Xilinx Tool Flow Lab

Introduction

This lab provides a basic introduction to the ISE software tools. You will complete and implement an existing PicoBlaze design. PicoBlaze will be used in the labs throughout this workshop to illustrate the ISE design flow and various point tools. This is by no means training on designing with PicoBlaze.

Objectives

After participating in this demonstration, you will be able to:

- Create a new project
- Simulate a design
- Implement a design

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 5 primary steps: You will create a new project, add an existing design, complete the design, simulate the design, and, finally, implement the design.

Note: If you are unable to complete the lab at this time, you can download the original lab files for this lab from www.xilinx.com/training/downloads.htm. These are the original lab files and do not contain any work that you may have previously completed.

General Flow for this Lab

Create a New Project — Step 1

1-1. Create a new project targeting the Spartan 6 device that is on the Nexys3 board. Specify your language of choice, VHDL or Verilog, to complete the lab.

1-1-1. Launch ISE: Select **Start → All Programs → Xilinx ISE Design Suite 13.2 → ISE Design Tools → Project Navigator**.

1-1-2. In the Project Navigator, select **File → New Project.** The New Project Wizard opens.

1-1-3. For Project Location, use the "…" button to browse to one of the following directories, and then click **OK**.

- Verilog users: *c:\xup\fpgaflow\labs\verilog\lab1*
- VHDL users: *c:\xup\fpgaflow\labs\vhdl\lab1*

1-1-4. For Project Name, type ***Flow_lab.***

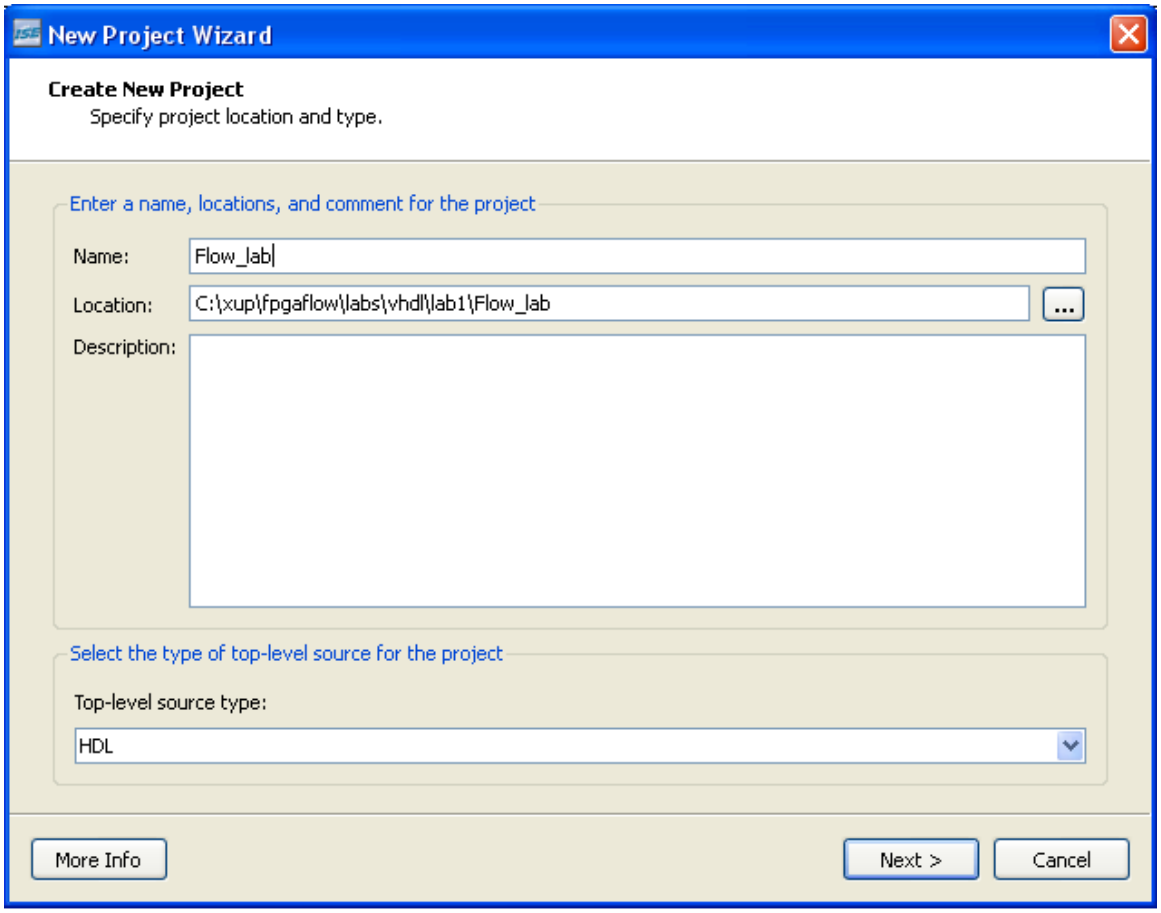


Figure 1. New Project Wizard

1-1-5. Click **Next.**

1-1-6. Select the following options and click **Next**:

