Processor Audio SDK Setup Guide

Texas Instruments

Rev. 0.10

23 Aug 2016

Revision History			
Version	**Date**	**Description of changes**	**Author(s)**
0.10	23-Aug-16	Initial version	Frank Livingston

Table of Contents

1 Introduction

This document provides installation and setup instructions for the alpha release of Processor Audio SDK for K2G (PASDK 1.0) provided by Texas Instruments, Inc. (TI). PASDK 1.0 is an Open Source release, and contains no 3rd party (e.g. Dolby or DTS) IP. The features of the release include:

- I-topology.
- PCM decoder (executing on ARM A15).
- PCM encoder.
- FIL, DEL, and VOL ASP's.
- AE0 ASP, disabled by default.
- Real-time input via 8-channel ADC and S/PDIF (optical only).
- Real-time output via 8-channel DAC.
- Real-time control I/O via UART.
- Initial framework allowing decoding to be performed on ARM.

The software executes on the K2G Rev. D EVM (SPRW274) and K2G EVM Audio Daughter Card Rev. B (SPRW279).

2 Software Installation

2.1 Code Composer Studio (CCS)

PASDK 1.0 was developed using CCS 6.1.2.00015 on Windows. Download and install this version of CCS from the following URL:

http://processors.wiki.ti.com/index.php/Download_CCS#Code_Composer_Studio_Version_6_Downloads

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

- For “Processor Support”, select “Multi Core Processors (all)”
- For “Select Debug Probes”, select “TI XDS Debug Probe Support” and “Spectrum Digital Debug Probes and Boards”

PASDK 1.0 was developed using the code generation tools listed below. These tools are installed as part of CCS.

- GNU ARM Tools 4.8.0.2014q3-20140805
- C6000 v8 Compiler Tools: 8.1.0

2.2 Processor SDK-RTOS

PASDK 1.0 was developed using Processor SDK-RTOS 2.0.2.11 for K2G on Windows. Release information for this version of Processor SDK-RTOS is provided at the URL below, along with links to the installer:

http://software-dl.ti.com/processor-sdk-rtos/esd/K2G/02_00_02_11/index_FDS.html

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

- SYS/BIOS 6.45.01.29
- XDAIS 7.24.00.04
- XDC Tools 3.32.00.06

2.3 Processor Audio SDK

PASDK1.0 is distributed using a self-extracting executable. Follow the steps below to install the software., accept the click-wrap license agreement, and install the software to a t

1. Double click the exectuble 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 to c:\ti_temp in the Choose Destination Location window. Complete the installation.
4. The Processor SDK-RTOS 2.0.2.11 and PASDK 1.0 both install PDK K2G in a folder named “pdk_k2g_1_0_1”. The PASDK version is enhanced and should replace the version installed by Processor SDK-RTOS. Hence, move the “pdk_k2g_1_0_1” folder in c:\ti to another folder (e.g. c:\ti_bak).
5. IPC engineering release 3.43.00 is installed with PASDK 1.0 as .zip file. Unzip the contents of this .zip file into c:\ti. After unzipping, c:\ti should contain the folder “ipc_3_43_00_00_eng”, which should in turn contain various folders beginning with the “docs” folder.
6. Move the contents of c:\ti_temp to c:\ti. The temporary directory c:\ti_temp can be deleted.

2.4 CCS Update

Update the CCS installation as below:

1. Launch CCS and create a new workspace (e.g. c:\ti\processor_audio_sdk_1_00_00_00\pasdk\workspace_v6_1).
2. Allow installation of newly discovered products located in c:\ti.

- Ignore any messages about unsigned content
- Restart CCS when prompted

3. Update Keystone2 device support
    - Select Help→Installation Details from the CCS menu
    - Select Keystone2 device support and click Uninstall.
    - Restart CCS when prompted.
    - Select Help→Install New Software.
    - Select "--All Available Sites--" from the "Work with:" drop-down menu.
    - Uncheck "Show only the latest version of the available software".
    - Click the right arrow next to Keystone2 Device Support to display the Keystone2 device support version options.
    - Select the second version 1.1.5 Keystone2 device support option. Continue to and accept the license agreement.
    - Restart CCS when prompted.
4. Update Emulators
    - Select Help→Check for updates from the CCS menu
    - Update Spectrum Digital Emulators to 5.2.0.14
    - Update TI Emulators to 6.0.407.3
    - Restart CCS when prompted

2.5 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 ti\processor_audio_sdk_1_00_00_00\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 5.3.

