Creating and Using Platform for an Application

Introduction

This lab guides you through the steps of creating a custom platform for an audio application.

Objectives

After completing this lab, you will be able to:

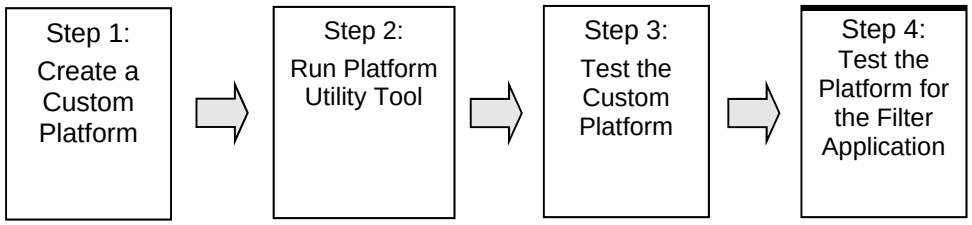
- Create an SDx platform for an custom application
- Use the SDx environment to test the platform for an audio filtering Standalone application

Procedure

This lab is separated into steps that consist of general overview statements that provide information on the detailed instructions that follow. Follow these detailed instructions to progress through the lab.

This lab comprises three primary steps: You will create an SDx project, test the custom platform, and test the platform for the filter application.

General Flow for this Lab

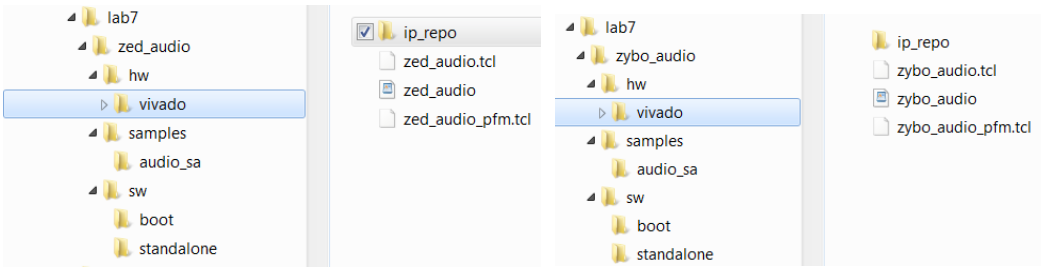


Create a Custom Platform Step 1

1-1. Launch Vivado, create the platform design, generate the output products and export the hardware.

1-1-1. Using the Windows Explorer, copy the **zed_audio** directory (for Zed) or **zybo_audio** directory (for Zybo) from the *source\lab7* directory and place it in the *c:\xup\SDSoC\labs\lab7* directory.

This will copy all the necessary directories and files, creating the required directory structure.



(a) Zed **(b) Zybo**

Figure 1. The directory structure for creating the SDx platform

1-1-2. Open Vivado by selecting **Start > All Programs > Xilinx Design Tools > SDx 2017.2 > Vivado Design Suite > Vivado 2017.2**

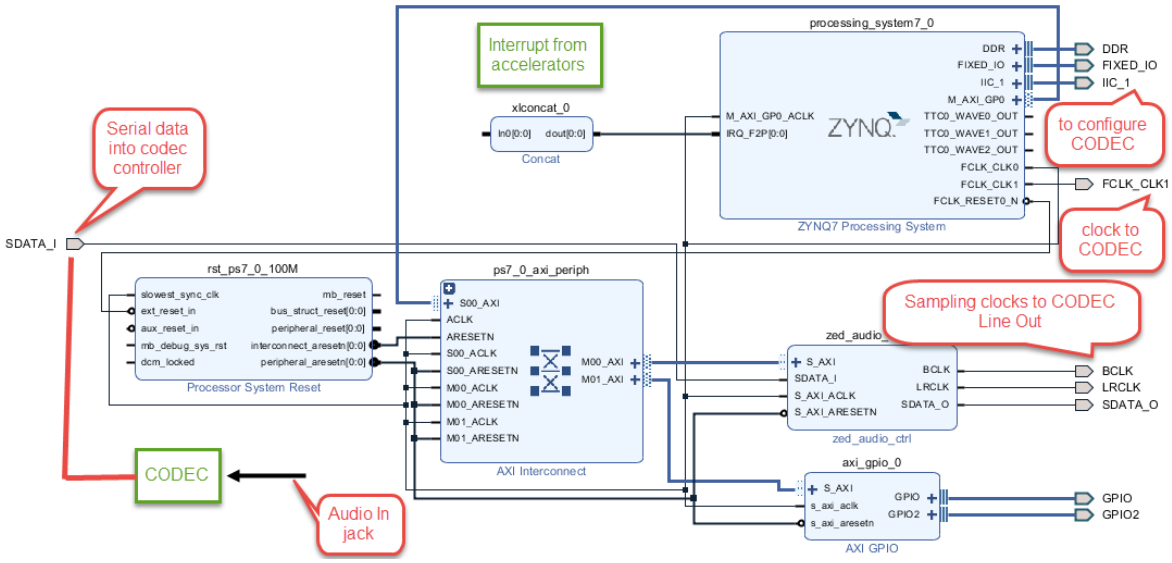
1-1-3. In the Vivado's Tcl Console window change the directory to the *c:/xup/SDSoC/labs/lab7/zed_audio/hw/vivado/* or *c:/xup/SDSoC/labs/lab7/zybo_audio/hw/vivado/* using the **cd** command.

```
cd c:/xup/SDSoC/labs/lab7/<zed | zybo>_audio/hw/vivado
```

