Processor Audio SDK Setup Guide

Texas Instruments

Rev. 1.10

08 December 2017

Revision History			
Version	**Date**	**Description of changes**	**Author(s)**
0.10	23-Aug-16	Initial version	Frank Livingston
0.20	30-Aug-16	Added gmake clean to UART LLD build instructions	Frank Livingston
0.30	08-Dec-16	Updated for AR2	Frank Livingston
0.40	17-Mar-17	Updated for AR4	Frank Livingston
0.50	14-Jun-17	Updated for BR1	Frank Livingston
0.60	08-Sep-17	Updated for BR3	Frank Livingston
0.70	29-Sep-17	Updated for 1.0.0.6	Frank Livingston
1.00	30-Nov-17	Updated for 1.1.0.0	Frank Livingston
1.10	08-Dec-17	Updated for 1.1.0.1	Frank Livingston

Table of Contents

1 Introduction

This document provides installation and setup instructions for Release 1.1.0.1 (R1.1.0.1) of Processor Audio SDK (PASDK) 1.0 for K2G provided by Texas Instruments, Inc. (TI). R1.1.0.1 incorporates several packages: (1) the Open Source (OS) package; (2) the Firmware Deliverable (FD) package; (3) the Dolby Atmos IP package; and (4) the DTS DTSX IP package. The R1.1.0.1 features provided by these packages are detailed below.

1. OS package
    - Audio input drivers: HDMI input, and 8-channel ADC or S/PDIF (optical only) input.
    - Audio output drivers: up to 16-channel DAC output.
    - Control I/O drivers: UART, SPI and I2C.
    - PCM decoder.
    - PCM encoder with DEL and VOL ASP phases.
    - DEL3, DM, FIL, SRC, ML0 and AE0 ASP's.
    - Framework supporting I-topology, with decoders executing on ARM and ASP's executing on DSP.
2. FD package
    - SNG decoder.
    - BM2, DEM and GEQ3 ASP's.
3. Dolby Atmos IP packages
    - Dolby Intrinsics 1.8.1.1.
    - Dolby Digital Plus (DDP) decoder 4.7.2.
    - Metadata-enhanced Audio Transmission (MAT) decoder 2.1.2.
    - TrueHD (THD) decoder 3.3.3.
    - Channel Audio Renderer (CAR) 1.0.0.
    - Object Audio Renderer (OAR) 1.0.0 & Object Audio Renderer Interface (OARI) 1.1.11 from SIDK 1.3.
    - Bass Management (BMDA) from SIDK 1.3.
4. DTS DTSX IP packages
    - DTSX/PARMA 3.90.50.1.

2 Required Hardware

The hardware listed below is required for complete R1.1.0.1 functionality:

1. Mistral K2G Rev. D EVM (SPRW274).
2. Mistral K2G EVM Audio Daughter Card Rev. B (SPRW279).
3. Momentum Data Systems (MDS) HDMI repeater kit, including:
    - MDS HSR41P HDMI Repeater module Rev. A.
    - MDS DA10X Interface board (IFB) Rev. A.
    - 5V AC to USB power supply for IFB and HSR41P.
    - Short 26- and 34-wire ribbon cables for connecting HSR41P and IFB.
    - Long 34-wire ribbon cable for connecting IFB and Audio Daughter Card.

4. MPC-HC player.
    - It is used to stream DTS streams to K2G EVM via HDMI from Windows PC.
    - The MPC-HC installer can be downloaded at: https://mpc-hc.org/downloads/
    - The detailed test setup and integration information can be found in APPENDIX A.

3 Software Installation

3.1 Code Composer Studio (CCS)

R1.1.0.1 was developed using CCS version 7.1.0.00016 on Windows. Download the Off-line Installer for this version of CCS at the following URL:

http://processors.wiki.ti.com/index.php/Download_CCS#Code_Composer_Studio_Version_7_Downloads

Install CCS 7.1.0 to C:\ti. During CCS installation:

- For "Processor Support", select "66AK2x multicore DSP + ARM Processors & C66x Keystone multicore DSP"
- For "Select Debug Probes", select "Spectrum Digital Probes and Boards"

R1.1.0.1 was developed using the code generation tools listed below.

- GNU ARM Tools 4.8.0.2014q3-20140805.
    - The installer can be downloaded at https://launchpad.net/gcc-arm-embedded/4.8/4.8-2014-q3-update.
    - Install in c:\ti\gcc-arm-none-eabi-4_8-2014q3.
    - Note this is also installed as part of CCS 6.1.2 in C:\ti\ccsv6\tools\compiler\gcc-arm-none-eabi-4_8-2014q3.
- C6000 v8 Compiler Tools 8.1.0.
    - The installer can be downloaded at http://software-dl.ti.com/codegen/non-esd/downloads/download.htm.
    - Install in C:\ti\ti-cgt-c6000_8.1.0.
    - Note this is also installed as part of CCS 6.1.2 in C:\ti\ccsv6\tools\compiler\ti-cgt-c6000_8.1.0.

3.2 Processor SDK-RTOS

R1.1.0.1 was developed using Processor SDK-RTOS 4.0.0.4 for K2G on Windows. Release information for this version of Processor SDK-RTOS is provided at the URL below, along with a link to the installer:

http://software-dl.ti.com/processor-sdk-rtos/esd/K2G/04_00_00_04/index_FDS.html

Install all components of Processor SDK-RTOS to C:\ti. Among others, this will install the tools in the following list:

- SYS/BIOS 6.46.05.55

- XDAIS 7.24.00.04
- XDC Tools 3.32.01.22

3.3 Open Source Package

The R1.1.0.1 OS package is distributed using a self-extracting executable. Follow the steps below to install the software.

1. Double click the executable to initiate the installation process. Continue to the License Agreement.
2. Accept the terms of the click-wrap license agreement.
3. Change the installation destination folder if desired, or accept the default c:\ti\processor_ sdk_audio_1_01_00_01 in the Choose Destination Location window. Complete the installation.

Upon completion of the installation, the installation root folder (<pasdk_root>) contains the sub-folders described in the table below.

Folder	Contents
docs\	User documentation
pasdk\	PASDK OS Dev (paf\) ARM application (common\, shared\, test_arm\ DSP application (common\, shared\, test_dsp\)
psdk_cust\	Library Architecture and Framework (libarch_k2g_1_0_1_0\)
scripts\	Build scripts
tools\	Alpha communication tools

3.4 Firmware Deliverable Package

The R1.1.0.1 FD package is distributed as a .zip archive. To install the software, unzip the package .zip file to <pasdk_root>\pasdk\paf. The FD package root folder is "pa", so unzipping the file should overwrite the pa folder in paf and overwrite or create the sub-folders described in the table below.