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
   c:\ti\processor_audio_sdk_1_00_00_00\pasdk\setup_env.bat).
2. Obtain Python 2.7.6 from https://www.python.org/download/releases/2.7.6
   executable installer, python-2.7.6.amd64.msi.
   Install to c:\PA_Tools\Python27 (or change installation location in
   c:\ti\processor_audio_sdk_1_00_00_00\pasdk\setup_env.bat).

3 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 . . .
```

4 Software Build

4.1 PASDK OS Dev Libraries

The PASDK 1.0 ARM and DSP applications depend on several libraries located in the ti\processor_audio_sdk_1_00_00_00\pasdk\paf folder. Detailed instructions for building any of the libraries in this folder are contained in C:\ti\processor_audio_sdk_1_00_00_00\pasdk\paf\ReadMe.txt. However, batch files for building only the necessary libraries for the ARM/DSP applications are provided in C:\ti\processor_audio_sdk_1_00_00_00\pasdk. To build the required libraries:

1. Open a DOS prompt.
2. From the DOS prompt, execute:
```
> cd C:\ti\processor_audio_sdk_1_00_00_00\pasdk
> setup_env.bat
```

3. Modify the ARCH variable in c:\ti\processor_audio_sdk_1_00_00_00\pasdk\paf\pa\build\target.mk to select the c66x target.
4. From the DOS prompt, execute:
   ```
   > build_libs.bat DSP
   ```
5. Modify the ARCH variable in c:\ti\processor_audio_sdk_1_00_00_00\pasdk\paf\pa\build\target.mk to select the a15 target.
6. From the DOS prompt, execute:
   ```
   > build_libs.bat ARM
   ```

4.2 UART Low-Level Driver (LLD)

Control I/O via UART depends on the UART LLD supplied with the enhanced PDK K2G 1.0.1. To build the UART LLD library:

1. Open a DOS prompt
2. From the DOS prompt, execute:
   ```
   > cd C:\ti\pdk_k2g_1_0_1\packages
   > pdksetupenv.bat
   > cd ti\drv\uart
   > gmake all
   ```

4.3 K2G Platform Library

Configuration and control of the K2G EVM and Audio Daughter Card require the K2G platform library supplied with the enhanced PDK K2G 1.0.1. To build the K2G platform library:

1. If necessary, re-launch CCS using the workspace created above in Section 2.4.
2. Import the “platform_lib_evmk2g” CCS project into the CCS workspace. The project is located in C:\ti\pdk_k2g_1_0_1\packages\ti\platform\evmk2g\platform_lib.
3. Rebuild the project using the Debug build profile.

4.4 DSP Application

PASDK 1.0 does not support I/O switching, and the DSP application must be rebuilt for each desired I/O configuration. The I/O configuration is determined by the at-boot shortcuts in the CUS_ATBOOT_S definition in c:\ti\processor_audio_sdk_1_00_00_00\pasdk\test_dsp\application\itopo\evmk2g\atboot.c. The table below shows the I/O shortcut required in CUS_ATBOOT_S for a particular I/O configuration.

I/O Configuration	Input Shortcut	Output Shortcut
ADC (8-ch) / DAC (8-ch)	execPAIInAnalog	execPAIOutAnalogSlave
S/PDIF / DAC (8-ch)	execPAIInDigital	execPAIOutAnalog

To build the DSP application:

1. Import the “test_dsp” CCS project into the CCS workspace. The project is located in C:\ti\processor_audio_sdk_1_00_00_00\pasdk\test_dsp.

2. Ensure the correct component software versions are selected by RTSC.
    - Right-click “test_dsp” project in CCS Project Explorer, select Properties
    - On left-hand side of window, Select General. Select RTSC tab on the right-hand side of General window.
    - Confirm the following are installed and selected: (1) EDMA3 Low Level Driver 2.12.1; (2) SYS/BIOS 6.45.1.29; (3) System Analyzer (UIA Target) 2.0.3.43; (4) XDAIS 7.24.0.04; and (5) k2g PDK 1.0.1
    - For IPC, select version 3.42.0.02 and then (re-)select version 3.43.0.00_eng.
    - Click OK.
3. Rebuild the project using the Debug build profile. Right-click “test_dsp” project in CCS Project Explorer, select Rebuild Project.

4.5 ARM Application

To build the ARM application:

1. Import the “test_arm” CCS project into the CCS workspace. The project is located in C:\ti\processor_audio_sdk_1_00_00_00\pasdk\test_arm.
2. Ensure the correct component software versions are selected by RTSC.
    - Right-click “test_arm” project in CCS Project Explorer, select Properties
    - On left-hand side of window, Select “General”. Select “RTSC” tab on the right-hand side of “General” window.
    - Confirm the following are installed and selected: (1) SYS/BIOS 6.45.1.29; (2) System Analyzer (UIA Target) 2.0.3.43; and (3) XDAIS 7.24.0.04.
    - For IPC, select version 3.42.0.02 and then (re-)select version 3.43.0.00_eng.
    - Click “OK”.
3. Rebuild the project using the Debug build profile. Right-click “test_arm” project in CCS Project Explorer, select Rebuild Project.

5 Application Execution and Test

5.1 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 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 CP2105_EN switch to the “ON” position.
5. Connect the EVM emulation port to the PC as described in Section 3.0
6. Re-connect the 12V power supply to the EVM. Turn on power to the EVM using SW1.

5.1.1 ADC Input and DAC 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.

5.1.2 S/PDIF Input and DAC Output

Connect S/PDIF inputs 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.

5.2 Load and Execute Code

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

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 MSMC SHM heap.

5.3 PyAlpha Python Scripts for Alpha Communication

PASDK1.0 supports alpha command control over UART using the PyAlpha tool located in c:\ti\processor_audio_sdk_1_00_00_00\tools\pyalpha. 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).