Device Family: **Spartan6**
Device: **XC6SLX16**
Package: **CSG324**
Speed Grade: **–2**
Synthesis Tool: **XST (VHDL/Verilog)**
Simulator: **ISim (VHDL/Verilog)**
Preferred Language: **Verilog** or **VHDL** (select your preference)

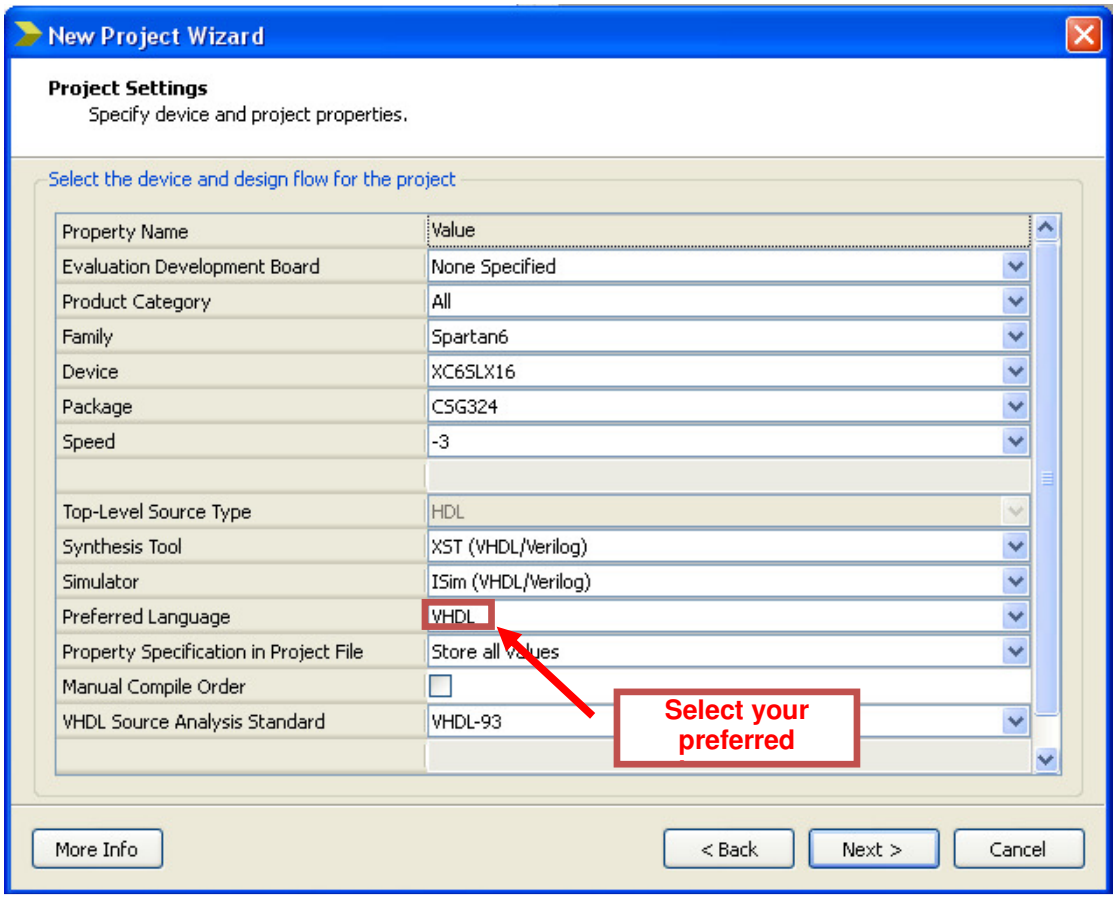


Figure 2. Device and Design Flow Dialog

1-1-7. Click **Finish.**

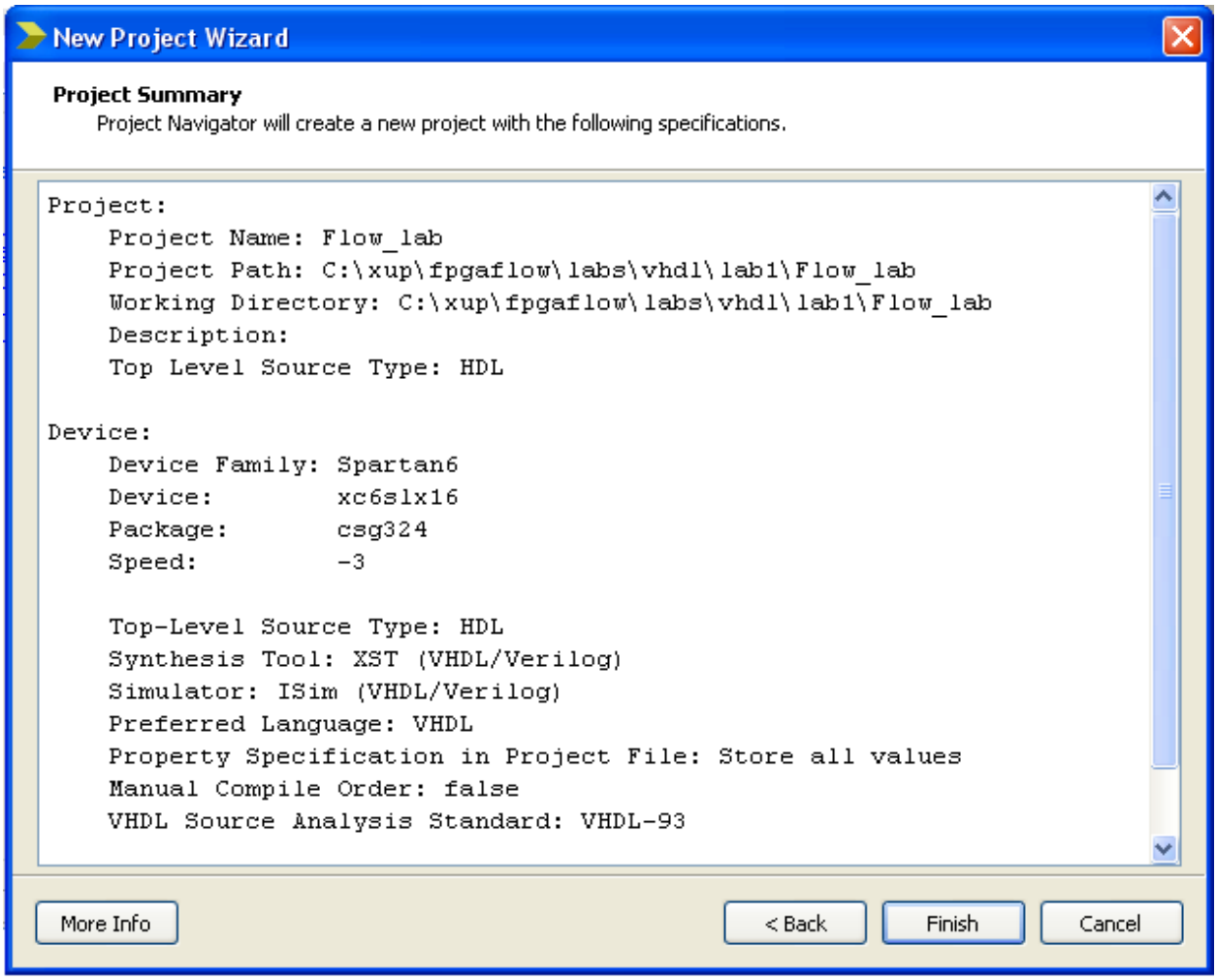


Figure 3. Project Summary Dialog

Add an Existing Design to the Project — Step 2

2-1. Add HDL source files for an example PicoBlaze design. You may review the PicoBlaze documentation to become familiar with the 8-bit microcontroller architecture and assembler. Refer to KCPSM3_manual.pdf in the ..\KCPSM3\docs\ directory.

