A Basic SoC

Introduction

This lab guides you through the process of using Vivado IDE to create a simple Cortex-M0 soft core based SoC design targeting the Nexys4 board. You will simulate, synthesize, and implement the design with default settings. Finally, you will generate the bitstream and download it in to the hardware to verify the design functionality

Objectives

After completing this lab, you will be able to:

- Create a Vivado project sourcing HDL model(s) and targeting a specific FPGA device attaching to LEDs located on the Nexys4 board
- Use the provided Xilinx Design Constraint (XDC) file to constrain the pin locations
- · Simulate the design using the Vivado simulator
- Synthesize and implement the design
- Generate the bitstream
- Configure the FPGA using the generated bitstream and verify the functionality

Procedure

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

NOTE: Before proceeding with the lab, please copy the Cortex M0 Design Start files (CORTEXM0DS.v and cortexm0ds_logic.v) in to these locations

- 1. P6/Lab Part1/FPGA/Source
- 2. P6/Lab Part2/FPGA/Source
- 3. P7/Lab/FPGA/Source/CortexM0-DS
- 4. P8/Lab/FPGA/Source/CortexM0-DS

Design Description

The design consists of the Cortex-M0 Design Start (DS) core, 1 KB of RAM implemented in FPGA, AHB2LED IP to drive LEDs as shown in **Figure 1**.



Figure 1. The Completed Design

General Flow

Functionality in Hardware





Create a Vivado Project using IDE

Step 1

- 1-1. Launch Vivado and create a project targeting the XC7A100TCSG324-1 device and using the Verilog HDL. Use the provided files from the *P6\FPGA\Source* directory.
- 1-1-1. Open Vivado by selecting Start > All Programs > Xilinx Design Tools > Vivado 2013.4 > Vivado 2013.4
- **1-1-2.** Click **Create New Project** to start the wizard. You will see *Create A New Vivado Project* dialog box. Click **Next**.
- **1-1-3.** Click the Browse button of the *Project location* field of the **New Project** form, browse to <**labs_install>\P6\FPGA**, and click **Select**.
- **1-1-4.** Enter **lab1** in the *Project name* field. Make sure that the *Create Project Subdirectory* box is checked. Click **Next**.

🚴 New Project	X
Project Name	
Enter a name for your project and specify a directory where the project data files will be stored	
Project name: lab1	8
Project location: C:/xup/SoC_Workshop/P6/FPGA	×
Create project subdirectory	
Project will be created at: C:/xup/SoC_Workshop/P6/FPGA/lab1	
< Back Next > Finish C	ancel

Figure 2. Project Name and Location entry

- 1-1-5. Select RTL Project option in the Project Type form, and click Next.
- **1-1-6.** Using the drop-down buttons, select **Verilog** as the *Target Language* and *Simulator Language* in the *Add Sources* form.



1	New Pro	oject			
Ac	dd Sourc	es			
	new sou	Irce file o	n disk and	add it to your project. '	You can also add and create sources later.
	Teday	Marriso	Liburnus		
	Index	Name	Library	HDL Source For	
				dd Eilos	dd Directories Create File
		L LL DT			Create File
	Scan an		L Include fi	les into project	
	Copy so	urces into	o project		
	Add sou	irces from	n subdirect	ories	
1	Target lar	nguage:	Verilog 🔻	Simulator language:	Verilog 🔻
					< Back Next > Finish Cancel

Figure 3. Selecting Target and Simulator language

- 1-1-7. Click on the Add Files... button, browse to the <labs_install>\P6\FPGA\Source directory, select AHB2LED.v, AHB2BRAM.v, AHBDCD.v, AHBLITE_SYS.v, AHBMUX.v, CORTEXMODS.v, and cortexm0ds_logic.v, (do not select lab1_tb.v) and click OK, and then click Next to get to the Add Existing IP form.
- 1-1-8. Since we do not have any IP to add, click Next to get to the Add Constraints form.
- 1-1-9. Click on the Add Files... button, browse to the <labs_install>\P6\FPGA\Source directory (if necessary), select Nexys4_Master.xdc and click OK (if necessary), and then click Next.

This Xilinx Design Constraints file assigns the physical IO locations on FPGA to the clk, reset, and LEDs located on the board. This information can be obtained either through the board's schematic or the board's user guide.

1-1-10. In the *Default Part* form, using the **Parts** option and various drop-down fields of the **Filter** section, select the **XC7A100TCSG324-1** part. Click **Next**.