```
> cd c:\ti\processor_audio_sdk_1_00_00_00\tools\pyalpha
> python pyalpha –help
usage: PyAlpha [-?] -h FILE [-I DIR] [-p PORT] [-b BAUD] [-t DIR] [-r]

               [--fast]
               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: .]
  -r, --retain          Retain temporary files
  --fast                Skip post-processing
```

5.4 Alpha Commands for AE0 Configuration/Status

As previously mentioned, PASDK 1.0 includes the AE0 ASP in the signal chain. This ASP transforms stereo (Left/Right) PCM into 3stereo (Left/Right/Center) PCM. AE0 documentation is provided in the PASDK 1.0 release in “DA10x_UG_AspExample.pdf”. The alpha commands contained in C:\ti\processor_audio_sdk_1_00_00_00\tools\alpha\ae_a.h can be used to configure AE0 and consult AE0 status. In the alpha command sequence below, AE0 is configured to generate 3stereo using a gain of 0.5 (16384 in S16Q15 format) using COM port 1.

```
> cd c:\ti\processor_audio_sdk_1_00_00_00\tools\pyalpha
> python.exe pyalpha -h pa_i13_evmda830_io_a -p COM1 readAEMode
> alpha 0xfa00,0x0400
> python.exe pyalpha -h pa_i13_evmda830_io_a -p COM1
writeAEModeEnable
alpha /* none */
> python.exe pyalpha -h pa_i13_evmda830_io_a -p COM1 readAEMode
alpha 0xfa00,0x0401
> python.exe pyalpha -h pa_i13_evmda830_io_a -p COM1 readAEScaleQ15
alpha 0xfb00,0x0006,0x0000
> python.exe pyalpha -h pa_i13_evmda830_io_a -p COM1
writeAEScaleQ15(16384)
alpha
> python.exe pyalpha -h pa_i13_evmda830_io_a -p COM1 readAEScaleQ15
alpha 0xfb00,0x0006,0x4000
> python.exe pyalpha -h pa_i13_evmda830_io_a -p COM1
readSYSChannelConfigurationRequest
alpha 0xce20,0x2008,0x010c,0x0000,0x0000,0x0000
> python.exe pyalpha -h pa_i13_evmda830_io_a -p COM1
writeSYSChannelConfigurationRequestStereo
alpha /* none */
```

```
> python.exe pyalpha -h pa_i13_evmda830_io_a -p COM1
readSYSChannelConfigurationRequest
alpha 0xce20,0x2008,0x0003,0x0000,0x0000,0x0000
> python.exe pyalpha -h pa_i13_evmda830_io_a -p COM1
writeSYSChannelConfigurationRequestSurround4_1
alpha /* none */
> python.exe pyalpha -h pa_i13_evmda830_io_a -p COM1
readSYSChannelConfigurationRequest
alpha 0xce20,0x2008,0x010c,0x0000,0x0000,0x0000
> python.exe pyalpha -h pa_i13_evmda830_io_a -p COM1
writeAEModeDisable
alpha 0xfa00,0x0400
python.exe pyalpha -h pa_i13_evmda830_io_a -p COM1 readAEMode
```