SDSoC Tool Overview Demo Script

Introduction

This demonstration provides high-level instructions for some of the key features of the SDSoC™ development environment, such as creating a project and importing sources, marking a function for hardware, running SDEstimate, and reviewing important aspects of a completed design. It is designed to be used prior to the "SDSoC Tool Overview" lecture module.

Preparation:

- Required files: Necessary files are located at *C:\training\hls\support\toolDemo*
- Required hardware: ZC702 or ZedBoard; no daughter cards are needed
- Supporting materials: None
- Demo supports both the ZC702 and ZedBoard in Standalone mode

SDSoC Tool Overview

Action with Description	Point of Emphasis and Key Takeaway
• Launch SDSoC development environment 2016.1.	• Launch the SDSoC tool and select the workspace directory.
• Select the workspace to be the *C:\training\hls\demos\toolDemo* directory.	• The SDSoC tool stores your projects in a folder called a workspace. • Choose a workspace folder to use for your project.

Action with Description	Point of Emphasis and Key Takeaway

• The SDSoC tool launches with the Welcome screen when the workspace is empty. When it is not empty, this process must be done manually.
• You will use the Welcome screen to launch the new project creation process.
• If the Welcome screen is closed, it can be re-opened using **Help** > **Welcome**.
	• The Getting Started page provides links to:
	  • Create an SDSoC tool project
	  • Import an existing project
	  • View the SDSoC tool user guide
	  • View the SDK tool user guide

• Click the **Create SDSoC Project** link in the Getting Started page. • If the Getting Started page is not available, select **File** > **New** > **SDSoC Project**.	• The New Project Wizard provides an easy way to quickly build a project.

Action with Description	Point of Emphasis and Key Takeaway

• Create a new project using the New Project Wizard:
  • Project name: **SDSoC_lab_design**
  • Target Platform: **ZC702 or ZedBoard**
  • OS: **Standalone**
	• Platform:
	  • A hardware platform includes the Zynq® All Programmable SoC PS, which contains dual ARM® Cortex-A9 CPUs and other hard IP, including a memory controller that provides access to external DDR.
	  • A platform will often include peripherals implemented in the Zynq AP SoC programmable logic (PL); e.g., to interface with DACs, ADCs, or HDMI I/O or to access external DDR through a MIG memory controller.
	  • A platform can also include additional functionality that is entirely independent of the logic generated by the SDSoC compiler (sdscc/sds++).
	• OS:
	  • You can choose **Standalone**, **Linux**, or **FreeRTOS**.

• Click **Next**.	

Action with Description	Point of Emphasis and Key Takeaway

• Select **Empty Application**.	• You can select existing available templates including:
	  • Matrix Multiplication
	  • Matrix Multiplication Data Size
	  • Matrix Multiplication and Addition
	  • Matrix Multiplication (area reduced)
	  • Synthesizable FIR Filter
	  • Others
	Note that the available templates that you see are dependent on the OS that was selected in the previous page.
	• You are selecting Empty Application in this demo in order to use custom design files.

• Click **Finish**.	• Explain the different SDSoC tool views such as (shown below):
	  • SDSoC Project Overview
	  • Project Explorer
	  • Outline
	  • Target Connections
	  • Problems, Console, Properties, and Terminal tabs

• Expand **SDSoC_lab_design** in the Project Explorer window.	• All the applications created in this workspace will be visible in the Project Explorer window.
• Right-click the **src** directory. • Select **Import** to open the Import dialog box.	• Add the design files by selecting the import option.