🚴 New Proj	ject				-				X
Default Part	t								
Choose a	Choose a default Xilinx part or board for your project. This can be changed later.								
Specify	Filtor								
Specity	Printer Dura durati ant					alua an Lana 2	24		
W Parts	Product cat	egory All	_		• Pa	ckage csg3	24		<u> </u>
📓 Boards	F	-amily Artix	.7		Speed	grade All R	temaining		-
	Sub-F	amily Artix	7		▼ Temp	grade C			<u> </u>
				Rese	et All Filters				
Search: Q	r								
Part		I/O Pin Count	Available IOBs	LUT Elements	FlipFlops	Block RAMs	DSPs	Gb Transceivers	G Ti
xc7a75tcs	q324-3	324	210	63400	126800	135	240	0	0
xc7a75tcs	g324-2	324	210	63400	126800	135	240	0	0
🔷 xc7a75tcs	g324-2L	324	210	63400	126800	135	240	0	0
🔷 xc7a75tcs	g324-1	324	210	63400	126800	135	240	0	0
🔷 🔷 xc7a100tc	:sg324-3	324	210	63400	126800	135	240	0	0
🔷 xc7a100tc	:sg324-2	324	210	63400	126800	135	240	0	0
🔷 xc7a100tc	:sg324-2L	324	210	63400	126800	135	240	0	0
🔷 xc7a100tc	:sg324-1	324	210	63400	126800	135	240	0	0
•								4	
	< Back Next > Finish Cancel								

Figure 4. Part Selection

You can select the Boards Specify option, select Artix-7 under the Library filter and select the appropriate board. Notice that Nexys4 is not listed as it is not in the tools database.

🚴 New Project				mentalise a	X
Default Part					
Choose a default Xilinx part or board for your	project. This	can be change	ed later.		
Specify Filter					
Parts Board Ven	dor xilinx.co	m		v	
Boards Libr	ary artix7			-	
Na	me ac701			-	
Vers	sion 1.0			-	
	Re	eset All Filters			
Search: Q-					
Board	Board Vendor	Board Library	Board Name	Board Version	Part
Artix-7 AC701 Evaluation Platform	xilinx.com	artix7	ac701	1.0	🔷 xc7a200tfbg676
		< Ba	ack Nex	t > Fini	sh Cancel

Figure 5. Part Selection using Boards Specify filter



1-1-11. Click **Finish** to create the Vivado project.

Use the Windows Explorer and look at the <labs_install>\P6\FPGA\lab1 directory. You will find that the lab1.data and lab1.srcs directories and the lab1.xpr (Vivado) project file have been created. The lab1.data directory is a place holder for the Vivado program database. Two directories, constrs_1 and sources_1, are created under the lab1.srcs directory; deep down under them, the copied Nexys4_Master.xdc (constraint) and all Verilog (source) files respectively are placed.

🛯 🝌 P6
a 🗼 FPGA
🛛 👢 labl
🔺 👢 lab1.data
👢 constrs_1
👢 sim_1
👢 sources_1
👢 wt
🛛 👢 lab1.srcs
⊿ 👢 constrs_1
4 👢 imports
📜 _{Source} 🛛 xdc file
⊿ ↓ sources_1
Source Verling file

Figure 6. Generated directory structure

1-2. Open the AHBLITE_SYS.v source and analyze the content.

1-2-1. In the *Sources* pane, double-click the **AHBLITE_SYS.v** entry to open the top-level file in text mode.

You can click on the + sign to expand the hierarchy and see lower-level instantiated modules and associated source files.



Figure 7. Opening the source file



- **1-2-2.** Line 38 defines the beginning (marked with keyword **module**) and Line 239 defines the end of the module (marked with keyword **endmodule**).
- **1-2-3.** Lines 40-45 define the input and output ports, lines 49-85 define internal connections/nets, whereas lines 144-238 define various modules instantiations.

1-3. Open the Nexys4_Master.xdc source and analyze the content.

1-3-1. In the *Sources* pane, expand the *Constraints* folder and double-click the **Nexys4_Master.xdc** entry to open the file in text mode.



Figure 8. Opening the constraint file

1-3-2. Lines 8-10 define the clock pin location, its I/O property, and 100 MHz frequency constraint. Lines 13-14 define the reset pin location which is controlled by SW15 of the board. Lines 20-42 define the pin locations of the output LEDs [7:0] and lines 44-45 define the LOCKUP signal on LED15 of the board.