Folder	Contents
pa\asp	ASP IP (bm2, dem, geq3) header files
pa\build	Decoder and ASP IP libraries for ARM (sng1) and DSP (bm2, dem, geq3)
pa\dec	Decoder IP (sng1) header files
pa\docs	User documentation

3.5 Dolby IP Packages

The R1.1.0.1 Dolby IP packages are distributed using a select-extracting executable. Follow the steps below to install the software.

1. Double click the executable to initiate the installation process. Continue to the License Agreement.
2. Accept the terms of the click-wrap license agreement.

3. Accept the default installation folder, or change the folder to the desired location of the Dolby IP packages (e.g. <pasdk_root>\release\dhip_package). Complete the installation.
4. Upon completion of the installation, unzip all Dolby IP .zip packages (bmda, car, oar, ddp, intrinsics, matthd, and oar) to <pasdk_root>\dolby_ip\dh-ip. After unzipping the Dolby IP .zip packages, the dh-ip folder should contain the sub-folders described in the table below.

Folder	**Contents**
asp\	ASP IP (bmda, car, oar) header files
build\	IP libraries for ARM (ddp, mat, thd) and DSP (bmda, car, instrinsics, oar)
dec\	Decoder IP (ddp, mat-thd) header files
DOC\	User documentation

3.6 DTS IP Packages

The R1.1.0.1 DTS IP packages are distributed using a select-extracting executable. Follow the steps below to install the software.

1. Double click the executable to initiate the installation process. Continue to the License Agreement.
2. Accept the terms of the click-wrap license agreement.
3. Accept the default installation folder, or change the folder to the desired location of the DTS IP packages (e.g. <pasdk_root>\release\dtsip_package). Complete the installation.
4. Upon completion of the installation, the unzip dtsx and parma .zip packages to <pasdk_root>\3p-ip-dts\dtsx-ip. After unzipping .zip packages, the dtsx-ip folder should contain the sub-folders described in the table below.

Folder	**Contents**
asp\	ASP IP (parma) header files
build\	IP libraries for ARM (dtsx) and DSP (parma)
dec\	Decoder IP (dtsx) header files
DOC\	User documentation

3.7 CCS Update

Update the CCS 7.1.0 installation as below:

1. Launch CCS and create a new workspace (e.g. <pasdk_root>\pasdk\workspace_pasdk).
2. Allow installation of newly discovered products located in c:\ti.
   - Ignore any messages about unsigned content
   - Restart CCS when prompted
3. By default CCS will recognize codegen tools installed to c:\ti and c:\ti\ccsv7\tools\compiler. If codegen tools are installed to other folders, the CCS tools discovery path must be updated for CCS to recognize these tools. For example, if codgen tools are installed in the CCS 6.1.2 default codegen tools installation folder c:\ti\ccsv6\tools\compiler, update the CCS tools discovery path as follows to access these tools:
   - Select Window→Preferences
   - Select Code Composer Studio→Build→Compilers

- Click Add next to Product Discovery Path, add folder c:/ti/ccsv6/tools/compiler

3.8 PASDK OS Dev Build and Alpha Communication Tools

Download and install the additional tools below. Sed and Python are required to build the PASDK OS DEV software contained in the <pasdk_root>\pasdk\paf folder. Python is also required for alpha command control I/O over using the PyAlpha tool. The PyAlpha tool is further described below in Section 6.4.

1. Obtain Sed 4.2.1 from http://gnuwin32.sourceforge.net/packages/sed.htm,
   executable installer, sed-4.2.1-setup.exe.
   Install to c:\PA_Tools\GnuWin32 (or change installation location in
   <pasdk_root>\pasdk\setup_env.bat).
2. Obtain Python 2.7.14 from https://www.python.org/downloads/release/python-2714/
   Windows x86 MSI installer, python-2.7.14.msi.
   Install to c:\PA_Tools\Python27 (or change installation location in
   <pasdk_root>\pasdk\setup_env.bat).
3. Once Python is installed, run **<pasdk_root>\scripts\setup_env.bat** and then navigate to **<pasdk_root>\tools\** and run **setup.bat** to install PyAlpha dependencies.

4 Emulator Firmware Update

The K2G EVM on-board emulator firmware must be updated for correct emulator operation. Update the firmware as described below.

1. Connect the EVM to the host PC using the provided USB Mini B to A plug type cable. Connect the Mini B plug on the cable to XDS_USB connector J1 on the EVM.
2. Connect the EVM to power using the supplied 12V power supply.
3. Apply power to the EVM by pushing the SW1 switch to the “ON” position.
4. Open a DOS prompt on the PC. At the DOS prompt, issue these commands:
   ```
   > cd c:\ti\ccsv6\ccs_base\common\uscif\xds2xx
   > update_xds2xx.bat xds200
   ```
5. The batch file output should appear as shown below.
   ```
   Updating CPLD ...
   .
   Updating Firmware ...
   .
   Rebooting ...
   .
   Reading Configuration ...
   .
   Check swRev is 1.0.0.8 or higher
   .
   boardRev=4
   ipAddress=0.0.0.0
   ipConfig=dhcp
   ipGateway=0.0.0.0
   ipNetmask=0.0.0.0
   ```

```
productClass=XDS2XX
productName=XDS200
serialNum=00:0E:99:03:90:0F
swRev=1.0.0.8
hostCPU=AM1802
emuCtrlType=Bit bang
extMemType=SDRAM
portUSB=true
portENET=false
portWIFI=false
portRS232=false
EnableUSBSerial=false
CurrentMeasure=false
Press any key to continue . . .
```

5 Software Build

5.1 PASDK OS Dev Libraries

The ARM and DSP applications depend on several libraries located in the <pasdk_root>\pasdk\paf folder. Detailed instructions for building any of the libraries in this folder are contained in <pasdk_root>\pasdk\paf\ReadMe.txt. However, batch files for building only the necessary libraries for the ARM/DSP applications are provided in the <pasdk_root>\scripts folder. To build the required libraries:

1. Open a DOS prompt.
2. From the DOS prompt in <pasdk_root>, execute:
   ```
   > cd scripts
   > setup_env.bat
   > build_paf_libs.bat
   ```