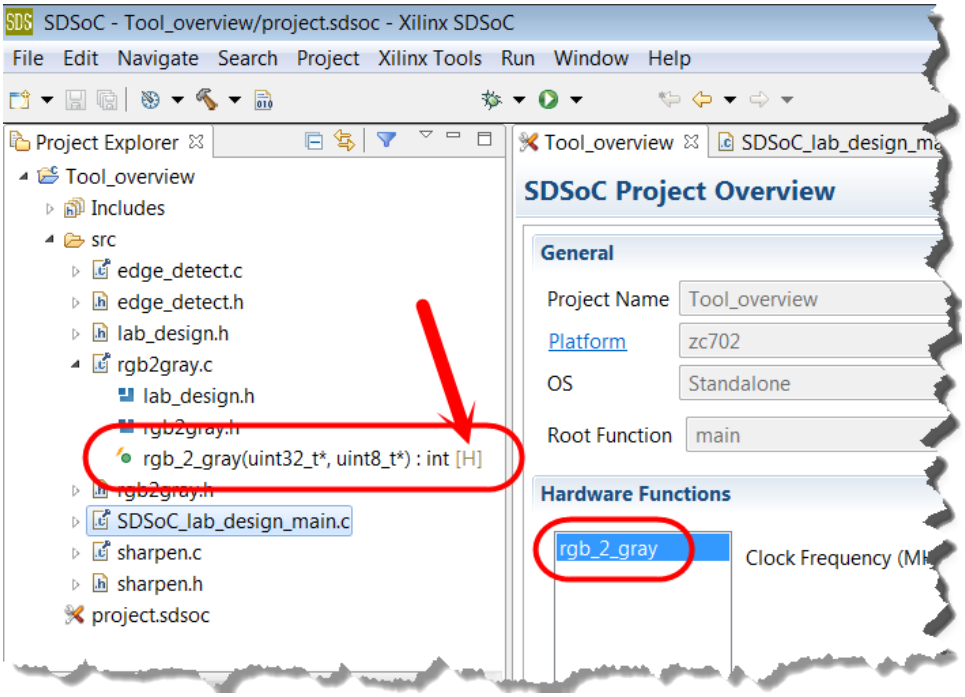
Action with Description	Point of Emphasis and Key Takeaway

• Expand **General** and select **File System**.	• Different options to import:
	  • Archive File: Import resources from an archive file into an existing project.
	  • Existing Projects into Workspace.
	  • File System: Import resources from the local file system into an existing project.

• Click **Next** and enter **C:/training/hls/support/toolDemo** in the *From directory* field. • Select only the **edge_detect.c** and **edge_detect.h** files for now. • Click **Finish**.	• Select the archive option to import the existing project.
• Use Windows Explorer, browse to *C:\training\hls\support\toolDemo*. • Select all the ***.c** and ***.h** files except the *edge_detect* files (because they have already been added). • Drag-and-drop the files under the *src* directory. • Select "copy files", which will make a local copy of the sources for the project.	• Select the existing project archive file.
• Expand **SDSoC_lab_design** > **src** in the Project Explorer window.	• Review the design files in the project.

Action with Description	Point of Emphasis and Key Takeaway
• Double-click **SDSoC_lab_design_main.c** to open it in the editor. Also enable line numbers. • Review the code structure, particularly the area near the comment "process image". The application consists of three computationally intensive functions: *rgb_2_gray*, *sharpen_filter*, and *sobel_filter*. You will begin by moving the *rgb_2_gray* function into hardware.	• Review the editor and enable line numbers. • Instrumentation will determine if the project meets performance specifications. • Profiling will show which functions consume the most CPU cycles.

• Select the **SDSoC_lab_design** tab and click the **Add Hardware Function** icon ( ) in the Hardware Functions section.	• The SDSoC Project Overview tab provides a central location for setting project values.
	  • Options:
	  ▪ Set the Data Motion Clock Frequency
	  ▪ Generate Bitstream
	  ▪ Generate SD Card
	  ▪ Insert AXI Performance Monitor
	  ▪ Enable Event Tracing
	  ▪ Estimate Performance
	  • Hardware functions:
	  ▪ Add the functions that need to be moved to hardware
	  ▪ Set the clock frequency

Action with Description	Point of Emphasis and Key Takeaway
• Select the **rgb_2_gray** function to select it in the Matching elements. The selected function appears in the "Qualified name and location" window. • Click **OK**.	• Select the function(s) that will be moved to hardware. • Multiple functions can be selected using standard Windows multiple selection techniques.
• The functions are visible in the SDSoC Project Overview as selections in the Hardware Functions lists and are marked in the Project Explorer view under the *source_file.function* name. That is, you would expand the C/C++ source file to see the header files that are included and the function in the source file. Here you will see that the *rgb_2_gray* function is marked to target hardware (the [H] annotation).	
	