1-4. Perform RTL analysis on the source file.

1-4-1. Expand the Open Elaborated Design entry under the RTL Analysis tasks of the Flow Navigator pane and click on **Schematic**.

A warning indicating that the BRAM data file could not be found will be displayed. Click **OK** to close the warning box for now.

The model (design) will be elaborated and a logic view of the design is displayed. Click on the Zoom-In button ($\stackrel{\circ}{\sim}$) in the left vertical toolbar to see zoom in.





Figure 9. A logical view of the design





Figure 10. Hierarchy view

1-4-3. Click on the Previous button ($\stackrel{\bigstar}{\bullet}$) to see the upper-level view.

Simulate the Design using the Vivado Simulator Step 2

2-1. Add the lab1_tb.v testbench file.

- **2-1-1.** Click Add Sources under the *Project Manager* tasks of the *Flow Navigator* pane.
- 2-1-2. Select the Add or Create Simulation Sources option and click Next.



Add Sources		RT
	Add Sources	
	This guides you through the process of adding and creating sour	ces for your project
	Add or Create Constraints	
	Add or Create Design Sources	
	Add or Create Simulation Sources	
	Add or Create DSP Sources	
	Add Existing Block Design Sources	
	C Add Existing IP	

Figure 12. Selecting Simulation Sources option

- 2-1-3. In the Add Sources Files form, click the Add Files... button.
- **2-1-4.** Browse to the **<labs_install>\P6\FPGA\Source** folder and select *lab1_tb.v* and click **OK**.
- 2-1-5. Click Finish.
- 2-1-6. Select the Sources tab and expand the Simulation Sources group.

The lab1_tb.v file is added under the *Simulation Sources* group, and **AHBLITE_SYS.v** and its lower-level modules are automatically placed in its hierarchy as a *dut* instance.

Sources		×					
으 🄀 🖨 🖬 🔂							
🕞 😚 Design Sources (2)							
H . AHBLITE_SYS (AHBLITE_SYS.v) (5)							
🗄 🛅 Unknown (1)							
🗄 🗁 Constraints (1)	E Constraints (1)						
🖻 ն Simulation Sources (1)							
🖻 🗟 sim_1 (1)							
🖨 🐨 📲 lab1_tb (lab1_tb.v) (1)	7						
dut - AHBLITE_SYS (AHBLITE_SYS.v) (Section 2014)	5)						
Hierarchy Libraries Compile Order							
🕹 Sources 🛛 😰 RTL Netlist							

Figure 13. Simulation Sources hierarchy

- 2-1-7. Using the Windows Explorer, verify that the sim_1 directory is created at the same level as constrs_1 and sources_1 directories under the lab1.srcs directory, and that a copy of lab1_tb.v is placed under lab1.srcs > sim_1 > imports > sources.
- **2-1-8.** Double-click on the **lab1_tb** in the *Sources* pane to view its contents.



```
1 `timescale 1ns / 1ps
3 // Module Name: lab1 tb
5 module lab1_tb(
6
7
    );
8
9
    reg RESET, CLK;
10
    wire [7:0] LED;
11
    wire LOCKUP;
12
13
    AHBLITE_SYS dut(.CLK(CLK), .RESET(RESET), .LED(LED), .LOCKUP(LOCKUP));
14
15
    initial
16
    begin
17
       CLK = 0;
18
       forever
19
       begin
20
         #5 CLK = 1;
21
         #5 CLK = 0;
22
       end
23
    end
24
25
    initial
26
    begin
27
       RESET = 0;
28
       #30 RESET = 1;
29
       #20 RESET = 0;
30
    end
31
32 endmodule
```

Figure 14. The self-checking testbench

The testbench defines the simulation step size and the resolution in line 1. The testbench module definition begins on line 5. Line 13 instantiates the DUT (device/module under test). Lines 15 through 23 define the clock signal generation. Lines 25 through 30 define the RESET stimulus generation. Line 32 ends the testbench.

2-2. Add the memory file.

2-2-1. Click Add Sources under the Project Manager tasks of the Flow Navigator pane.