2-1-1. Go to **Project → Add Copy of Source** or click on  from the side button window and browse to the c*:\xup\fpgaflow\KCPSM3\VHDL or Verilog* folder.

2-1-2. Select the VHDL/Verilog files ***kcpsm3_int_test*** and ***kcpsm3*** files and click **Open.**

The dialogue below should appear which allows you to select a flow (none, implementation, simulation, or both) associated with each source file.

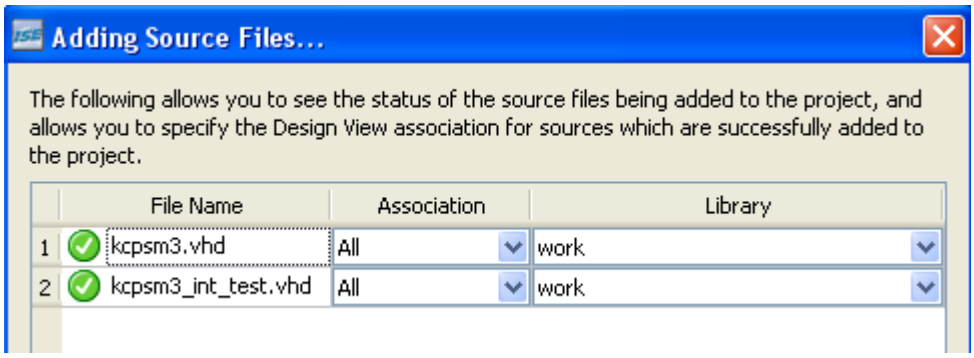


Figure 4. Choose Source Type

2-1-3. Click **OK** accepting the default setting of **All** for both source files.

Note: You should see a module called **int_test** listed in the hierarchy view with an orange question mark. This module is a BlockRAM that will contain the instructions for the PicoBlaze controller, which will be added in a later step.