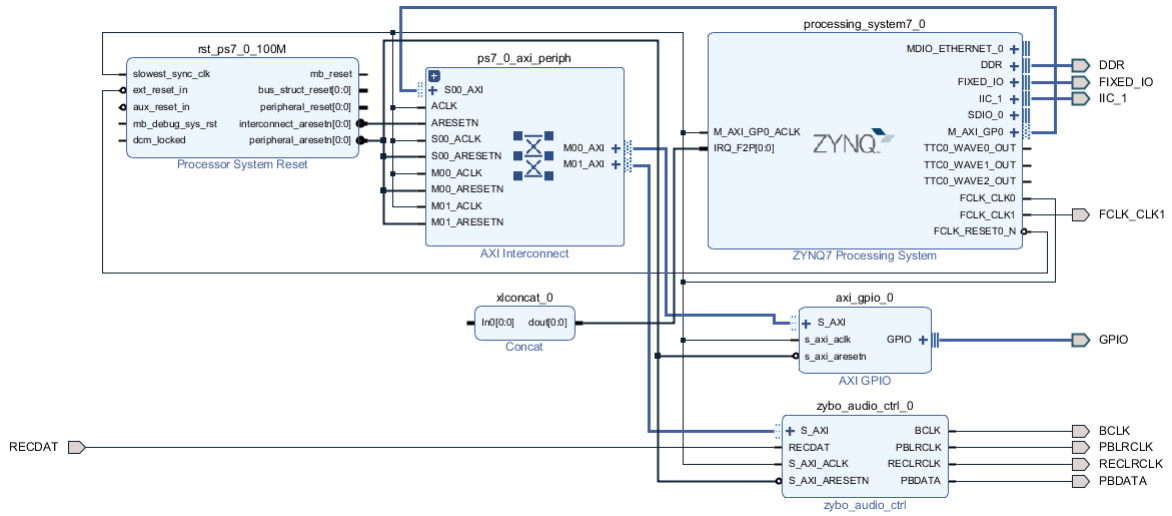
1-1-4. Execute the following command to generate the platform hardware.

```
source ./<zed | zybo>_audio.tcl
```

This will create an IPI design and an HDL wrapper, and add an xdc constraints file.



(a) Zed



(b) Zybo

Figure 2. The IPI design

1-1-5. Select **Sources > IP Sources** tab and select *<zed | zybo>_audio*.

1-1-6. Right-click and select **Generate Output Products**.

1-1-7. Click **Generate**.

Wait for approximately 7-8 minutes for the output products to be generated. You can see the status in the **Design Runs** tab.

1-1-8. Once the output products are generated, go to **File** > **Export** > **Export Hardware** to export the hardware. Make sure **Include Bitstream** is *unchecked*.

The *<zed | zybo>_audio.sdk* directory will be created under the *\hw\vivado* directory.

1-1-9. Select **File** > **Exit** to close the Vivado tool.

1-2. Build the software templates

1-2-1. Open SDx by selecting **Start > All Programs > Xilinx Design Tools > SDx 2017.2 > SDx IDE 2017.2**

The workspace dialog box will open.

1-2-2. Click on the **Browse** button, select the *c:\xup\SDSoC\labs\lab7\zed_audio\hw\vivado\zed_audio.sdk* (for zed) OR *c:\xup\SDSoC\labs\lab7\zybo_audio\hw\vivado\zybo_audio.sdk* (for zybo) directory and click **OK**.

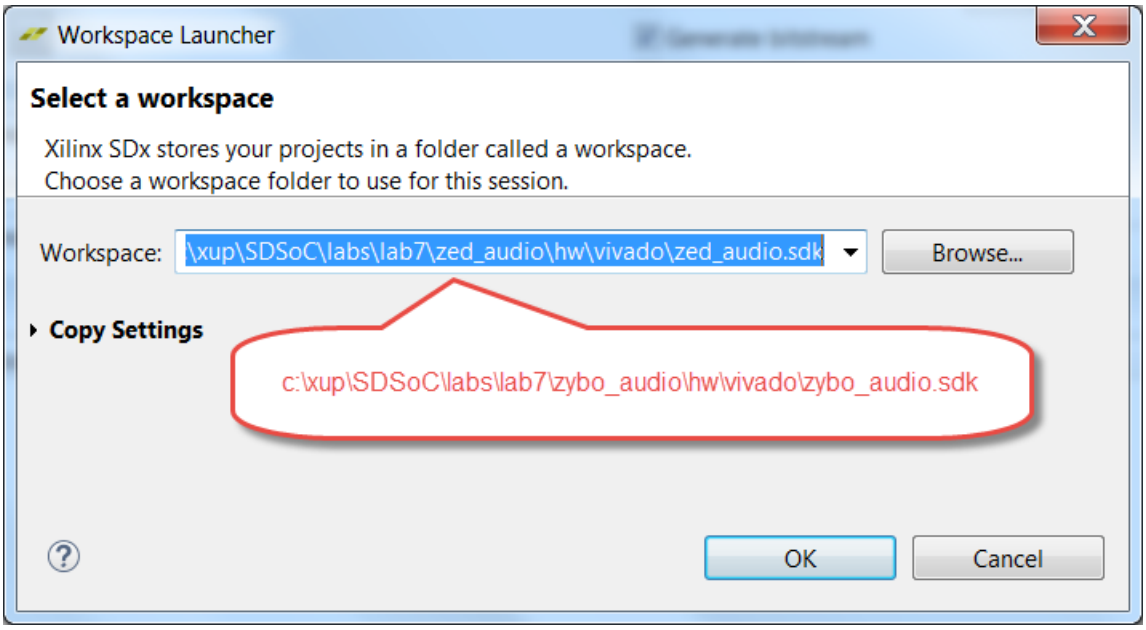


Figure 3. Selecting the workspace

1-2-3. Close the **Welcome** Page.

