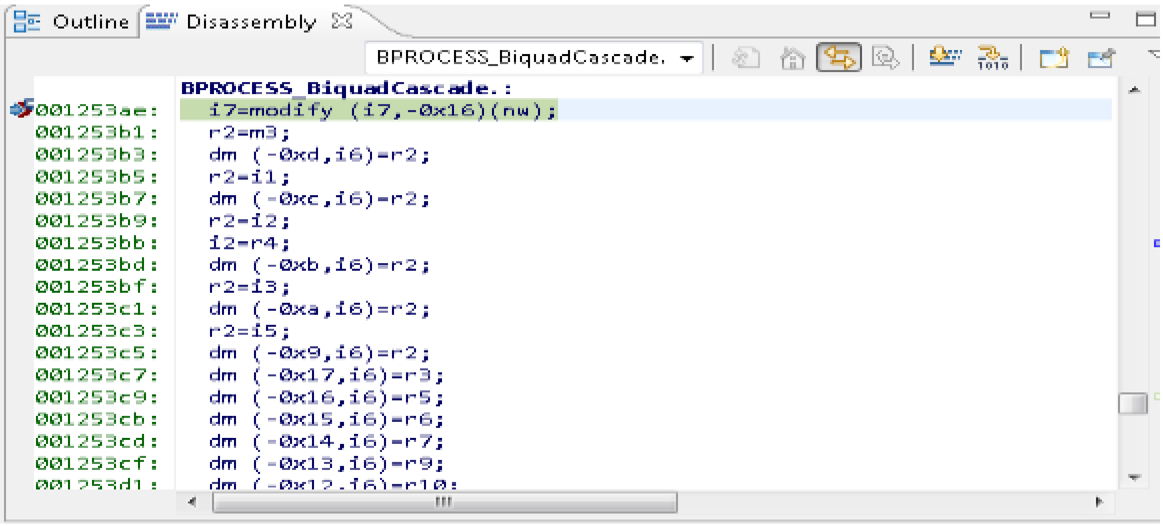


```
                SSSharcSSn.:
001246ce:         i4=r4;
001246d0:         i4=modify (i4,0x24);
001246d3:         nop;
001246d4:         nop;
001246d5:         nop;
001246d6:         r2=dm (m5,i4)(bw);
001246d8:         r2=leftz r2;
001246da:         if sv jump (pc,0x10);
001246dd:         cjump SSSharcProcess. (db);
001246e0:         dm (i7,m7)=r2;
001246e1:         dm (i7,m7)=0x1246e3;
001246e4:         i12=dm (m7,i6);
001246e6:         jump (m14,i12) (db);
001246e8:         rframe;
001246e9:         nop;
001246ea:         cjump SSSharcInit. (db);
```

Figure 37: Disassembly Step In to “SSSharcSSn.”

27. The source files ***(.c)*** of custom module can be dragged to CCES source window and set breakpoint on the source file module functions directly. Step-in through the code using debugger window step in options and debug the module function.

28. The other way is, enter the required symbol name of module function (byte addressed symbol followed by a ‘.’) which is to be debugged in the Disassembly window and set a breakpoint at the required instruction. For example, debugging Biquad Cascade module enter ***“BPROCESS_BiquadCascade.”*** symbol and set a breakpoint as shown inFigure 38. Run the core by clicking on “Resume”. The core halts at module function where the breakpoint is set.



```
                BPROCESS_BiquadCascade.:
001253ae:         i7=modify (i7,-0x16)(nw);
001253b1:         r2=m3;
001253b3:         dm (-0xd,i6)=r2;
001253b5:         r2=i1;
001253b7:         dm (-0xc,i6)=r2;
001253b9:         r2=i2;
001253bb:         i2=r4;
001253bd:         dm (-0xb,i6)=r2;
001253bf:         r2=i3;
001253c1:         dm (-0xa,i6)=r2;
001253c3:         r2=i5;
001253c5:         dm (-0x9,i6)=r2;
001253c7:         dm (-0x17,i6)=r3;
001253c9:         dm (-0x16,i6)=r5;
001253cb:         dm (-0x15,i6)=r6;
001253cd:         dm (-0x14,i6)=r7;
001253cf:         dm (-0x13,i6)=r9;
001253d1:         dm (-0x12,i6)=r10;
```