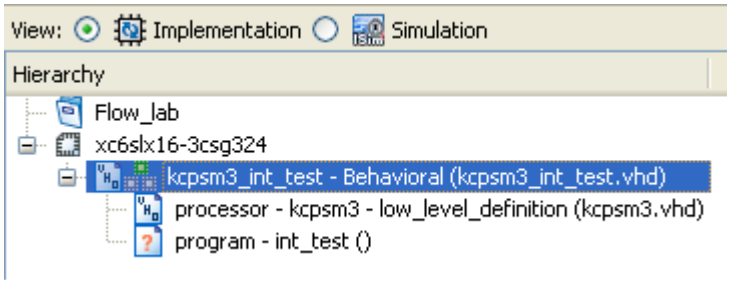


Figure 5. Design Hierarchy in Sources Window

Complete the Design Step 3

3-1. An example assembly (.psm) file called init_test.psm is included with the PicoBlaze distribution. You will assemble this file to generate the instruction ROM and add it to the design.

3-1-1. Open up Windows Explorer and browse to the Assembler provided in the KCPSM3 sub-directory (c:\xup\fpgaflow\KCPSM3\Assembler).

Note: The KCPSM3.exe assembler and ROM_form* template files along with two example PSM files should reside in this directory. Keep in mind that the assembled output files will be generated in the directory containing the assembler and template files. It may be beneficial to copy the assembler and template files to your project directory. For the workshop, we will keep the files in the current location.

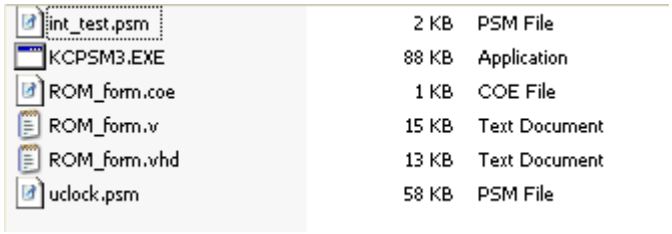


Figure 6. PicoBlaze Assembler Files

3-1-2. Open the **int_test.psm** file using a standard text editor, such as Wordpad, and review the code, referring to the *PicoBlaze 8-bit Embedded Microcontroller User Guide* or *KCPSM3 manual* for technical guidance. These documents are provided in the **docs** sub-directory.

3-1-3. Open a command window by clicking **Start → All Programs → Accessories → Command Prompt.**

3-1-4. Browse to the **Assembler** directory using the **cd** command
> cd c:\xup\fpgaflow\KCPSM3\Assembler

3-1-5. Generate the ROM definition files by assembling the example assembly application. Enter the following command at the command prompt
> kcpsm3 int_test.psm

Note: You should now see several files in the Assembler sub-directory starting with init_test*, including VHDL (int_test.vhd) and Verilog (int_test.v) ROM definition files.

3-1-6. Add the Verilog or VHDL ROM definition file to the project. Click **Project → Add Copy of Source** or click on  from the side button window and select either **int_test.vhd** or **int_test.v** file (**Figure 7**).

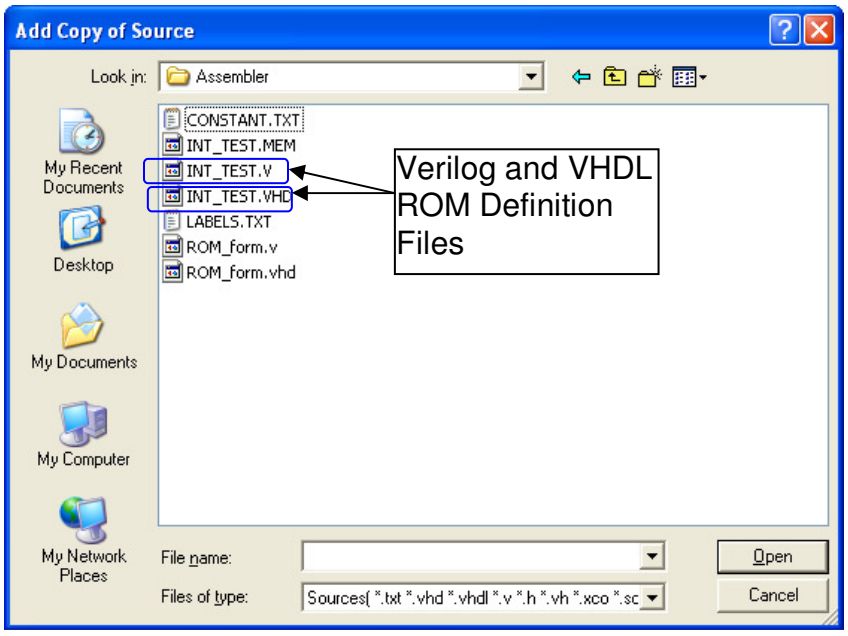


Figure 7. VHDL and Verilog ROM Definition Files

3-1-7. Click **Open** and then **OK** to add **INT_TEST** as a VHDL/Verilog Design File to the project (**Figure 8**).