1-2-4. Select **File > New > Project,** then expand Xilinx, select *Hardware Platform Specification* and click **Next**.

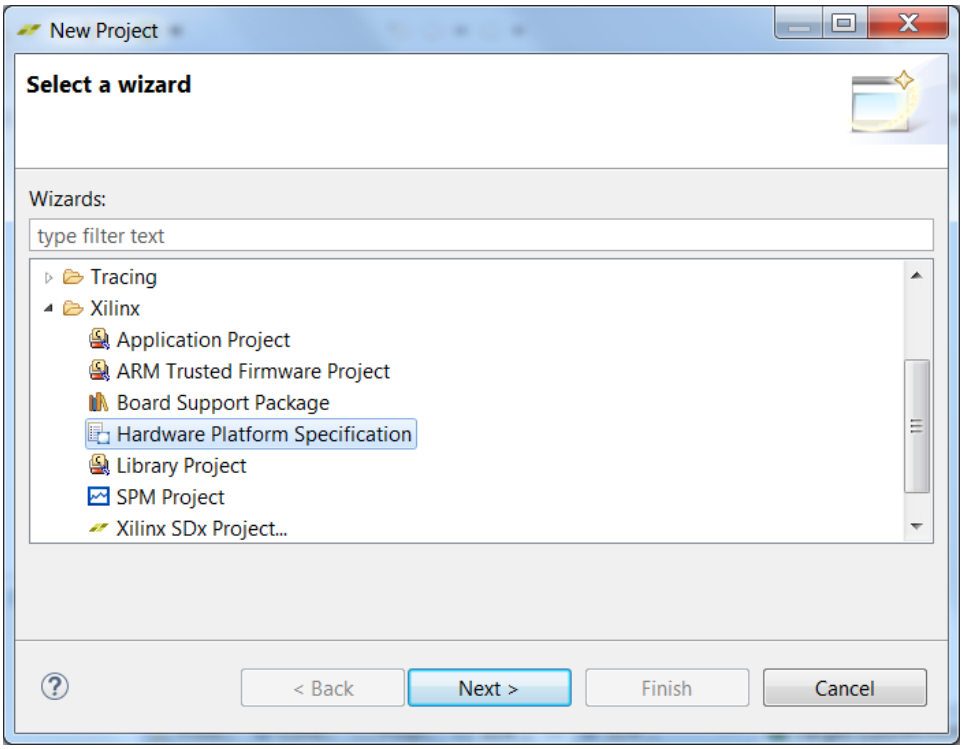


Figure 4. Creating the hardware platform specification project

The hardware platform specification describes the hardware design. This includes a full system memory map, the type(s) of processors present, active peripherals in the PS and PL for Zynq systems or a list of all peripherals for a non-Zynq systems.

Based on this description, software such as the board support package (BSP) and application can be tailored to the hardware

1-2-5. Enter **zed_audio** or **zybo_audio** as the *project name*, click on the browse button of *Target Hardware Specification* and browse to *c:\xup\SDSoC\labs\lab7\<zed | zybo>_audio\hw\vivado\<zed | zybo>_audio.sdk*, select *<zed | zybo>_audio_wrapper.hdf*, click **Open**, and then click **Finish**.

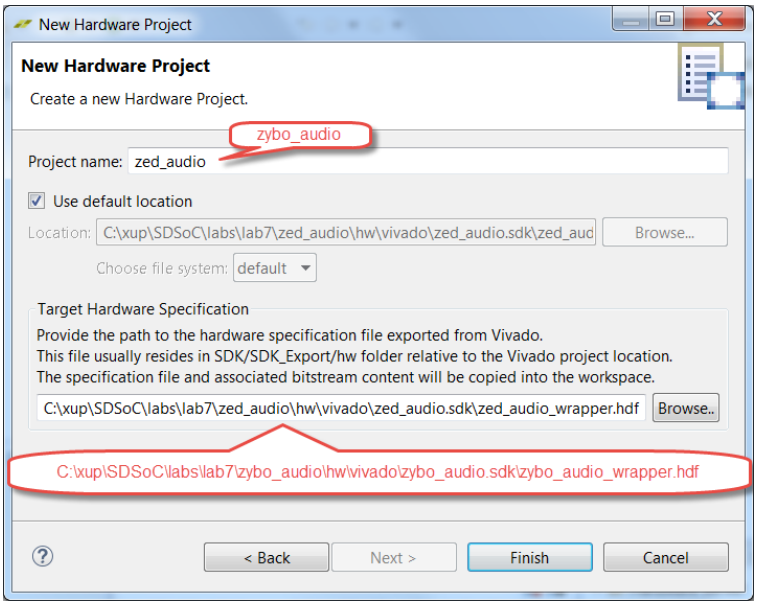


Figure 5. Creating hardware project in SDx

1-2-6. Select **File > New > Project**, then expand **Xilinx** and select **Board Support Package** and click **Next**.

1-2-7. Click **Finish** with *standalone_bsp_0* as the *Project name*, making sure that *zed_audio* or *zybo_audio* is selected as the *Hardware Platform*.