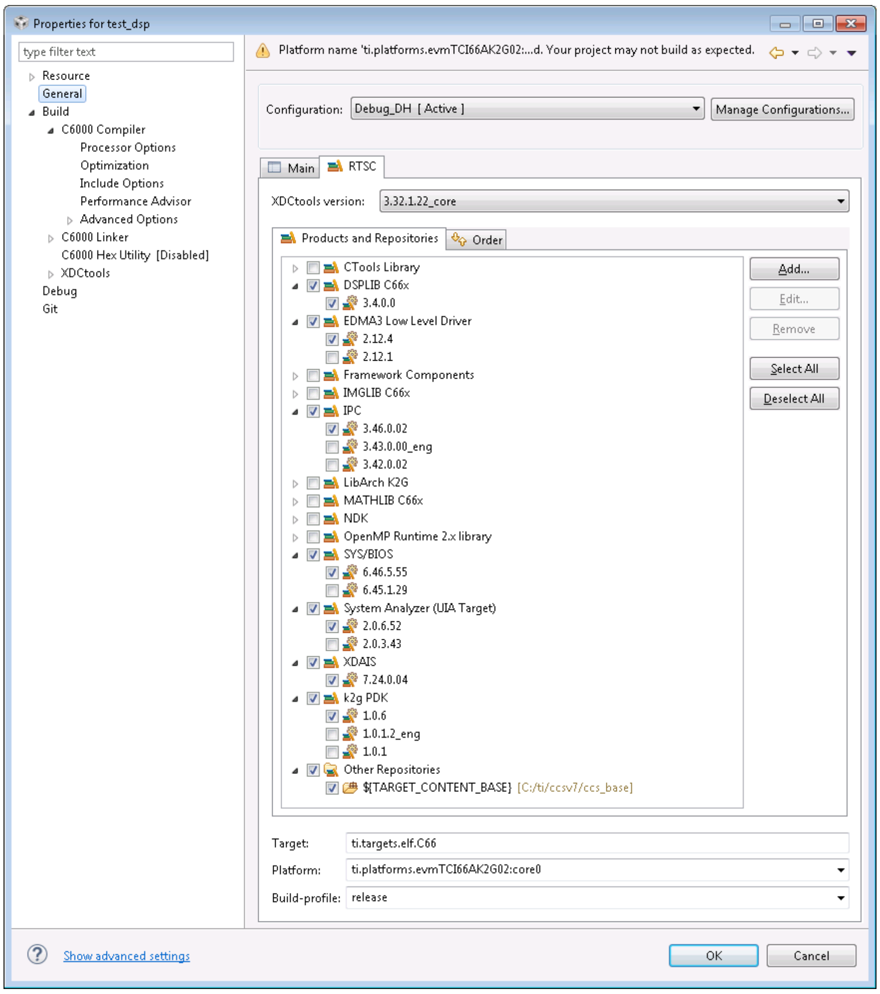
5.2 DSP Application

To build the DSP application:

1. Import the "test_dsp" CCS project into the CCS workspace. The project is located in <pasdk_rooot>\pasdk\test_dsp.
2. Select the desired build configuration
   - For Dolby Harmony: right-click test_dsp project in CCS Project Explorer, select Build Configurations→Set Active→Project→Debug_DH_IPPkgs.
   - For DTS:X: right-click test_dsp project in CCS Project Explorer, select Build Configurations→Set Active→Project→ Debug_DTSX_IPPkgs.
   - For No IP: right-click test_dsp project in CCS Project Explorer, select Build Configurations→Set Active→Project→ Debug_NoIP.
3. Ensure the correct component software versions are selected in the RTSC control panel (See Figure 1 below).
   - Right-click test_dsp project in CCS Project Explorer, select Properties.

- On left-hand side of window, select General. Select RTSC tab on right-hand side of General window.
- Confirm the following PrSDK 4.00.00.04 components are installed and selected:
  1. SYS/BIOS 6.46.5.55.
  2. System Analyzer (UIA Target) 2.0.6.52.
  3. XDAIS 7.24.0.04.
  4. DSPLIB C66x 3.4.0.0.
  5. IPC 3.46.00.02.
  6. EDMA3 LLD 2.12.4.
  7. k2g PDK 1.0.6.
- Click OK.

4. Rebuild the project using the selected build profile.
   - Right-click test_dsp project in CCS Project Explorer, select Rebuild Project.


Properties for test_dsp

Platform name 'ti.platforms.evmTCI66AK2G02:...d. Your project may not build as expected.

Configuration: Debug_DH [Active]

XDCtools version: 3.32.1.22_core

Products and Repositories:

```
[ ] CTools Library
[✓] DSPLIB C66x
    [✓] 3.4.0.0
[✓] EDMA3 Low Level Driver
    [✓] 2.12.4
    [ ] 2.12.1
[ ] Framework Components
[ ] IMGLIB C66x
[✓] IPC
    [✓] 3.46.0.02
    [ ] 3.43.0.00_eng
    [ ] 3.42.0.02
[ ] LibArch K2G
[ ] MATHLIB C66x
[ ] NDK
[ ] OpenMP Runtime 2.x library
[✓] SYS/BIOS
    [✓] 6.46.5.55
    [ ] 6.45.1.29
[✓] System Analyzer (UIA Target)
    [✓] 2.0.6.52
    [ ] 2.0.3.43
[✓] XDAIS
    [✓] 7.24.0.04
[✓] k2g PDK
    [✓] 1.0.6
    [ ] 1.0.1.2_eng
    [ ] 1.0.1
[✓] Other Repositories
    [✓] ${TARGET_CONTENT_BASE} [C:/ti/ccsv7/ccs_base]
```