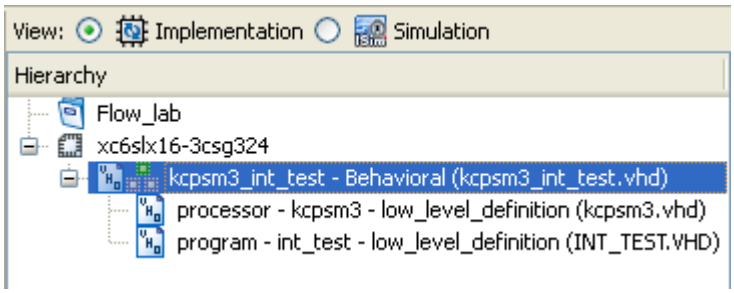


Figure 8. Hierarchical view of PicoBlaze design

Note: The top-level **kcpsm3_int_test** file contains an instantiation of the **int_test** ROM definition file. After adding this source code for the **int_test** to the design, the red question mark in the module view will disappear as it is no longer seen as a black box.

Simulate the Design — Step 4

4-1. Add the testbench and review the code. Run a behavioral simulation using the Xilinx ISIM simulator and analyze the results.

4-1-1. Click **Project → Add Copy of Source** and browse to c:\xup\fpgaflow\KCPSM3\vhdl (or verilog).

4-1-2. Select the testbench file (**test_bench.vhd** or **testbench.v)** and click **Open.**

4-1-3. Set the association to **Simulation** and click **OK** to add the test bench to the project. Click **Simulation** in the View pane, select *Behavioral* and click **testbench-behavior** in the *Hierarchy* window.

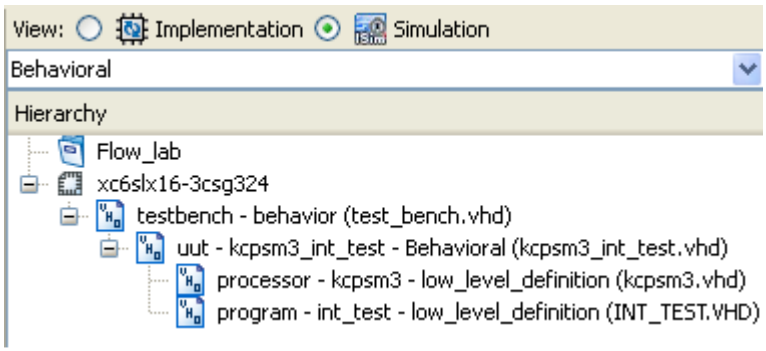


Figure 9. Hierarchical View Including Test Bench

4-1-4. Open **the ISE Simulator Properties** dialogue. Click the testbench so it is highlighted, expand the **ISim Simulator** toolbox in the Processes window, right-click on **Simulate Behavioral Model**, and select **Process Properties.**

4-1-5. Enter the value of 3000ns for the **Simulation Run Time** and click **OK.**