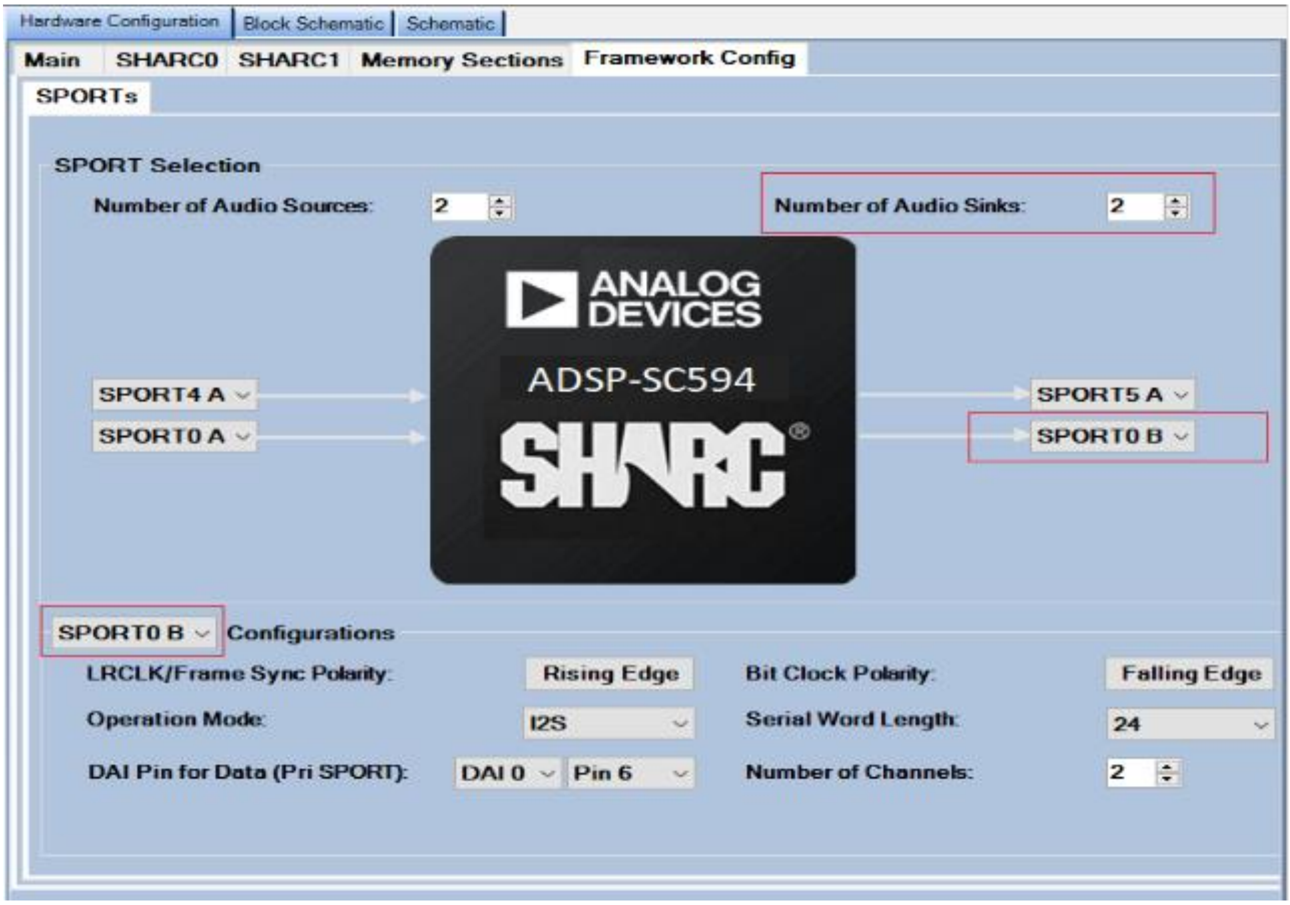
Figure 38: Breakpoint to Module Disassembly Function

29. Step-in through the code using disassembly step and debug the module function.

10 Enabling S/PDIF Transmitter Feature support In Example Demo applications

The S/PDIF transmitter feature is supported on all ADSP-SC5xx/ADSP-215xx processors with example demo application. This feature can be used to transmit S/PDIF data from the evaluation boards. The steps described below are to be followed to use this feature.