Target: ti.targets.elf.C66

Platform: ti.platforms.evmTCI66AK2G02:core0

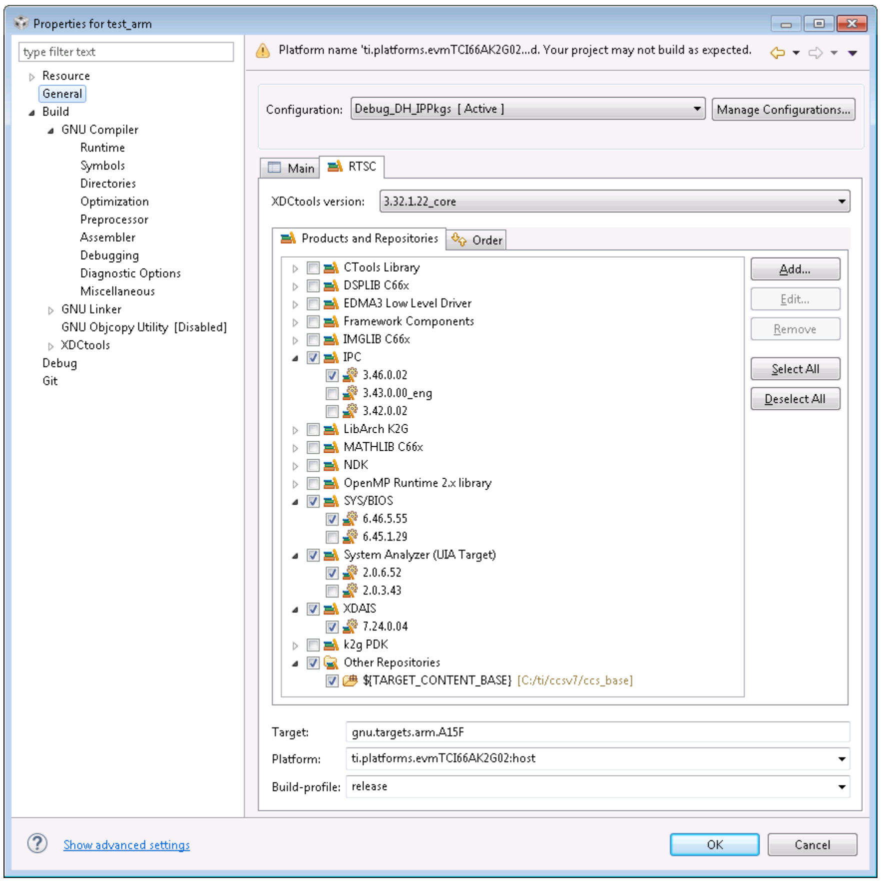
Build-profile: release

Figure 1: DSP Application RTSC Configuration

5.3 ARM Application

To build the ARM application:

1. Import the "test_arm" CCS project into the CCS workspace. The project is located in <pasdk_root>\pasdk\test_arm.
2. Select the desired build configuration
    - For Dolby Harmony: right-click test_arm project in CCS Project Explorer, select Build Configurations→Set Active→Project→Debug_DH_IPPkgs.
    - For DTS:X: right-click test_arm project in CCS Project Explorer, select Build Configurations→Set Active→Project→Debug_DTSX_IPPkgs.
    - For No IP: right-click test_arm project in CCS Project Explorer, select Build Configurations→Set Active→Project→ Debug_NoIP.
3. Ensure the correct component software versions are selected in the RTSC control panel (See Figure 2 below).
    - Right-click test_arm project in CCS Project Explorer, select Properties.
    - On left-hand side of window, select "General". Select RTSC tab on right-hand side of General window.
    - Confirm the following PrSDK 2.00.02.11 components are installed and selected:
        1. SYS/BIOS 6.46.5.55.
        2. System Analyzer (UIA Target) 2.0.6.52.
        3. XDAIS 7.24.0.04.
        4. IPC 3.46.00.02
    - Click OK.
4. Rebuild the project using the selected build profile.
    - Right-click test_arm project in CCS Project Explorer, select Rebuild Project.



Platform name 'ti.platforms.evmTCI66AK2G02...d. Your project may not build as expected.

Configuration: Debug_DH_IPPkgs [Active]

XDCtools version: 3.32.1.22_core

- [ ] CTools Library
- [ ] DSPLIB C66x
- [ ] EDMA3 Low Level Driver
- [ ] Framework Components
- [ ] IMGLIB C66x
- [✓] IPC
  - [✓] 3.46.0.02
  - [ ] 3.43.0.00_eng
  - [ ] 3.42.0.02
- [ ] LibArch K2G
- [ ] MATHLIB C66x
- [ ] NDK
- [ ] OpenMP Runtime 2.x library
- [✓] SYS/BIOS
  - [✓] 6.46.5.55
  - [ ] 6.45.1.29
- [✓] System Analyzer (UIA Target)
  - [✓] 2.0.6.52
  - [ ] 2.0.3.43
- [✓] XDAIS
  - [✓] 7.24.0.04
- [ ] k2g PDK
- [✓] Other Repositories
  - [✓] ${TARGET_CONTENT_BASE} [C:/ti/ccsv7/ccs_base]

Target: gnu.targets.arm.A15F

Platform: ti.platforms.evmTCI66AK2G02:host

Build-profile: release

Figure 2: ARM Application RTSC Configuration

6 Application Execution and Test

6.1 I/O Configuration

R1.1.0.1 supports boot- and run-time I/O configuration. Boot-time I/O configuration is determined by the at-boot I/O shortcuts in the CUS_ATBOOT_S definition in <pasdk_root>\pasdk\test_dsp\application\itopo\evmk2g\atboot.c, while run-time I/O configuration is performed by invoking I/O shortcuts via alpha commands. The available Input and Output shortcuts are defined in <pasdk_root>\pasdk\test_dsp\application\itopo\evmk2g\io.c and described in the tables below.

Input Configuration	**Input Shortcut**
ADC 8-ch (w/ DAC 8-ch input)	execPAIInAnalog
S/PDIF	execPAIInDigital
Stereo HDMI (PCM or DDP)	execPAIHDMIInStereo
HDMI (MAT-THD/PCM)	execPAIInHDMI

Output Configuration	**Output Shortcut**
DAC 8-ch (w/ ADC 8-ch input)	execPAIOutAnalogSlave
DAC (8-ch)	execPAIOutAnalog
DAC (12-ch)	execPAIOutAnalog12Ch
DAC (16-ch)	execPAIOutAnalog16Ch

6.2 Connect EVM

1. Turn off power to the EVM using SW1 on the EVM. Disconnect the 12V power supply.
2. Assemble the Audio Daughter Card (assembly instructions are packaged with Audio Daughter Card).
3. Attach the Audio Daughter Card to the K2G EVM. The Audio Daughter Card connects to the EVM through the JP1 “Audio Expansion” and J12 “Serial Expansion” connectors on EVM.
4. Connect the PC and K2G EVM using DB9 serial RS-232 cable delivered with EVM.
    - A USB to UART FTDI cable can be used on the PC side of the connection.
    - Attach the DB9 connector to the P1 “UART0” connector on the EVM.
    - Push the SW12-UART0_DB9_EN switch to the “ON” position.
5. Connect the EVM emulation port to the PC as described in Section 4.
6. Connect the MDS HSR41P to the MDS IFB.
    - Connect HSR41P J4 “Main” header to the IFB J5 “Main” header using the short 26-wire ribbon cable. Ensure the red wire on the ribbon cable is connected between HSR41P J4 Pin1 and IFB J5 Pin1.
    - Connect HSR41P J7 “Sec” header to the IFB J2 “Sec” header using the short 34-wire ribbon cable. Ensure the red wire on the ribbon cable is connected between HSR41P J7 Pin1 and IFB J2 Pin1.
7. Connect the MDS IFB to the Audio Daughter Card. Connect IFB J3 “McASP0” header to Audio Daughter Card “I2S Header” using the supplied long 34-wire ribbon cable. Ensure the red wire on the ribbon cable is connected between IFB J3 Pin1 and Audio Daughter Card “I2S Header” Pin1 (note the “I2S Header” is only 32 pins, so the two wires on the ribbon cable opposite Pin1 are unused).
8. Connect HDMI cable from audio source (e.g. Blu-ray player) to HSR41P J6 “IN”. Connect HDMI cable to audio sink (e.g. HDMI enabled monitor) from HSR41P J8 “HDMI OUT”.
9. Connect 5V AC to USB power supply to IFB power connector J1.
10. Re-connect the 12V power supply to the EVM. Turn on power to the EVM using SW1.