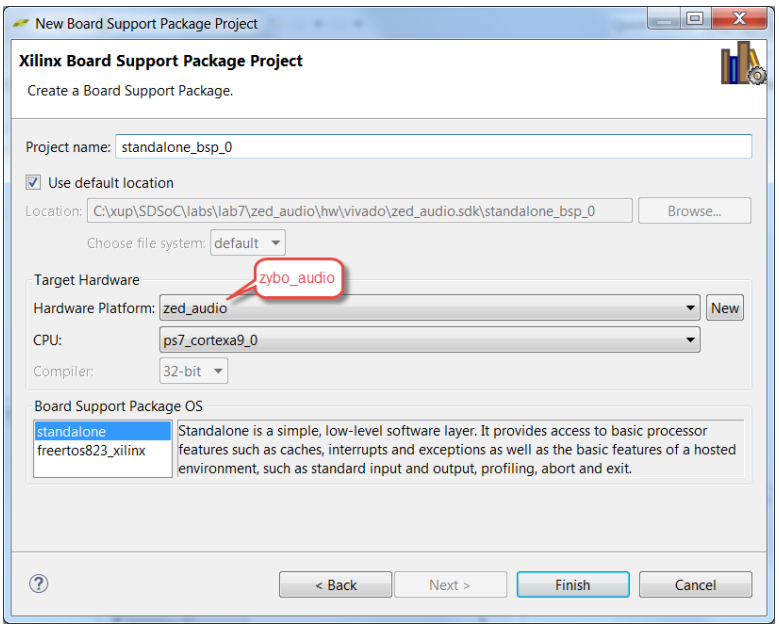


Figure 6. Creating the board support package for the platform

The *Board Support Package Settings* will open.

1-2-8. Select *xilffs* library and click **OK**.

1-2-9. Right-click on the *standalone_bsp_0* and select **build project**.

1-3. Generate the FSBL application so the board can be boot from the SD card.

1-3-1. Select **File> New > Application Project**

1-3-2. Enter **fsbl** in the *Project name* field.

1-3-3. For the Board Support Package, select **Create New** and click **Next.**

1-3-4. Select **Zynq FSBL** from the *Available Templates* pane, and click **Finish.**

1-3-5. Click **Yes** to open the C/C++ perspective.

1-3-6. Right-click on the *fsbl* entry and select **build project.**

The *fsbl.elf* file is required to create a bootable SD image.

The fsbl application does not include DDR as one of the target memories. The actual application typically does target DDR memory. Though your application may not be using alloc/calloc/malloc calls, which uses heap memory, the relatively larger memory is required for holding scheduling information and for event tracking structures during the booting up process.

There are three options you can choose from:

1. Create a dummy hello_world application, and change its heap size to 64 MB and use that linker script while creating custom platform in Step 2.
2. Use the fsbl application linker script during the custom platform creation and then manually edit the linker script of the generated custom platform (<custom_platform\sw\config0_0) as instructed in **1-5** below.
3. Use the provided linker script (lscrip.ld) file provided in the source\lab7\zed_audio\sw\standalone

1-4. <u>Option 1:</u> Generate a dummy custom hello_world application, generate its linker script with heap size changed to 64 MB. Make sure that DDR memory is selected for the stack/heap.

1-4-1. Select **File> New > Application Project**

1-4-2. Enter **hello_world** in the *Project name* field.

1-4-3. For the Board Support Package, select **fsbl_bsp** in the *Use existing* option, and click **Next.**

1-4-4. Select **Hello World** from the *Available Templates* pane, and click **Finish.**

1-4-5. Right-click on the *hello_world* application and select **Generate linker script**

1-4-6. Change the heap size to 64 MB (**67108864)**, making sure that *ps7_ddr_0* is the target memory for the Heap and Stack.

1-4-7. Click **Generate** to generate the script. Click **Yes** to overwrite.

There is no need to build this project as we really need the lscript.ld file.

1-5. Option 2: Edit the lscript.ld file generated for the fsbl application to include DDR memory as target, change the size of the HEAP to 0x4000000 (64 MB) and target the .stack and .heap sections to the DDR memory.

1-5-1. Using a text editor open the **lscript.ld** file under the *C:\xup\SDSoC\labs\lab7\zed_audio\hw\vivado\zed_audio.sdk\fsbl\src* or for zybo *C:\xup\SDSoC\labs\lab7\zybo_audio\hw\vivado\zybo_audio.sdk\fsbl\src* directory**.**

1-5-2. Change the *_HEAP_SIZE* from **0x2000** to **0x400000**

1-5-3. Add the following line in the MEMORY section (around line 47)

```
ps7_ddr_0_S_AXI_BASEADDR : ORIGIN = 0x00100000, LENGTH = 0x3FF00000
```

1-5-4. Change target memory for the *.heap* and *.stack* sections from **ps7_ram_0_S_AXI_BASEADDR** to **ps7_ddr_0_S_AXI_BASEADDR**

1-5-5. Save the changes and close the file.

Invoking the SDSoC Platform Utility Tool Step 2

2-1. Launch the SDx terminal to generate the required metadata files. Create lab7_plaform directory under c:\xup\SDSoC\labs.

In this step, you will use the SDSoC Platform Utility tool to create hardware (.dsa), software (.spfm) and the top level (.xpfm) files.

The XPFM (*zed_audio.xpfm*) file will contain references to the hardware (*zed_audio.dsa*) and software (*zed_audio.spfm*) metadata files. All path references in the *.spfm* are relative to the folder containing that file.

2-1-1. Using the Windows Explorer, create an empty directory **lab7_platform** under *c:\xup\SDSoC\labs*.