Figure 11. Add Sources

- 2-2-2. Select the Add or Create Simulation Sources option and click Next.
- **2-2-3.** Click the **Add Files**... button, select *All Files* in the *Files of type* in the filter field, select code.hex created in the software folder
- 2-2-4. Make sure you uncheck the box "Copy source into Project" before clicking Finish



S	pecify simul	ation set:	🛅 sim_1	▼
	Index	Name	Library	Location
8	1	code.hex	N/A	C:/Users/karshi01/Desktop/Workshop/USB/Full Material - V2/P6/Lab Part1/Software
				Add Files Add Directories
] Scan and	add RTL in	clude files	Add Files Add Directories Create File
] Scan and] Copy <u>s</u> ou	add RTL in	clude files	Add Files Add Directories Create File

2-3. Simulate the design for 1000 ns using the Vivado simulator.

2-3-1. Click on **Run Simulation > Run Behavioral Simulation** under the *Project Manager* tasks of the *Flow Navigator* pane.

The testbench and source files will be compiled and the Vivado simulator will be run (assuming no errors). You will see a simulator output similar to the one shown below.

🚴 lab1 - (C:/Users/karshi01/Deskto	p/Workshop/USB/Full I	Material - V2/P6/Lab P	art1/FPGA/lab1/lab1.x	pr] - Vivado 2013.4	-	-							- 0 -	x
<u>File Edit Flow Tools Windo</u>	w Layout <u>V</u> iew <u>R</u> i	un <u>H</u> elp										Q,+ Search	commands	_
🯄 📄 🕼 🕫 🗎 📉 🖇	> 🕨 🐮 🚳 🕺 🗎	∑ 🧔 😬 Default Lay	out 👻 🗶	🔌 🎉 🚧 🛼	(r) 10 us	- 🧏 📗 📮	I E						R	eady
Flow Navigator «	Behavioral Simula	Behavioral Simulation - Functional - sim_1 - lab1_tb X												
< 🔀 🚔	Scopes		_ 🗆 🖻 ×	Objects		- 🗆 🖉 🗡	0	AHBLITE_SYS.v ×	🚆 Untitled 1 🗙				00	×
	🔍 🖾 🖨 🚺 🕻	9 🖃 🚰 📒 F(x)	G	< 🎦 🗃 👪	16 18 18		÷						1,000.000 n	-
Project Manager Option Contract	Name	Design Unit	Block Type	Name	Value	Data Type	1	Name	Value	10		1500		
Add Sources	😑 📕 lab1_tb	lab 1_tb	Verilog Module	r 🐻 RESET	0	Logic	0+	15 DECET	0	0 ns		SUU na		4
TID Catalan	🖲 😫 dut	AHBLITE_SYS	Verilog Module	10 CLK	0	Logic		15 CIK	U					1
- ir Catalog	ど gibi	gbi	veniog module	LOCKUP	01010101	Logic			01010101	000000		0101010 \ 0101	10101010 10	4
 IP Integrator 								U LOCKUP	0			(0101010)	10101010 /(0	
ở Create Block Design							<u>*</u>							1
💕 Open Block Design														1
🍓 Generate Block Design														
4 Simulation							E							
Simulation Settings	U III													
Run Simulation							-							
	Scope 👗	Sources		1			le.							
 A RTL Analysis 	Emulation Econo Dra						-4							
Den Elaborated Design	Simulation Scope Pro	peroes	- 0 6 ^											
 Synthesis 	: 😝 lab1_tb			1			<u>çı</u> ı							
🍪 Synthesis Settings				1			-							
Run Synthesis	Name: /lab1	_tb												
Open Synthesized Design	Block type: Verilo	tD a Module												
4 Implementation	File: C:/Us	ers/karshi01/Desktop/Wo	rkshop/USB/Full Materia											
Implementation Settings														
Run Implementation														-
Open Implemented Design			•					<u> ۲</u>	+ <	- F - K	m			÷
	Td Console												- 0 2	×
 Program and Debug 	Vivado S	imulator 2013.4												*
Bitstream Settings	Time res	olution is 1 ps												
Cenerate Bitstream	INFO: IV	me (8): cpu = 00: 'ivado 12-13951 XS	im completed. Des	ign snapshot 'l	abl th behav	1K = 913.074	; ga:	in = 0.000						
Open Hardware Manager	aunch_x	sim: Time (s): cp	u = 00:00:11 ; el	apsed = 00:01:2	6 . Memory ()	(B): peak = 9	913.01	74 ; gain = 0.00	0					
Launch MPACT														-
	X		m										•	
	Type a Tcl	command here												
	📃 🗌 Tcl Console	e 🗋 🗩 Messages 🗌 🖾 Lo	g											
													Sim Time: 1 r	10

Figure 16. Simulator output

You will see several buttons next to the waveform window which can be used for the specific purpose as listed in the table below.