6.2.1 ADC 8-Ch Input and DAC 8-Ch Output

Connect ADC inputs to the ADC_IN0-3 input jacks on the Audio Daughter Card. Processed PCM samples will be output to the corresponding DAC output jacks DAC_OUT0-3 on the Daughter Card.

6.2.2 S/PDIF Input and DAC 8-Ch Output

Connect S/PDIF input to the MOD1 Digital Input (Optical) connector on the Audio Daughter Card. Processed PCM samples will be output to the DAC output jacks DAC_OUT0-3 on the Daughter Card.

6.2.3 HDMI Input and DAC Output

Connect HDMI input source to the HSR41P repeater module as described above. Processed PCM samples will be output to DAC output jacks DAC_OUT0-6 on the Daughter Card. The number of active DAC outputs depends on: (1) the selected Output Shortcut (execPAIOutAnalog, execPAIOutAnalog12Ch, or execPAIOutAnalog16Ch); (2) the configured ENC channel map from; and (3) the active renderer channel configuration request. For example, issue the alpha commands listed below to enable 12-channel DAC for channel-based audio using CAR with speaker layout L/R, C, Sub, Ls/Rs, Lrs/Rrs, Ltf/Rtf, Ltr/Rtr:

```
CMD> python pyalpha execPAIOutAnalog12Ch
alpha execPAIOutAnalog12Ch

CMD> python pyalpha
writeENCChannelMapFrom16(PAF_LEFT,PAF_RGHT,PAF_LSUR,PAF_RSUR,PAF_CNTR,PAF_SUBW,PAF_LBA
K,PAF_RBAK,PAF_LTFT,PAF_RTFT,PAF_LTRR,PAF_RTRR,-3,-3,-3,-3)

CMD> python pyalpha writeSYSChannelConfigurationRequestSurround4LtfRtfLtrRtr_1
alpha /* none */

CMD> python pyalpha readCARChannelCfgOverride
alpha writeCARChannelCfgOverrideUnknown
```

6.3 Load and Execute Code

Follow the steps below from within CCS to load and execute the DSP and ARM application code via JTAG:

1. Click View→Target Configurations.
2. In the Target Configurations window, open the Projects→test_dsp→targetConfigs folder.
3. Right-click on K2GEVM.ccxml contained in the targetConfigs folder.
4. Select Launch Selected Configuration.
5. In the Debug window, right-click on the C66x and select Connect Target. The output from the GEL code invoked on the target connection can be observed in the Console output window. The final line of this output should read "`C66xx: GEL Output: DDR3A initialization complete`".
6. In the Debug window, right-click on the CortexA15 and select Connect Target. The output from the GEL code invoked on the target connection can be observed in the Console output window. The final line of this output should read "`CortexA15: GEL Output: A15 non secure mode entered`".
7. In the Debug Window, click on the C66x and then open Run→Load→Load Program. In the Load Program window, click on Browse project, open the "test_dsp" folder, and click on "test_dsp.out" and OK to select the program to be loaded. Click on OK in the Load Program window to load the program to the C66x.
8. In the Debug Window, click on the CortexA15, and then open Run→Load→Load Program. In the Load Program window, click on Browse project, open the "test_arm" folder, and click on "test_arm.out" and OK to select the program to be loaded. Click on OK in the Load Program window to load the program to the A15.

9. In the Debug window, click on the C66x and then Open Run→Resume to execute the DSP code.
10. In the Debug window, click on the CortexA15 and then Open Run→Resume to execute the ARM code.
11. The CIO Console output window should display memory usage statistics. The final line of output should display the memory usage summary for the EXT NC SHM heap.

6.4 PyAlpha Python Scripts for Alpha Communication

R1.1.0.1 supports alpha command control over UART using the PyAlpha tool located in <pasdk_root>\tools. PyAlpha is invoked on the command-line in a DOS shell. PyAlpha documentation can be obtained by executing PyAlpha with a “--help” argument as below (note: the Python executable is contained in the DOS shell path for this example).

```
> T:
> cd tools
> python pyalpha --help
usage: PyAlpha [-?] [-h FILE] [-I DIR] [-p PORT] [-b BAUD] [-t DIR] [-bin DIR]
               [-r] [-s] [--fast] [--repeat N T N T] [-log LEVEL]
               [ALPHAS [ALPHAS ...]]

Alpha Command Communicator

positional arguments:
  ALPHAS                Alpha commands, separated by spaces

optional arguments:
  -?, --help            show this help message and exit
  -h FILE, --header FILE
                        Add FILE_a.h to symbol definition list and FILE_a.hdM
                        to symbol decomposition list
  -I DIR, --include DIR
                        Add DIR to the include path. [default: ./alpha]
  -p PORT, --port PORT  Communication port [default: COM1]
  -b BAUD, --baud BAUD  Baud rate [default: 19200]
  -t DIR, --temp DIR    Temporary file save location [default: .]
  -bin DIR              Binary file location [default: ./bin]
  -r, --retain          Retain temporary files
  -s, --session         Run a PyAlpha session, retains startup settings
  --fast                Skip post-processing
  --repeat N T N T      Repeat an alpha command N times with a T second break
                        between
  -log LEVEL, --loglevel LEVEL
                        Set log level
```

The file <tools>\config_cust.ini is used to configure default settings for PyAlpha and should be edited to reflect your system (these configurations can be overwritten with CLI flags).

When running PyAlpha with the –s flag, you can use tab-completion once 3 or more characters are typed into the console.

6.5 Alpha Commands for AE0 Configuration/Status

R1.1.0.1 includes the Audio Example 0 (AE0) ASP. This ASP transforms stereo (Left/Right) PCM into 3stereo (Left/Right/Center) PCM. AE0 can be included in the signal chain by defining the _AE0_ macro in <pasdk_root>\pasdk\test_dsp\framework\itopo\patchs.c and rebuilding the DSP application. AE0 documentation is provided in <pasdk_root>\docs\DA10x_UG_AspExample.pdf, and the alpha commands contained in <pasdk_root>\tools\alpha\ae_a.h can be used to configure AE0 and consult AE0 status.