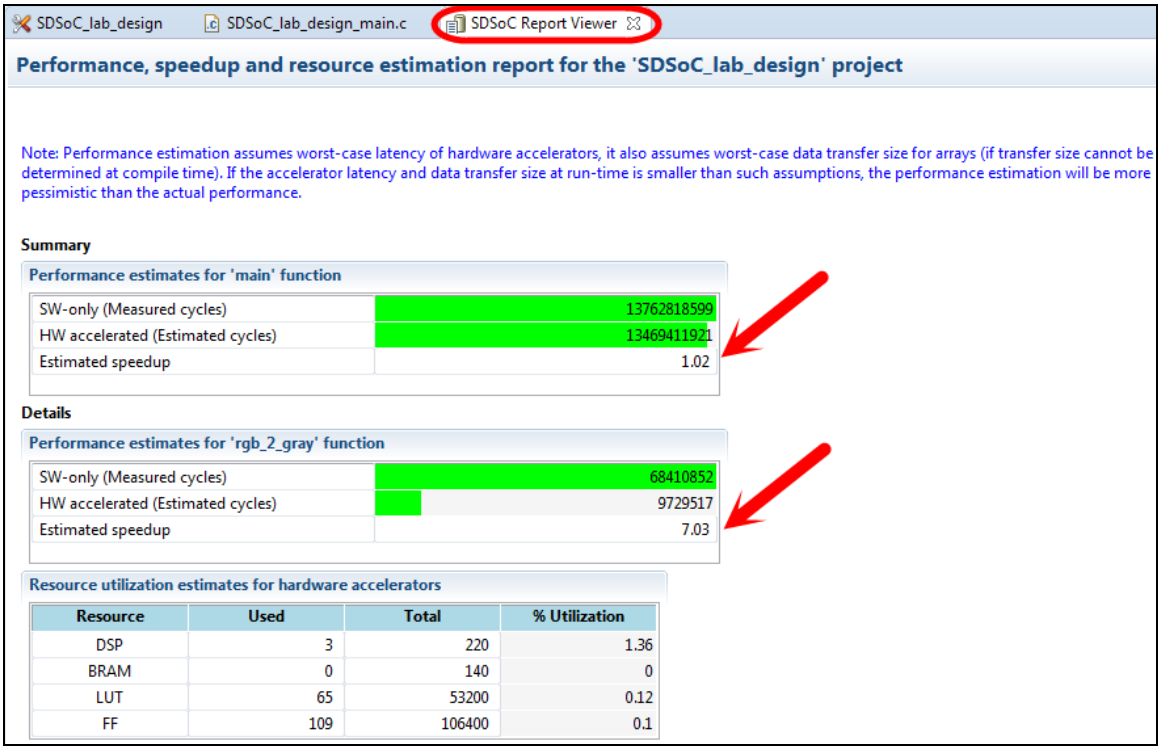
Action with Description	Point of Emphasis and Key Takeaway
• Double-click **rgb_2_gray** in the Project Explorer tab to open it in the editor. At this time, the Outline panel becomes active and enables quick navigation to the various functions and #define statements (if any) in the source. • Right-click **rgb_2_gray** in the Project Explorer panel and select **Toggle HW/SW** to mark a function for hardware. • This is an alternate method to what you just saw. Repeat this process to put the function back into hardware (it should be marked with an [H]).	• Alternative method to mark a function for hardware implementation.
• Remove power from the evaluation board. • Remove the SD card, if one is present. • Apply power to the board.	• The Lab Setup Guide provides instructions for configuring the boot jumpers. These jumpers should be set to boot from the SD card. If an SD card is not present, the boot process will "fall back" to JTAG mode, which will be used here.