_								
*	Waveform options							
冎	Save the waveform							
Q+	Zoom In							
<	Zoom Out							
	Zoom Fit							
<u> </u>	Zoom to cursor							
I	Go to Time 0							
	Go to Last Time							
1	Previous Transition							
2	Next Transition							
-4	Add Marker							
F.	Previous Marker							
-	Next Marker							
++	Swap Cursors							
\$1	Snap to Transition							

Table 1: Various buttons available to view the waveform

2-3-2. Click on the *Zoom Fit* button () to see the entire waveform.

You can also float the simulation waveform window by clicking on the Float button on the upper right hand side of the view. This will allow you to have a wider window to view the simulation waveforms. To reintegrate the floating window back into the GUI, simply click on the Dock Window button.

www.xilinx.com/support/university



99 ns 🖌 🔺

Figure 18. Float Button



Figure 19. Dock Window Button

2-4. Change the display format if desired.

2-4-1. Select the LED[7:0] in the waveform window, right-click, select *Radix*, and then select *Hexadecimal*.

2-5. Add more signals to monitor the lower-level signals. and continue to run the simulation for 500 ns.

2-5-1. Expand the **lab1_tb > dut** and select the **u_cortexm0ds** instance in the *Scopes*.

Many signals including AHBLITE interface and internal processor registers will be displayed in the *Objects* window.



Scopes		_ 🗆 🖻 ×	C	bjects		2
a 🔁 😂 🚺 🕲 🔁 🖉	Ca 🚺 🗐 F(x) 🕻	4	C	C C C C C C C C C C C C C C C C C C C	60	
Name	Design Unit	Block Type	N	ame	Data T	Value
寻 🛢 lab1_tb	lab1_tb	Verilog Mo	1	- 🛅 HCLK	Logic	0
🖻 📒 dut	AHBLITE_SYS	Verilog Mo		🖞 HRESETn	Logic	1
📃 📕 u_cortexm0ds	CORTEXMODS	Verilog Mo	E	1 MADDR[31:0]	Array	XXXXXXXXXX
🖳 📒 u_logic	cortexm0ds	Verilog Mo	Ē	📸 HBURST[2:0]	Array	000
🛛 📒 uAHBDCD	AHBDCD	Verilog Mo		HMASTLOCK	Logic	0
🛛 📒 uAHBMUX	AHBMUX	Verilog Mo	Ē	📸 HPROT[3:0]	Array	XX1X
··· 🛢 uAHB2MEM	AHB2MEM	Verilog Mo	E	📸 HSIZE[2:0]	Array	0XX
🔤 uAHB2LED	AHB2LED	Verilog Mo	Œ	🛅 HTRANS[1:0]	Array	X0
🏪 🛢 glbl	glbl	Verilog Mo	E	📸 HWDATA[31	Array	XXXXXXXXXX
				- 🛅 HWRITE	Logic	х
			E	📅 HRDATA[31:0]	Array	110111101
				🗥 🚻 HREADY	Logic	1
				🗥 🚻 HRESP	Logic	0
				"🚻 NMI	Logic	0
			E	📅 IRQ[15:0]	Array	00000000
				TXEV	Logic	X
				RXEV	Logic	0
				LOCKUP	Logic	x
				BYSRESETR	Logic	x
				SI FEPING	Logic	X
Source & Source	es		1±	nt cm0_r00[31	Array	XXXXXXXXXX
Circulation Conver Duran			1±	nt cm0_r01[31	Array	XXXXXXXXXX
Simulation Scope Proper	Ties		L±	ntext and the second se	Array	XXXXXXXXXX
🔶 🔿 🚱 🔶			1±	📲 cm0_r03[31	Array	XXXXXXXXXX
u cortoym0dc			ļ	no_r04[31	Array	XXXXXXXXXX
u_contexinious			1±		Array	XXXXXXXXXX
No		_	L±	📲 cm0_r06[31	Array	XXXXXXXXXX
Name: /lab1_tb/c	ut/u_cortexmud	IS	L±		Array	XXXXXXXXXX
Design unit: CORTEXM	10DS				Array	XXXXXXXXXX
Block type: Verilog M	odule			cm0_r09[31	Array	XXXXXXXXXX
- Verlog M			L+	r10[31	Array	XXXXXXXXXX
File: <u>C:/xup/So</u>	C Workshop/P6	/FPGA/lab1/lab1.	4	cm0_r11[31	Array	XXXXXXXXXX
				r12[31	Array	XXXXXXXXXX
			4	- cm0_msp[3	Array	XXXXXXXXXX
			L+	cm0_psp[31	Array	XXXXXXXXXX
			4		Array	XXXXXXXXXX
			1	<pre>~ cm0_pc[31:0]</pre>	Array	XXXXXXXXX
4			ļ	cm0_xpsr[3	Array	XXXX000X
		P		em0_control	Array	000000000