In the example alpha command sequence below, AE0 is configured to generate 3stereo using a gain of 0.5 (16384 in S16Q15 format) using COM port 1. After disabling AE0, the Master Volume Control is set to -20 (= 40*0.5 dB).

*(Note: '**CMD>**' indicates the commands issued from a DOS shell and '**>**' indicates commands from a PyAlpha session).*

```
CMD> T:
CMD> cd tools
CMD> python pyalpha --session
> readAEMode
alpha writeAEModeDisable
> writeAEModeEnable
alpha /* none */
> readAEMode
alpha writeAEModeEnable
> readAEScaleQ15
alpha wroteAEScaleQ15,0x0000
> writeAEScaleQ15(16384)
alpha
> readAEScaleQ15
alpha wroteAEScaleQ15,0x4000
> readSYSChannelConfigurationRequest
alpha 0xce20,0x2008,0x010c,0x0000,0x0000,0x0000
> writeSYSChannelConfigurationRequestStereo
alpha /* none */
> readSYSChannelConfigurationRequest
alpha 0xce20,0x2008,0x0003,0x0000,0x0000,0x0000
> writeSYSChannelConfigurationRequestSurround4_1
alpha /* none */
> readSYSChannelConfigurationRequest
alpha 0xce20,0x2008,0x010c,0x0000,0x0000,0x0000
> writeAEModeDisable
alpha /* none */
> readAEMode
alpha writeAEModeDisable
> writeVOLControlMasterN(-40)
alpha /* none */
```

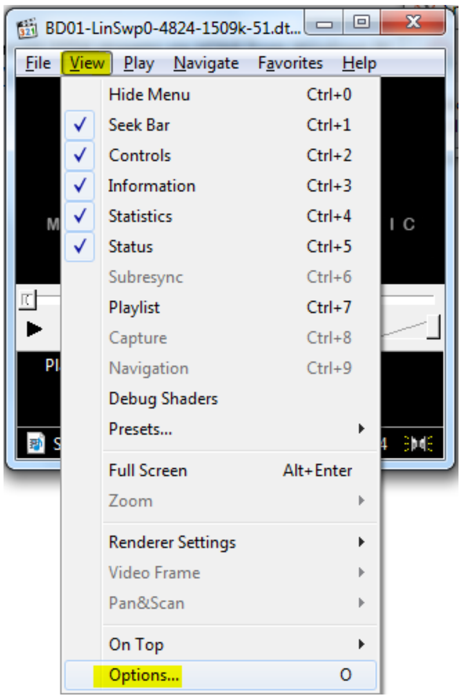
7 APPENDIX A: DTSX Test Setup and Integration

7.1 Test Setup

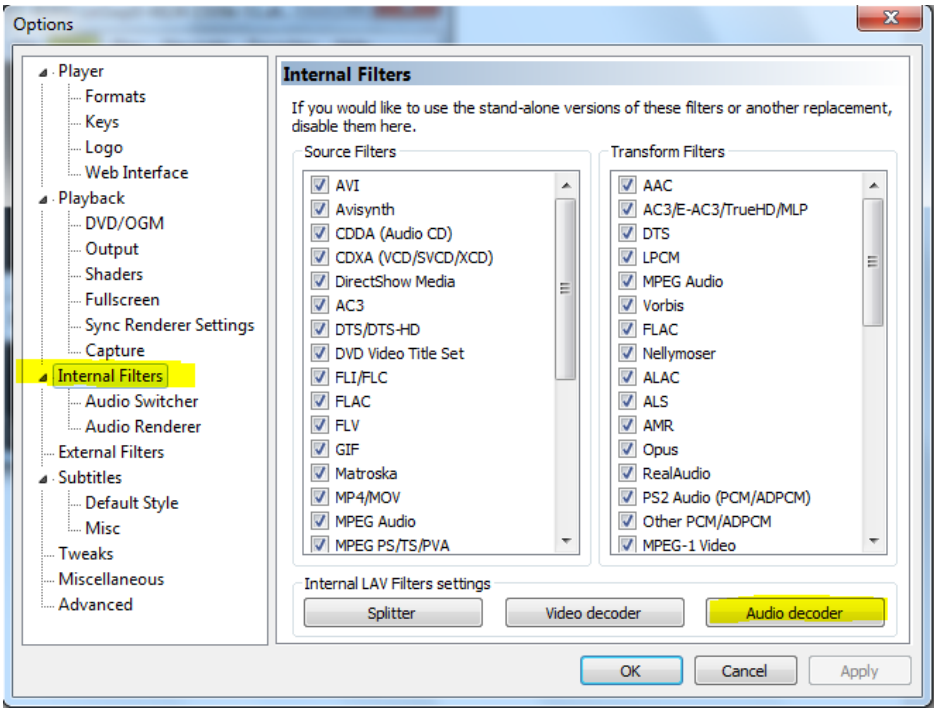
MPC-HC player is used to stream DTS streams to K2G EVM via HDMI from Windows PC. The MPC-HC installer can be downloaded at: https://mpc-hc.org/downloads/.

Once MPC-HP player is installed, follow the following setup steps:

1. GOTO options in View menu:



2. Select Internal Filters > Audio Decoder:



Internal Filters

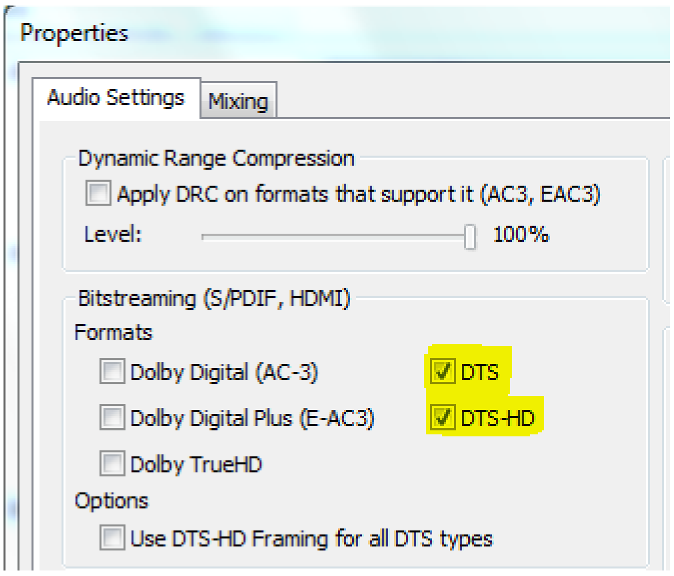
If you would like to use the stand-alone versions of these filters or another replacement, disable them here.