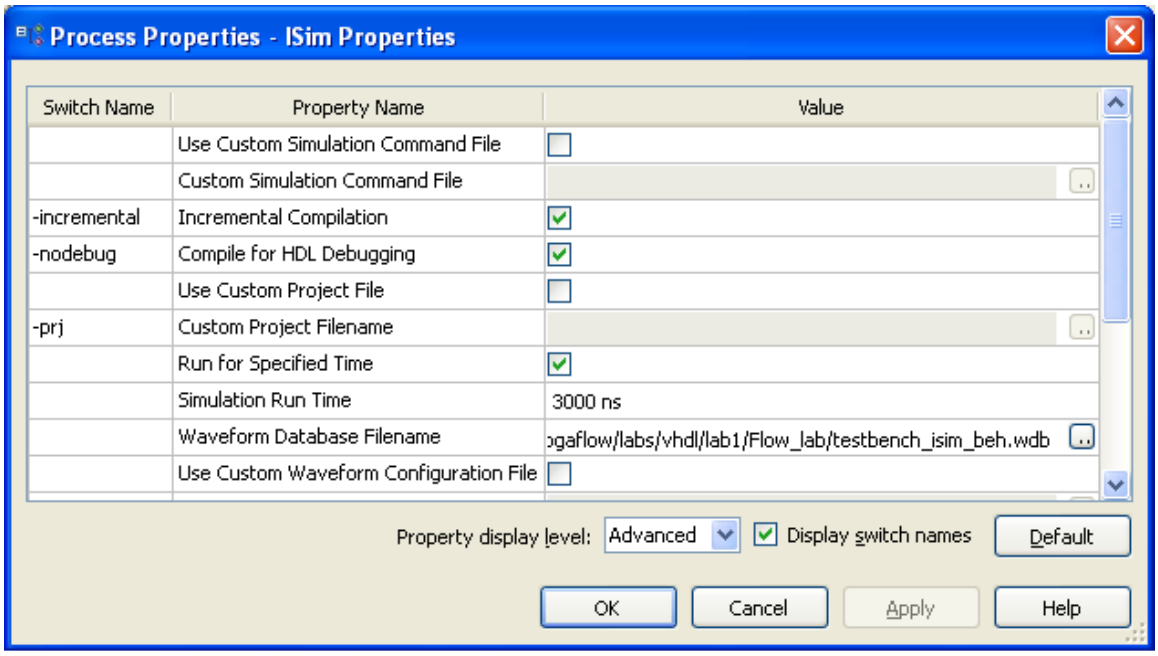


Figure 10. ISIM Behavioral Simulation Properties

4-1-6. Double-click **Simulate Behavioral Model** to simulate the design (**Figure 11**). Click on *Zoom to Full View* button. Change radix of waveforms signal by selecting it in the Name column of the waveform window, right-click, select Radix followed by Hexadecimal. You should see three input interrupt pulses and the displayed interrupt count value. You should also see an alternating inverted pattern displayed.

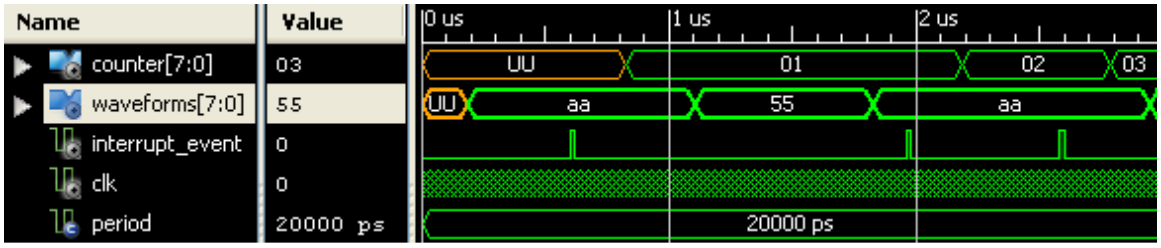


Figure 11. iSIM Behavioral Simulation Results

4-2. The steps below are for illustrative purposes only, and show how to analyze the internal signals of the design. The first step shows how to add internal signals to the waveform. The second step shows how to analyze the interrupt process. The third step shows how to analyze the output waveform process. You may optionally complete these steps if you have additional time at the end of the lab.

4-2-1. Monitor internal signals by adding them to the design.

You need to select the desired module entry [uut] in **Sources** window, and then select the desired signal [address] in Processes window. Right-click on it and select **Add to Wave Window**. Similarly add interrupt, interrupt_ack, and instruction signals.

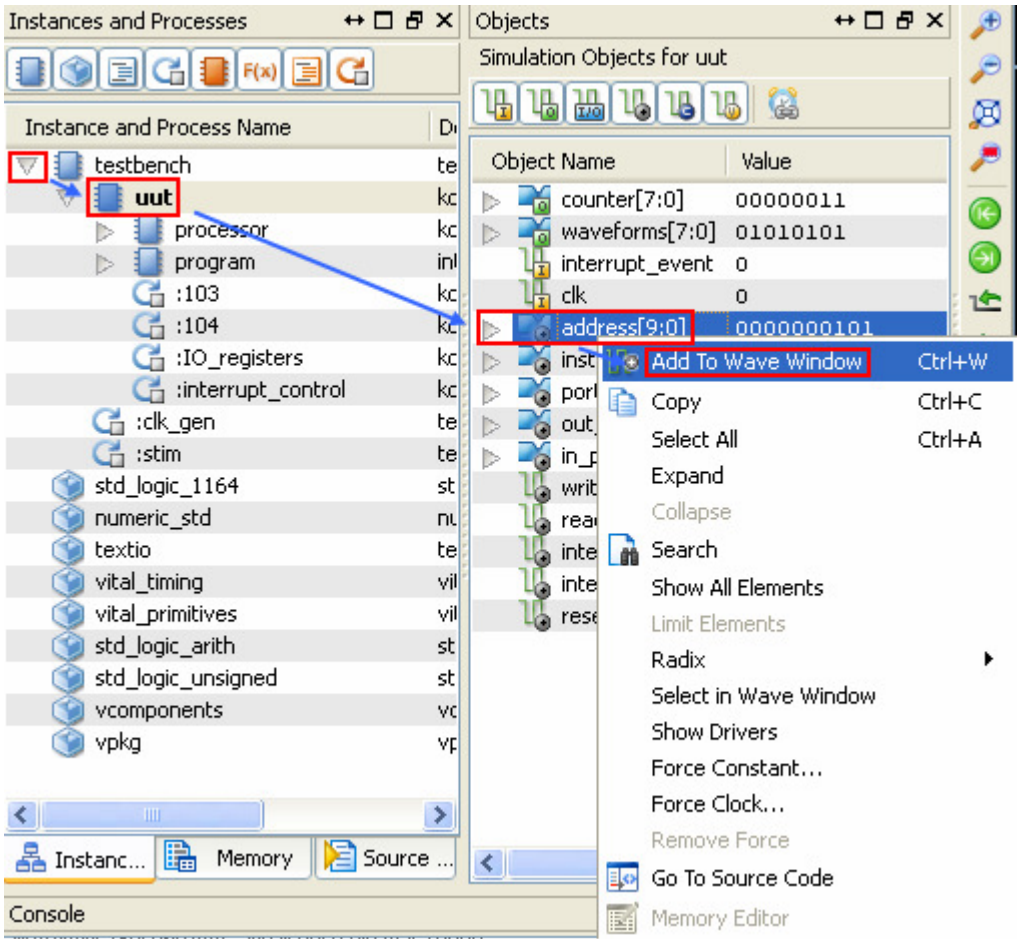
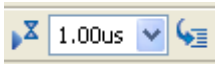