Figure 20. Selecting lower-level signals

- **2-5-2.** Select the objects (H* and cm0_*) as shown in the above figure and drag them into the waveform window to monitor those lower-level signals.
- **2-5-3.** Select the added signals in the waveform window and change their radix to Hexadecimal.
- **2-5-4.** Close the simulator by selecting **File > Close Simulation**.
- **2-5-5.** Click **OK** and then click **Yes** to close it saving the waveform as **untitled_1**.
- 2-5-6. Click on Run Simulation > Run Behavioral Simulation.
- **2-5-7.** Click on the Zoom Fit button and see various signals, bus activities.



¥			1,000.000 ns
8	Name	Value	10 ne 1200 ne 1200 ne 1200 ne 1200 ne
0+	18 PESET		
0-		0	
	EER LED[7:0]	55	
10.00		0	
~	HADDR[31:0]	0000008c	
M	HBURST[2:0]	0	
Þ		0	
t	HPROT[3:0]	a	
2	# HSIZE[2:0]	2	
	HTRANS[1:0]	2	
E.	HWDATA[31:0]	00f7ff00	
	1/2 HWRITE	0	
-	🖽 📲 HRDATA[31:0]	d1fd1e40	(, (,., (, (,., (,.,.,.,
•	1 HREADY	1	
31	1 HRESP	0	
	⊞ 📲 cm0_r00[31:0]	002ffff7	<u>fffffff</u> <u>00000</u> <u>002ff</u>
	🖽 🔣 cm0_r01[31:0]	50000000	
	₩ cm0_r02[31:0]	fillit	
	🖽 📲 cm0_r03[31:0]	fffffff	
	🖽 📲 cm0_r04[31:0]	mmm	
	⊞ 📲 cm0_r05[31:0]	fttttttt	
	⊞ 📲 cm0_r06[31:0]	fillille	
	⊞ 📲 cm0_r07[31:0]	fillitte	
	₩ dm0_r08[31:0]	fffffff	
	H cm0_r09[31:0]	mmm	
	+ cm0_r10[31:0]		

Figure 21. Simulation output

- 2-5-8. You may want to use zoom in into the simulation window and analyze the output.
- **2-5-9.** Now groups AHB-Lite signals to see the AHB-Lite transactions Address & Control: HADDR, HTRANS, HWRITE,

Address & Control:	HADDR, HTRANS, H
Write Data:	HWDATA,
Read Data	HRDATA,
Slave Response	HREADY

2-5-10. You should now be able to see the AHB-Lite transactions happening like below





www.xilinx.com/support/university

xup@xilinx.com

2-5-11. Close the simulator by selecting **File > Close Simulation**.

END OF LAB PART 1



Synthesize the Design

- **3-1.** Synthesize the design with the Vivado synthesis tool and analyze the Project Summary output.
- **3-1-1.** Make sure you have compiled your software and generated the code.hex in LabPart2/Software folder
- **3-1-2.** Open the project Lab Part2/FPGA/Nexys4/Nexys4.xpr
- 3-1-3. Click on Run Synthesis under the Synthesis tasks of the Flow Navigator pane.

The synthesis process will be run on the AHBLITE_SYS.v file and all its hierarchical files. When the process is completed a *Synthesis Completed* dialog box with three options will be displayed.

3-1-4. Select the *Open Synthesized Design* option and click **OK** as we want to look at the synthesis output before proceeding to the implementation stage.

Click **Yes** to close the elaborated design if the dialog box is displayed.

3-1-5. Select the **Project Summary** tab and understand the various windows.

If you don't see the Project Summary tab then select Layout > Default Layout, or click the

Project Summary icon





Figure 22. Project Summary view

Click on the various links to see what information they provide and which allows you to change the synthesis settings.