1. The target framework application must be rebuilt by adding the compiler preprocessor “SPDIF_TX_ENABLE” in all the cores CCES projects.
2. In SigmaStudio hardware configuration tab, framework configuration must be opened and add 1 more sink for S/PDIF transmitter output. Please refer Figure 39.
3. Assign the desired sport channel based on DAI pin group (For DAI0 – SPORT 0A, 0B, 1A, 1B, 2A, 2B, 3A and 3B / For DAI1 – SPORT 4A, 4B, 5A, 5B, 6A, 6B, 7A and 7B). Please refer Figure 39.
4. In the drop-down menu below in configurations choose the S/PDIF Tx sport channel and update the required setting as 2 channels, I2S format, 24-bit width, frame sync and bit clock polarity. Please refer Figure 39.
5. The DAI pin to be used for the S/PDIF Tx feature is mentioned in the below table which varies for each target evaluation platform.
6. Once the sport configuration changes have been completed, regenerate the config file and overwrite the existing config file in the framework using “Fw Update → Generate Config File”. The generated config file must be overwrite for ADSP-2156x as “adi_ss_fw_config_2156x.h” in *“< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\ Examples\Framework\Include”* folder and for other ADSP-SC5xx/ADSP-215xx processors generated config file must be overwrite as “adi_ss_fw_config.h” in *“< Software Modules folder > \SigmaStudioForSHARC-SH-Rel4.7.0\Target\ Examples\Framework\Include”* folder.
7. In SigmaStudio schematic connect the audio data to output channel 8 and 9 for data to SPORT of S/PDIF Tx
8. Rebuild the example demo application to generate the DXE’s.
9. With all the above changes incorporated the target application and SigmaStudio schematic application are ready to support the S/PDIF Tx feature.



Number of Audio Sources: 2

Number of Audio Sinks: 2

SPORT0 B Configurations

LRCLK/Frame Sync Polarity: Rising Edge

Bit Clock Polarity: Falling Edge

Operation Mode: I2S

Serial Word Length: 24

DAI Pin for Data (Pri SPORT): DAI 0 Pin 6

Number of Channels: 2

Figure 39: Framework Configuration for S/PDIF Transmitter support

The DAI pin on each ADSP-SC5xx/ADSP-215xx evaluation boards which currently supports the S/PDIF Tx feature is mentioned in Table 35. The same DAI pin must be assigned in the SigmaStudio schematic SPORT configuration for S/PDIF Tx Sink.

Evaluation Target Platform	S/PDIF Tx SPORT DAI Pin	S/PDIF Tx Data DAI Pin
ADSP-21569 EZ-KIT	DAI0 Pin 02	DAI0 Pin 10
ADSP-21569 SOM-CRR EZ-KIT	DAI0 Pin 11	DAI0 Pin 10
ADSP-21593/ADSP-SC594 SOM-CRR EZ-KIT	DAI0 Pin 06	DAI0 Pin 10
ADSP-SC584/ADSP-SC589 EZ-KIT	DAI0 Pin 06	DAI0 Pin 20
ADSP-SC573 EZ-KIT	DAI0 Pin 18	DAI0 Pin 10

Table 35: DAI Pins Used for S/PDIF Tx Feature

Note: -

The SPORT selection must be done based on DAI pin group.

The DAI pin for the S/PDIF Tx data can be changed in SRU routing based on developer requirement and the change should be done in SigmaStudio schematic SPORT configuration.