Figure 12. Accessing Internal Signals

4-2-2. Change radix of address and instructions to hexadecimal. Enter **3000ns** in 

in the tool buttons bar where 1.00us is displayed; click **Simulation → Restart,** followed by **Simulation → Run** to re-simulate the design. Analyze the waveform for Interrupt Service Routine process (**Figure 1-13**).

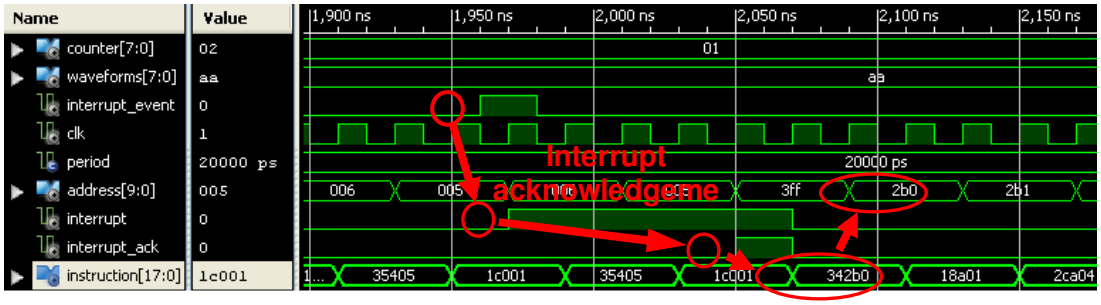


Figure 13. Interrupt Service routine

4-2-3. Change simulation run time to 25000ns, add write_strobe signal and re-simulate the design. Analyze the output waveform process (**Figure 14**).

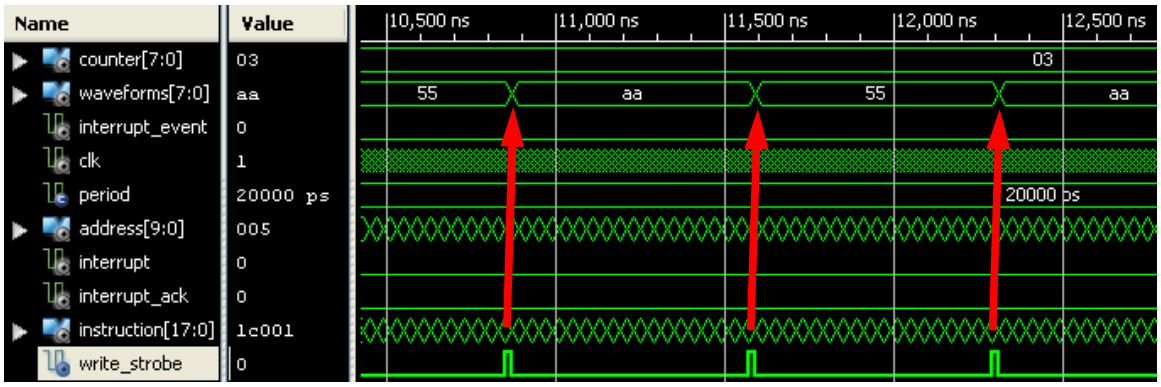


Figure 14. Output Waveform

Note: The int_test.log file in the Assembly directory shows the address and code for each instruction

4-2-4. Close the **simulator windows**. Click **Yes** to confirm that you want to end the simulation and click **NO.**

Implement the Design — Step 5

5-1. Implement the design. During implementation, some reports will be created. You will look more closely at some of these reports in a later module.

5-1-1. In the Sources in Project window, select **Implementation** in the Sources pane and select the top-level design file ***kcpsm3_int_test.vhd/v*** (**Figure 15**).

Make sure that it set as Top Level Module. If not then make it as the Top Level module by selecting it, right-clicking, and choosing the option.

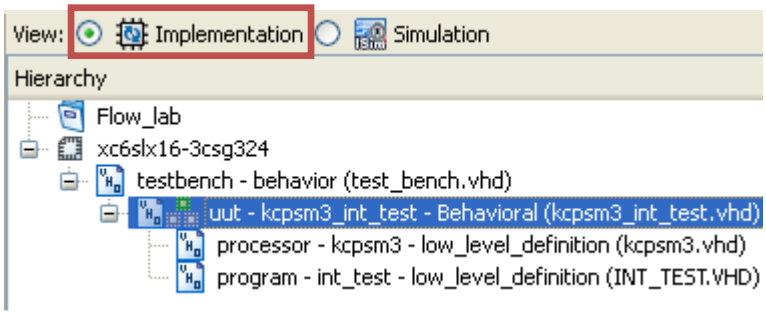


Figure 15. Sources Window Pane

5-1-2. In the Processes for Source window, double-click **Implement Design** (**Figure 1-16**).