Source Filters
- ✓ AVI
- ✓ Avisynth
- ✓ CDDA (Audio CD)
- ✓ CDXA (VCD/SVCD/XCD)
- ✓ DirectShow Media
- ✓ AC3
- ✓ DTS/DTS-HD
- ✓ DVD Video Title Set
- ✓ FLI/FLC
- ✓ FLAC
- ✓ FLV
- ✓ GIF
- ✓ Matroska
- ✓ MP4/MOV
- ✓ MPEG Audio
- ✓ MPEG PS/TS/PVA

Transform Filters
- ✓ AAC
- ✓ AC3/E-AC3/TrueHD/MLP
- ✓ DTS
- ✓ LPCM
- ✓ MPEG Audio
- ✓ Vorbis
- ✓ FLAC
- ✓ Nellymoser
- ✓ ALAC
- ✓ ALS
- ✓ AMR
- ✓ Opus
- ✓ RealAudio
- ✓ PS2 Audio (PCM/ADPCM)
- ✓ Other PCM/ADPCM
- ✓ MPEG-1 Video

Internal LAV Filters settings: Splitter | Video decoder | Audio decoder

3. Make the setting as shown below:



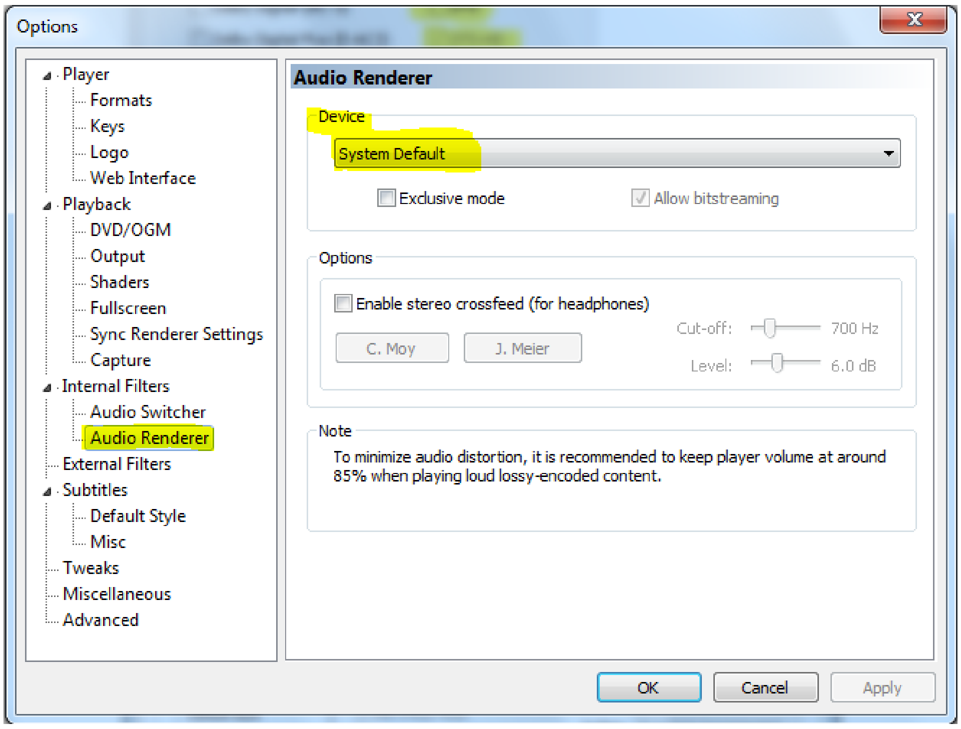
Dynamic Range Compression
- ☐ Apply DRC on formats that support it (AC3, EAC3)
- Level: 100%

Bitstreaming (S/PDIF, HDMI)

Formats
- ☐ Dolby Digital (AC-3)
- ☐ Dolby Digital Plus (E-AC3)
- ☐ Dolby TrueHD
- ✓ DTS
- ✓ DTS-HD

Options
- ☐ Use DTS-HD Framing for all DTS types

4. Select Audio Renderer, make sure it is System Default:


Options

- Player
  - Formats
  - Keys
  - Logo
  - Web Interface
- Playback
  - DVD/OGM
  - Output
  - Shaders
  - Fullscreen
  - Sync Renderer Settings
  - Capture
- Internal Filters
  - Audio Switcher
  - Audio Renderer
- External Filters
- Subtitles
  - Default Style
  - Misc
- Tweaks
- Miscellaneous
- Advanced

Audio Renderer

Device: System Default

☐ Exclusive mode ☑ Allow bitstreaming

Options

☐ Enable stereo crossfeed (for headphones)