2-1-2. Select **Start** > **All Programs** > **Xilinx Design Tools** > **SDx 2017.2** > **SDx Terminal 2017.2**

2-1-3. Enter the following command in the SDx Terminal to change directory to c:*xup\SDSoC\labs\lab7*

```
cd c:\xup\SDSoC\labs\lab7
```

2-1-4. Enter the following command to start the SDSoC Platform Utility tool:

```
sdspfm -gui
```

2-1-5. Enter the following information:

Platform Vendor: **Xilinx Inc** (you can enter any name)

Platform Name: **zed_audio** or **zybo_audio** depending on the target board

Platform Version: *Major*: **1** *Minor*: **0** you can put any meaningful numbers

Platform Description: Enter some meaningful description

2-1-6. Browse to various directories and select:

Output Directory: **c:\xup\SDSoC\labs\lab7_platform** (create this directory)

Vivado Project: **C:\xup\SDSoC\labs\lab7\zed_audio\hw\vivado\zed_audio.xpr** for zybo choose **C:\xup\SDSoC\labs\lab7\zybo_audio\hw\vivado\zybo_audio.xpr**

Note that *Top BD* field will be automatically populated

Platform Tcl: **C:\xup\SDSoC\labs\lab7\zed_audio\hw\vivado\zed_audio_pfm.tcl** for zybo choose **C:\xup\SDSoC\labs\lab7\zybo_audio\hw\vivado\zybo_audio_pfm.tcl**

Sampled Directory: **C:\xup\SDSoC\labs\lab7\zed_audio\samples** for zybo choose **C:\xup\SDSoC\labs\lab7\zybo_audio\samples**

2-1-7. Make sure that *ARM Cortex-A9* is shown in **Type** field of the *Processor Information* section. Click the **Add** button.

2-1-8. Click the **Add** button of the *Supported OSes* in the **Boot Information** section. Note that the **Config ID** and **Config Name** fields are auto populated.

2-1-9. Browse to various directories and select:

BIF File: **C:\xup\SDSoC\labs\lab7\zed_audio\sw\boot\standalone.bif** or for zybo **C:\xup\SDSoC\labs\lab7\zybo_audio\sw\boot\standalone.bif**

Readme File: **C:\xup\SDSoC\labs\lab7\zed_audio\sw\boot\generic.readme** for zybo choose **C:\xup\SDSoC\labs\lab7\zybo_audio\sw\boot\generic.readme**

Boot directory: **C:\xup\SDSoC\labs\lab7\zed_audio\hw\vivado\zed_audio.sdk\fsbl\Debug** for zybo choose **C:\xup\SDSoC\labs\lab7\zed_audio\hw\vivado\zybo_audio.sdk\fsbl\Debug**

Linker Script: Depending on the option you have decided in Step 1-4/5 select lscript.ld from the appropriate path

Option 1:
C:\xup\SDSoC\labs\lab7\zed_audio\hw\vivado\zed_audio.sdk\hello_world\src\lscript.ld for zybo choose **C:\xup\SDSoC\labs\lab7\zybo_audio\hw\vivado\zybo_audio.sdk\ hello_world \src\lscript.ld**

Option 2:

C:\xup\SDSoC\labs\lab7\zed_audio\hw\vivado\zed_audio.sdk\fsbl\src\lscript.ld for zybo choose **C:\xup\SDSoC\labs\lab7\zybo_audio\hw\vivado\zybo_audio.sdk\fsbl\src\lscript.ld**

Option 3:

C:\xup\SDSoC\labs\lab7\zed_audio\sw\standalone\lscript.ld for zybo choose **C:\xup\SDSoC\labs\lab7\zybo_audio\sw\standalone\lscript.ld**

2-1-10. In the **Include Path** section, click on the Browse button, then browse to **C:\xup\SDSoC\labs\lab7\zed_audio\hw\vivado\zed_audio.sdk\standalone_bsp_0\ps7_cortexa9_0\include** for zybo select **C:\xup\SDSoC\labs\lab7\zybo_audio\hw\vivado\zybo_audio.sdk\standalone_bsp_0\ps7_cortexa9_0\include**

2-1-11. Click the **Add** button.