For ADSP-21569 EZ-KIT the SRU configuration should be updated as “DAI0_PB_02_O → SPDIF0_TX_DAT_I” for SPDIF Tx Data input.

The S/PDIF Tx feature not supported for ADSP-SC589 Mini evaluation board.

11 FIR/IIR Hardware Accelerator Multi-Instance support on ADSP-2159x/ADSP-SC59x

The multi-Instance FIR/IIR hardware accelerator supported for ADSP-2159x/ADSP-SC59x.

- All the available SH cores can use all the IIR/FIR accelerator HW channels.
- Same IIR, FIR accelerator HW channel can be used in both the SH cores.
- Any number of instances of same IIR/FIR HW channel is supported in the schematic.
- Any combination of accelerator connection is supported in the schematic.
- If any unsupported IIR/FIR accelerator for a specific processor is present in the schematic, error message will be prompted during Link Compile.
- Other family of target processor IIR/FIR accelerators works same as existing releases.

Processors	**Number Of FIR Accelerators Supported**	**Number Of IIR Accelerators Supported**
ADSP-SC591/ ADSP-SC592	1	4
ADSP-21591/ ADSP-21593/ ADSP-21594/ ADSP-SC594	2	8

Table 36: ADSP-2159x/ADSP-SC59x FIR/IIR Hardware Accelerators

A. Utility for formatting Exported data from SigmaStudio

The ARM as microcontroller application requires the exported data from SigmaStudio to be formatted in a specific way so that the exported information can be used as '.c' arrays within the application. A utility is provided with the installer to do this formatting. The '*exportCodeParam*' utility is located in *"< Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\Target\utilities\exportCodeParam".*

Follow the steps below to format the exported data from SigmaStudio using the '*exportCodeParam' utility*

1. Copy all the SigmaStudio exported files into *"< Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\Target\utilities\exportCodeParam"* folder.

2. Open 'cmd' prompt and change the working directory to *"< Software Modules folder >\SigmaStudioForSHARC-SH-Rel4.7.0\Target\utilities\exportCodeParam"* folder.

3. Run the utility as below:
   exportCodeParam.exe <Visa> <Export File Name> <IC_name> <ProcessorType>
   For e.g. exportCodeParam.exe "SWC" "Volume_Mute_Block_SC5xx" "IC_1" ”SC5xx” or exportCodeParam.exe "SWC" "Volume_Mute_Block_SC5xx" "IC_1" ”215xx”
   This generates C formatted source and header file in the same folder.

This utility must be run multiple times corresponding to the number of ICs within the schematic. For schematics using the 'Dual Core' option, the SMAP, code and parameters for each of the SHARC cores are in a single '.c' and '.h' file.

Terminology

Term	Description
Core	CORE 0 = ARM Cortex A5 CORE 1 = SHARC Core 1 CORE 2 = SHARC Core 2

Table 37: Terminology

References

Reference No.	Description
[1]	AE_42_SS4G_ReleaseNotes.pdf
[2]	ADSP-SC573, ADSP-SC584, ADSP-SC589, ADSP-SC594, ADSP-21593 and ADSP-21569 Evaluation kit Manuals. This manual may be obtained by contacting ADI technical support team as detailed in **[1]**.
[3]	SigmaStudio for SHARC Algorithm Designer Guide (AE_42_SS4G_AlgorithmDesignerGuide.pdf)
[4]	AE_42_SS4G_IntegrationGuide.pdf

Table 38: References

[i] ADSP-SC5xx refers to ADSP-SC589/ADSP-SC584/ADSP-SC573/ADSP-SC594 processor in the entire document.

[ii] ADSP-SC57x refers to ADSP-SC573 processor in the entire document.

[iii] ADSP-SC58x refers to ADSP-SC589/ADSP-SC584 processor in the entire document.

[iv] ADSP-SC59x refers to ADSP-SC594 processor in the entire document.

[v] ADSP-2159x refers to ADSP-21593 processor in the entire document.

[vi] ADSP-2156x refers to ADSP-21569 processor in the entire document.