C. Moy J. Meier

Cut-off: 700 Hz

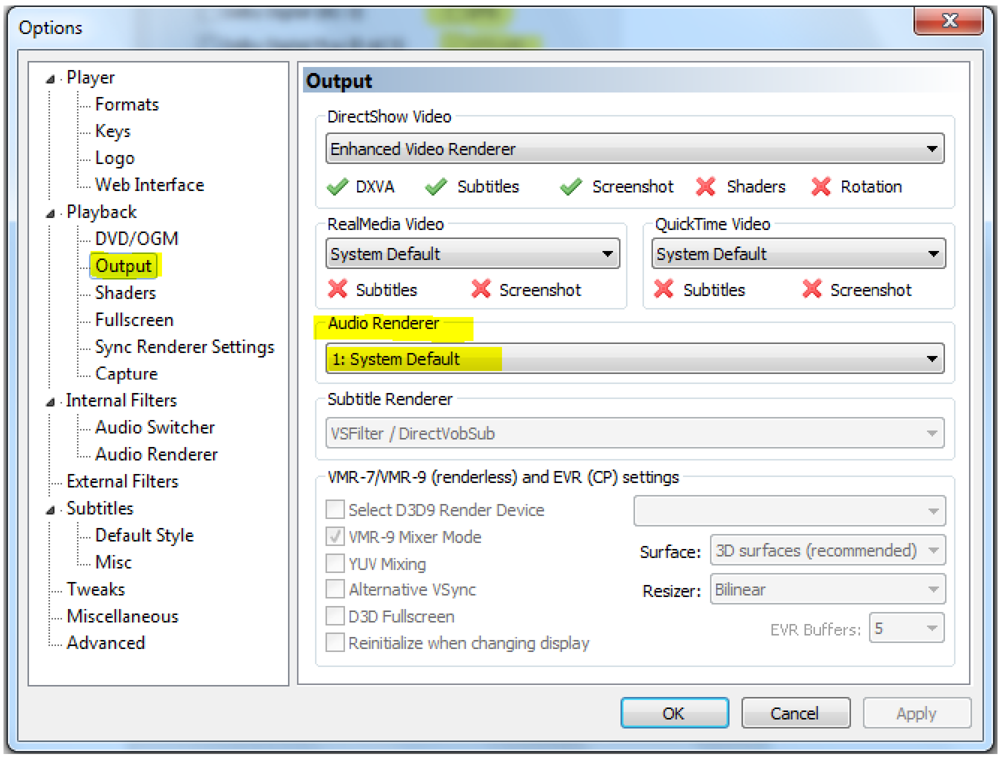
Level: 6.0 dB

Note

To minimize audio distortion, it is recommended to keep player volume at around 85% when playing loud lossy-encoded content.

OK Cancel Apply

5. Select Output, make sure it is System Default:


Options

- Player
  - Formats
  - Keys
  - Logo
  - Web Interface
- Playback
  - DVD/OGM
  - Output
  - Shaders
  - Fullscreen
  - Sync Renderer Settings
  - Capture
- Internal Filters
  - Audio Switcher
  - Audio Renderer
- External Filters
- Subtitles
  - Default Style
  - Misc
- Tweaks
- Miscellaneous
- Advanced

Output

DirectShow Video: Enhanced Video Renderer

✓ DXVA ✓ Subtitles ✓ Screenshot ✗ Shaders ✗ Rotation

RealMedia Video: System Default

✗ Subtitles ✗ Screenshot

QuickTime Video: System Default

✗ Subtitles ✗ Screenshot

Audio Renderer: 1: System Default

Subtitle Renderer: VSFilter / DirectVobSub

VMR-7/VMR-9 (renderless) and EVR (CP) settings

☐ Select D3D9 Render Device

☑ VMR-9 Mixer Mode

☐ YUV Mixing

☐ Alternative VSync

☐ D3D Fullscreen

☐ Reinitialize when changing display

Surface: 3D surfaces (recommended)

Resizer: Bilinear

EVR Buffers: 5

OK Cancel Apply

7.2 Integration

DTSX decoder is integrated on K2G ARM core and PARMA is integrated on K2G C66x core.

7.2.1 DTSX Decoder Integration on K2G ARM Core

1. Get DTSX decoder component libraries from DTSX IP release package:

   - dts-3d.lib
   - dts-base.lib
   - dts-flib.lib
   - dts-lbr.lib
   - dts-lbr-dec.lib
   - dts-parma-dec.lib
   - dtsx-c-decoder.lib
   - la-strm-reconstruction.lib
   - dts-alg.lib

2. Modify patchs.c on ARM to include DTSX decoder into the system.

```
#define _DTS_
#ifdef _DTS_
#include <dtsuhda.h>
#include <dtsuhda_tid.h>
#endif
const PAF_ASP_LinkInit decLinkInitI13[] =
{
   PAF_ASP_LINKINIT(STD, PCM, MDS),

#ifdef _DTS_
   PAF_ASP_LINKINIT(STD, DTSUHDA, TID),
#endif

   PAF_ASP_LINKNONE,
};
```

3. Modify .cmd file to link the DTSX libraries into a build.
   For example, copy all the libraries in C:\ti\processor_sdk_audio_1_01_00_01\3p-ip-dts\dtsx-ip\build\a15\release\ folder and add the following lines in .cmd.

```
SEARCH_DIR ( C:\ti\processor_sdk_audio_1_01_00_01\3p-ip-dts\dtsx-ip\build\a15\release )

INPUT ( dts-3d.lib dts-base.lib dts-flib.lib dts-lbr.lib dts-lbr-dec.lib dts-parma-dec.lib dtsx-c-decoder.lib
la-strm-reconstruction.lib dts-alg.lib )
```

4. Add the path to DTSX header files in CCS Project Directories
   For Example:

```
"${PROC_AUDIO_SDK_ROOT}/3p-ip-dts/dtsx-ip/dec/rel/dtsx/alg"
"${PROC_AUDIO_SDK_ROOT}/3p-ip-dts/dtsx-ip/dec/rel/dtsx/alpha"
```

7.2.2 PARMA Integration on K2G C6x Core

1. Get PARMA component libraries from DTSX IP release package.

   - dts-3d.lib
   - dts-base.lib
   - dts-flib.lib
   - dts-lbr.lib
   - dts-lbr-dec.lib
   - dts-parma-dec.lib
   - dtsx-c-decoder.lib
   - la-strm-reconstruction.lib
   - dts-alg.lib

2. Modify patchs.c on DSP to include PARMA into the system.

```
#define _DTS_
#ifdef _DTS_
#include <dtsuhdb.h>
#include <dtsuhdb_mds.h>
#endif
const PAF_ASP_LinkInit aspLinkInitAllI13[] =
{
      .
      #ifdef _DTS_
         PAF_ASP_LINKINIT(STD, DTSUHDB, MDS),
      #endif
         PAF_ASP_LINKNONE,
};
```

3. Modify .cmd file to link the PARMA libraries into a build.
   For example, copy all the libraries in C:\ti\processor_sdk_audio_1_01_00_01\3p-ip-dts\dtsx-ip\build\c66x\release\ folder and add the following lines in .cmd.

```
 -i"C:\ti\processor_sdk_audio_1_01_00_01\3p-ip-dts\dtsx-ip\build\c66x\release"

-l"dts-3d.lib"
-l"dts-base.lib"
-l"dts-flib.lib"
-l"dts-lbr.lib"
-l"dts-lbr-dec.lib"
-l"dts-parma-dec.lib"
-l"dtsx-c-decoder.lib"
-l"la-strm-reconstruction.lib"
-l"dts-alg.lib"
```

4. Add the path to PARMA header files in CCS Project Directories
   For Example:

```
"${PROC_AUDIO_SDK_ROOT}/3p-ip-dts/dtsx-ip/asp/rel/parma/alg"
"${PROC_AUDIO_SDK_ROOT}/3p-ip-dts/dtsx-ip/asp/rel/parma/alpha"
"${PROC_AUDIO_SDK_ROOT}/3p-ip-dts/dtsx-ip/dec/rel/dtsx/alg"
```

```
"${PROC_AUDIO_SDK_ROOT}/3p-ip-dts/dtsx-ip/dec/rel/dtsx/alpha"
```