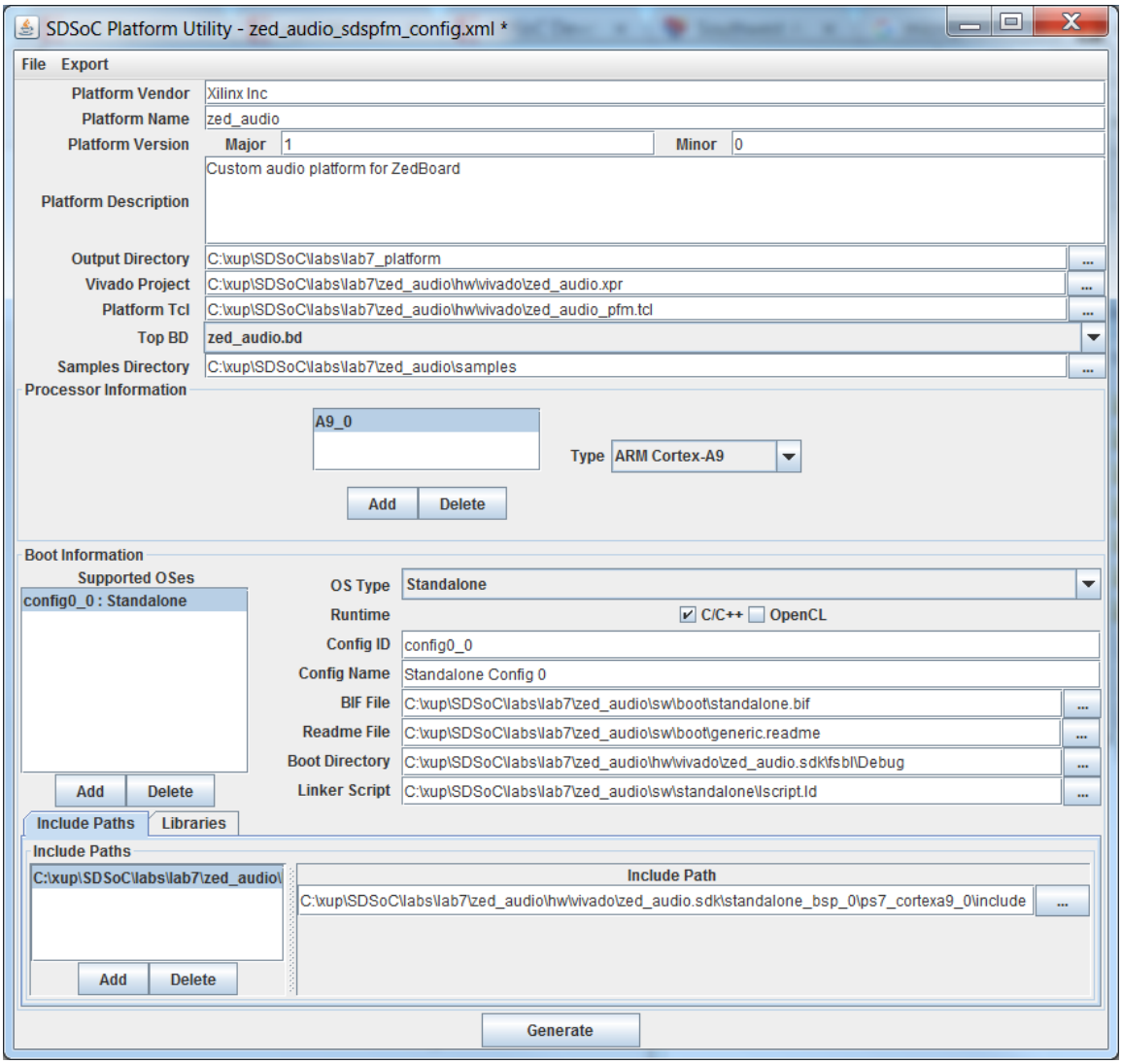


Figure 7. SDSoC Platform Utility

2-1-12. Click **Generate**.

View generation progress in the SDx Terminal window. The whole process may take about 5 minutes.

2-1-13. When generation is completed successfully, click **File > Save As**, browse to *c:\xup\SDSoC\lab7*, enter <zed | zybo>**_audio_sdspfm_config** in the *File Name* field, and click **Save**.

You can use the saved configuration file if you want to make changes in future.

2-1-14. Click **File > Exit**

2-1-15. Close the SDx Terminal window.

Test the Built Platform Step 3

3-1. In SDx change the workspace to c:\xup\SDSoC\labs\lab7. Create a new SDx project called *audio_test* using zed_audio or zybo_audio as the platform and Standalone as the OS, and selecting Audio Playback template provided in the *samples* directory.

3-1-1. In SDx change the workspace to *c:\xup\SDSoC\labs\lab7* by selecting **File > Switch Workspace > other**.

3-1-2. Click **OK**.

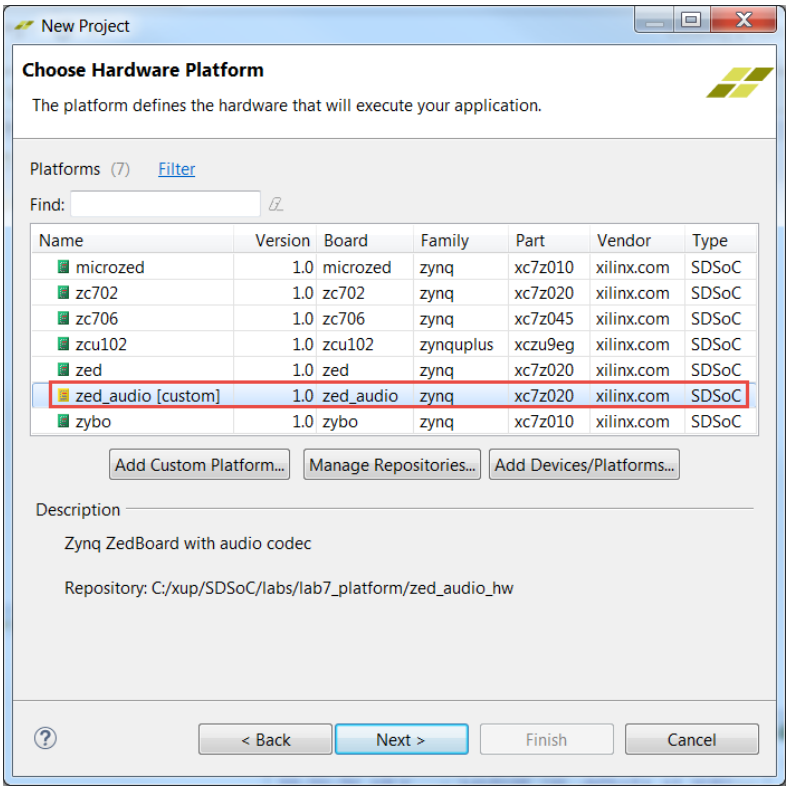
3-1-3. Close the **Welcome** page.