Action with Description	Point of Emphasis and Key Takeaway

• Run SDDebug or open a previously run project.
• To run SDDebug:
  • Enable **Estimate Performance** from the SDSoC_lab_design project tab.
  • Select **Project** > **Build All**.
  • **Note:** This process takes about two minutes to run.
  • You may have to click the **Click Here** option to get the software-only application performance.
  • **Note**: Need to wait for approximately two minutes to see the output.
	• This provides the performance and resource estimation for the project.

• The SDSoC Report Viewer report will be displayed. • Review the initial report as shown below (ZedBoard). This report contains estimate summaries.	

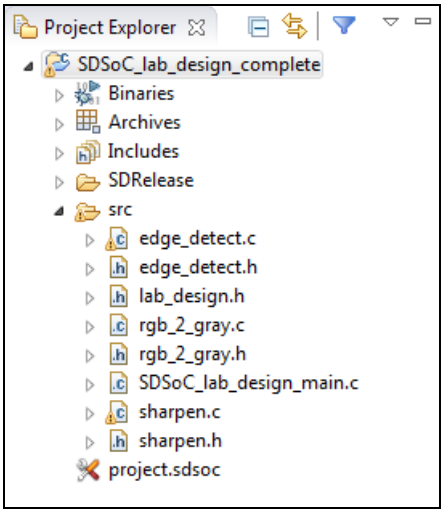
Action with Description	Point of Emphasis and Key Takeaway



- The first line of the estimation report shows the estimated speedup for the top-level function. This function is set to main by default. However, there might be code that you would like to exclude from this comparison, such as code that is allocating buffers, initialization and setup, etc.
- If you want to see the overall speedup when considering some other function, you can do this by specifying a different function as the root for the performance estimation flow. The flow works with the assumption that all functions selected for hardware acceleration are children of the root.

Action with Description	Point of Emphasis and Key Takeaway
• The pull-down "hammer" (build) menu shows what type of release the tools are set for. Building for SDRelease is possible but it will take more time than is appropriate for a demo (> 30 minutes). • **Note**: Before running the SDRelease, you should disable Estimate Performance from the SDSoC Project Overview tab.	• Generates the executable, bitstream, and SD card boot image.
• Import the completed project (Standalone files provided). *C:\training\hls\support\toolDemo\toolDemo_standalone_zc702_complete.zip* or *C:\training\hls\support\toolDemo\toolDemo_standalone_zed_complete.zip*	

Action with Description	Point of Emphasis and Key Takeaway

• Explore the project directory structure.
 
 	
	• The root folder of a project hierarchy is named after the project itself (*SDSoC_lab_design_complete* in the figure).
	• Three virtual branches are provided: Binaries, Archives, and Includes. These list executables, libraries, and header files found in or used by the project.
	• Other branches (SDDebug, SDRelease, and src) represent actual folders on the file system, build outputs, and source files. Common to every build folder are several important components:
	  • The compiled application executable (*.elf*) is placed at the root of the build folder.
	  • Several auto-generated make files used by the IDE to initiate the build process.
	  • A *src* folder that contains compiled source files (i.e., object files).
	  • An *sd_card* folder that contains the SD card image.
	  • The final *_sds* folder is a catch all for everything else (reports, files generated by back-end tools, software stubs, etc...).
	• The final element at the root level is the actual SDSoC project file: *project.sdsoc*.

Action with Description	Point of Emphasis and Key Takeaway
• Review the compilation/synthesis report by looking in the *reports* folder under *SDSoC_lab_design_complete* > *SDRelease* > *_sds*.	
• You can open the block design and show how the accelerator is connected to the system. You will find the Vivado Design Suite project in the *C:\training\hls\demos\toolDemo\SDSoC_lab_design_complete\SDRelease_sds\p0\ipi* folder.	

Summary

There is a distinct flow through the SDSoC tools. Once a project is created or imported it can be examined, profiled, modified, and estimated. Typically this is an iterative process that completes when system performance is met. The details of improving the performance of the accelerators themselves is a topic covered in several other topic clusters.

References:

- *SDSoC Environment User Guide* (UG1027)
- *SDSoC Environment User Guide: Getting Started* (UG1028)