Notice that the tools run all of the processes required to implement the design. In this case, the tools run Synthesis before going into Implementation.

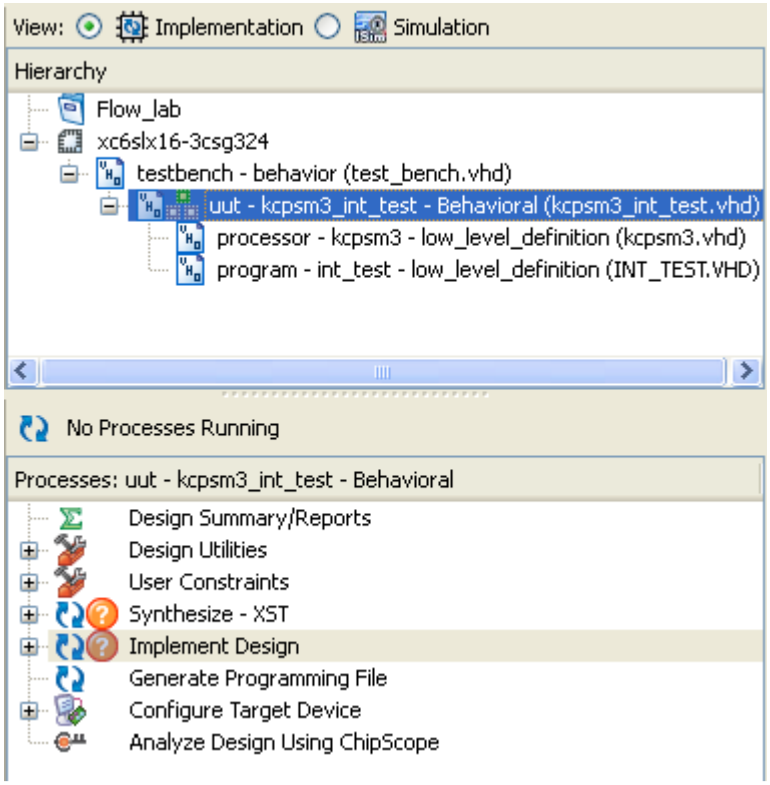


Figure 16. Processes for Source Window

5-1-3. While the implementation is running, click the **+** next to **Implement Design** to expand the implementation step and view the progress. We refer to this as *expanding* a process.

After each stage is completed, a symbol will appear next to each stage:

Check mark for successful
Exclamation point for warnings
X for errors

For this particular design, there may be an exclamation point (warnings) for some steps. The warnings here are okay to ignore.

5-1-4. Read some of the messages in the message window located across the bottom of the Project Navigator window.

5-1-5. When implementation is complete, review the design utilization in the **Design Summary** window (**Table 17**).

Table 17. Design Summary

kcpsm3_int_test Project Status (08/09/2011 - 13:55:47)			
Project File:	Flow_lab.xise	**Parser Errors:**	No Errors
Module Name:	kcpsm3_int_test	**Implementation State:**	Placed and Routed
Target Device:	xc6slx16-3csg324	**• Errors:**	No Errors
Product Version:	ISE 13.2	**• Warnings:**	1 Warning (0 new)
Design Goal:	Balanced	**• Routing Results:**	All Signals Completely Routed
Design Strategy:	Xilinx Default (unlocked)	**• Timing Constraints:**	All Constraints Met
Environment:	System Settings	**• Final Timing Score:**	0 (Timing Report)

Device Utilization Summary				[-]
Slice Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Registers	92	18,224	1%	
Number used as Flip Flops	92			
Number used as Latches	0			
Number used as Latch-thrus	0			
Number used as AND/OR logics	0			
Number of Slice LUTs	121	9,112	1%	
Number used as logic	98	9,112	1%	
Number using O6 output only	59			
Number using O5 output only	0			
Number using O5 and O6	39			
Number used as ROM	0			
Number used as Memory	22	2,176	1%	
Number used as Dual Port RAM	8			
Number using O6 output only	0			
Number using O5 output only	0			
Number using O5 and O6	8			
Number used as Single Port RAM	14			
Number using O6 output only	10			
Number using O5 output only	0			
Number using O5 and O6	4			
Number used as Shift Register	0			
Number used exclusively as route-thrus	1			

Conclusion

In this lab, you completed the major stages of the ISE™ design flow: creating a project, adding source files, simulating the design, and implementing the design.

In the following labs, you will examine some of the software reports, determine how the design was implemented, and determine whether or not your design goals for area and performance were met.