3-1-4. Select **File > New > Xilinx SDx Project…**

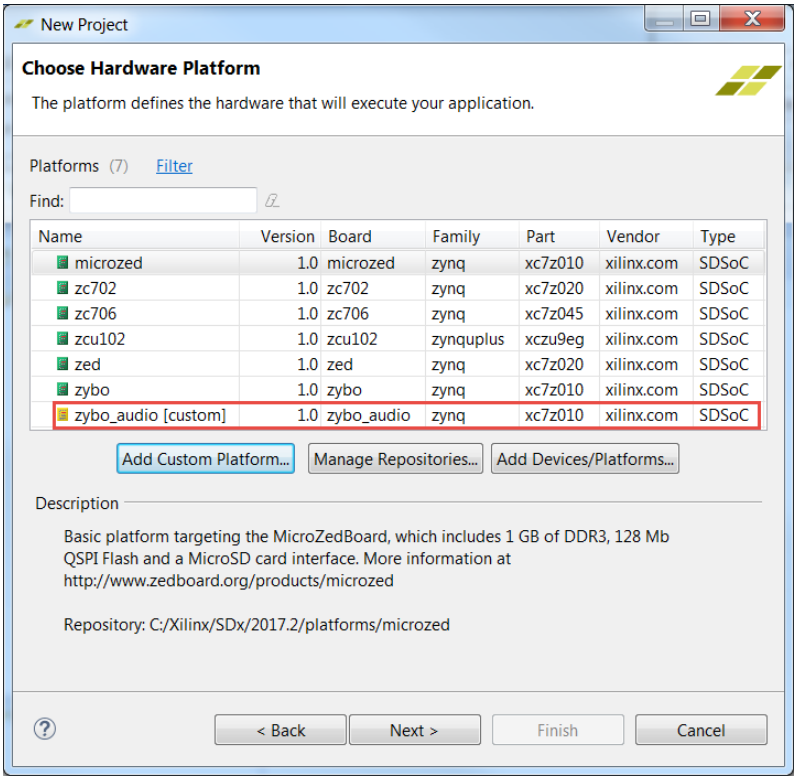
3-1-5. Enter **audio_test** in the *Project name* field and click **Next**.

3-1-6. Click **Add Custom Platform…,** browse to *c:\xup\SDSoC\labs\lab7_platform* and select either **zed_audio** or **zybo_audio** and click **OK**.

The custom platform entry will appear in the available.



(a) Zed



(b) Zybo

Figure 8. Selecting and adding the custom platform

3-1-7. Select *zed_audio* or *zybo_audio* and click **Next**.

3-1-8. For the *OS*, select **Standalone Config 0**.

3-1-9. Click **Next**.

The Templates window will be displayed with Audio Playback as one of the two possible templates. This entry is picked up from the samples directory of the created platform.

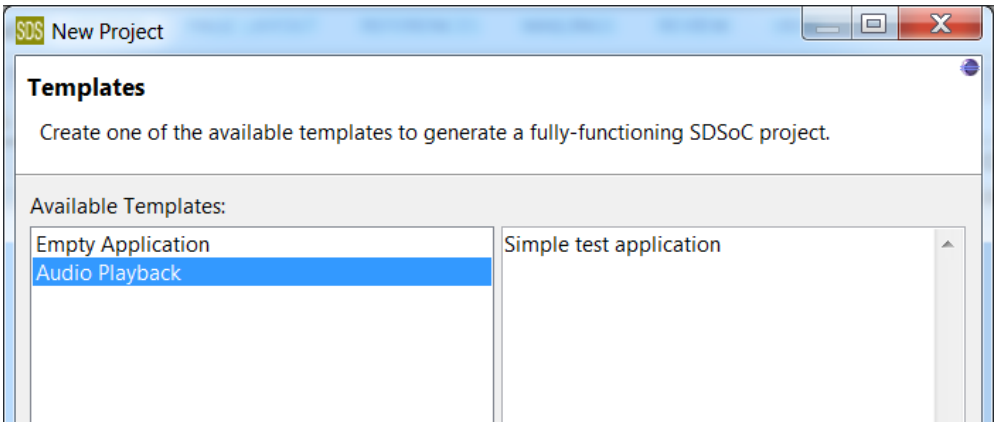


Figure 9. Selecting a test template

3-1-10. Select the *Audio Playback* application and click **Finish**.

3-1-11. Expand the *audio_test* entry in the Project Explorer pane and note the two source files (audio.h and audio.c) make up the application.

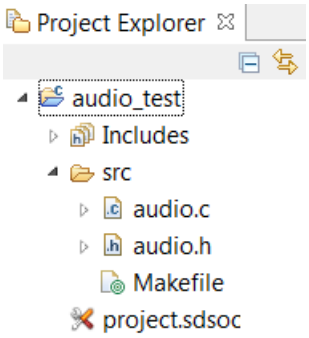


Figure 10. Test application directory

The *audio.c* application configures the CODEC, samples the CODEC and writes back into the CODEC illustrating the platform does function.

3-1-12. Uncheck Generate SD Card Image box as we will test it using JTAG mode.

3-1-13. Right-click on **audio_test** in the **Project Explorer** and select *C/C++ Build Settings*. Select **Miscellaneous** under *SDS++ Linker*, click "+" button of the *Other Options* and enter **–maxjobs <host core count>/2** (substitute <host core count> with the actual number of cores of your machine) and click **OK**

3-1-14. Right-click on the *audio_test* entry and select **Build Project**.

3-2. Connect the board and test the application.

3-2-1. Connect an audio patch cable between the PC's headphone output and Line-In connector of the board.

3-2-2. Connect a headphone to the Line Out connector (on Zed) or HPH OUT connector (on Zybo) of the board.

3-2-3. Connect the board and power it ON.

3-2-4. Right-click on the *audio_test* folder and select **Run As > Launch on Hardware (SDSoC Debugger)** to run the application.

This will download the bit file to configure the FPGA, download the *audio_test.elf* application, and run the application.

3-2-5. Play some music on the PC and you should be able to hear the same on your headphone.

3-2-6. When satisfied, power OFF the board.

Test the Platform for the Filter Application Step 3

4-1. Create a fir_test application targeting the custom platform and Standalone OS. Import the provided audio.h, audio.c, and fir_test.c files from the c:\xup\SDSoC\source\lab7 folder.

4-1-1. Select **File > New > Xilinx SDx Project**

4-1-2. Enter **fir_test** in the *Project name* field and click **Next**.

4-1-3. Select either **zed_audio** or **zybo_audio** and click **Next**.

4-1-4. Select **Standalone** as the *Target OS* and click **Next**.

4-1-5. Select *Empty Application* template and click **Finish**.

4-1-6. Expand **fir_test > src**, right-click on the *src* folder and select **Import…**

4-1-7. Expand General, select File System and click **Next**

4-1-8. Browse to *c:\xup\SDSoC\source\lab7*, and import **audio.c, audio.h, fir_coef.dat** and **fir_test.c** files.

4-2. Add the fir function to be accelerated. Set the build option to use half of CPU cores and build the project.

4-2-1. Add the **fir** function in the *HW Function* panel.

4-2-2. Right-click on **fir_test** in the **Project Explorer** and select *C/C++ Build Settings*. Select **Miscellaneous** under *SDS++ Linker*, click "+" button of the *Other Options* and enter **–maxjobs <host core count>/2** (substitute <host core count> with the actual number of cores of your machine) and click **OK**

4-2-3. Right-click on the **fir_test** entry in the *Project Explorer* panel, and select **Build Project**.

This will take about 25 minutes.

4-3. Connect the board and test the application.

4-3-1. Connect an audio patch cable between the PC's headphone output and Line-In connector of the board.

4-3-2. Connect a headphone to the Line Out connector (on Zed) or HPH OUT connector (on Zybo) of the board.

4-3-3. Connect the board and power it ON.

4-3-4. Right-click on the *fir_test* folder and select **Run As > Launch on Hardware (SDSoC Debugger)** to run the application.

This will download the bit file to configure the FPGA, download the *fir_test.elf* application, and run the application.

4-3-5. Play some music on the PC and you should be able to hear the same on your headphone.

4-3-6. When satisfied, turn OFF the board and exit the SDx program.

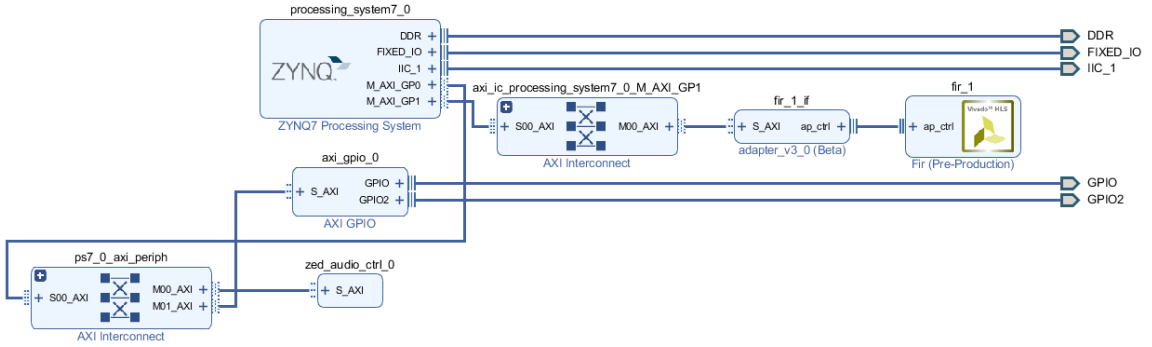
4-4. Open Vivado and view the built design.

4-4-1. Start Vivado by selecting **Start > All Programs > Xilinx Design Tools > SDx 2017.2 > Vivado Design Suite > Vivado 2017.2**

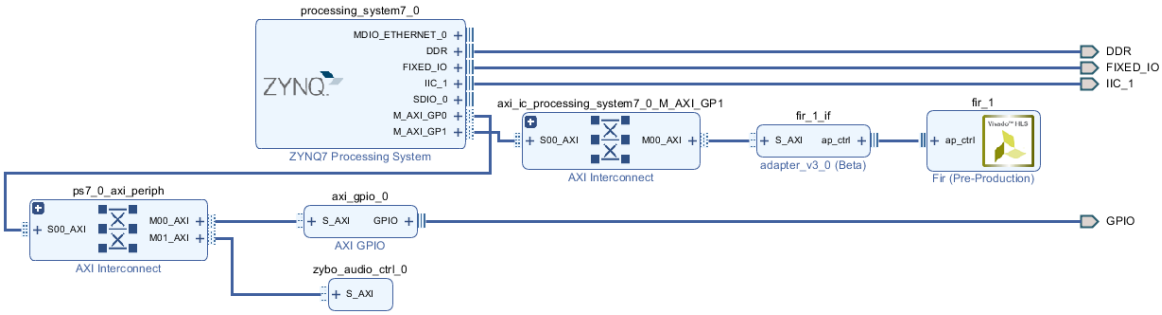
4-4-2. Click the **Open Project** link, open the design by browsing to *c:\xup\SDSoC\labs\lab7\fir_test\Debug_sds\p0_vpl/ipi* and selecting the **ipiprj.xpr**

4-4-3. Click on **Open Block Design** in the *Flow Navigator* pane. The block design will open.

4-4-4. Click on the **show interface connections only** () button followed by click on the **regenerate layout** () button.



(a) Zed



(b)

Figure 11. The generated block design

You can see *Fir* filter instance and the datamover adapter.

4-4-5. Close Vivado by selecting **File > Exit**

Conclusion

In this lab, you created a custom platform utilizing an audio CODEC IP in the base design. You then created a test application using the provided test template to test the custom platform. You then created a user application, importing the provided source files, targeted a *fir* function in hardware and then tested the application.