3-1-6. Click on the Table tab in the Project Summary tab.

Notice that there are an estimated 850 FFs, 3124 LUTs, one BUFG, and 11 IOs (2 input and 9 output) are used.

FF 850 126800 12000 LUT 3124 63400 90000 9000 9000 9	Resource	Estimation	Available	Utilization %
LUT 3124 63400 S Memory LUT 128 19000 S I/O 11 210 S	FF	850	126800	1
Memory LUT 128 19000 11 I/O 11 210 9	LUT	3124	63400	
I/O 11 210	Memory LUT	128	19000	1
PUEC 2 22	I/O	11	210	5
BUFG 2 32 0	BUFG	2	32	(

Figure 23. Resource utilization estimation summary

3-1-7. In The *Flow Navigator*, under *Synthesis* (expand *Synthesized Design* if necessary), click on **Schematic** to view the synthesized design in a schematic view.

www.xilinx.com/support/university





Figure 24. Synthesized design's schematic view

Notice that IBUFs and OBUFs are automatically instantiated (added) to the design as the input and output are buffered.

Using Windows Explorer, verify that **lab1.runs** directory is created under **lab1**. Under the **runs** directory, **synth_1** directory is created which holds several files related to synthesis.



Figure 25. Directory structure after synthesizing the design

Implement the Design

4-1. Implement the design with the Vivado Implementation Defaults (Vivado Implementation 2013) settings and analyze the Project Summary output.

4-1-1. Click on Run Implementation under the Implementation tasks of the Flow Navigator pane.

The implementation process will be run on the synthesized design. When the process is completed an *Implementation Completed* dialog box with three options will be displayed.

- **4-1-2.** Select **Open implemented design** and click **OK** as we want to look at the implemented design in a Device view tab.
- **4-1-3.** Click **Yes**, if prompted, to close the synthesized design.



The implemented design will be opened. Click **OK** to see the device view.

- **4-1-4.** In the *Netlist* pane, select one of the nets (e.g. n_1_uAHB2LED) and notice that the net displayed in the X1Y1 clock region in the Device view tab (you may have to zoom in to see it).
- **4-1-5.** If it is not selected, click the *Routing Resources* icon **u** to show routing resources.



Figure 26. Viewing implemented design

4-1-6. Close the implemented design view and select the **Project Summary** tab (you may have to change to the Default Layout view) and observe the results.

Select the Post-Implementation tab.

Notice that the actual resource utilization is three LUTs and 16 IOs. Also, it indicates that no timing constraints were defined for this design (since the design is combinatorial).



Pro	ect Summary	×
Z	Project Settings	Edit 🛠
*	Project name: lab1 Product family: Artix-7 Project part: xc7a100tcsg324-1 Top module name: AHBLITE_SYS	
	Synthesis *	Implementation *
	Status:	Status: Image: Complete Messages: Image: Lvarning Part: xc7a100tsg324-1 Strategy: Vvado Implementation Defaults Incremental Compile: None Summary Route Status
DRC Violations		Timing *
	Summary:	Timing information is not available because it hasn't been run Post-Synthesis Post-Implementation
	Utilization *	Power *
	FF 196 UUT 596 Memory LUT 196 J/O 596 BUFG 696 0 25 50 75 100 Utilization (%)	Total On-Chip Power: 0.108 W Junction Temperature: 25.5 °C Thermal Margin: 59.5 °C (12.9 W) Effective d3A: 4.6 °C/W Power supplied to off-chip devices: 0 W Confidence level: Medium
	Post-Synthesis Post-Implementation	Summary On-Chip

Figure 27. Implementation results

Using the Windows Explorer, verify that **impl_1** directory is created at the same level as **synth_1** under the **lab1_runs** directory. The **impl_1** directory contains several files including the implementation report files.

4-1-7. In Vivado, select the **Reports** tab in the bottom panel (if not visible, click *Window* in the menu bar and select **Reports**), and double-click on the *Utilization Report* entry under the *Place Design* section. The report will be displayed in the auxiliary view pane showing resource utilization. Note that since the design is combinatorial no registers are used.

Reports		
Name	Modified	Size
Synth Design (synth_design)		
🚔 📄 Vivado Synthesis Report	4/10/14 7:54 AM	48.9 KB
🗋 Utilization Report	4/10/14 7:54 AM	6.0 KB
Place Design (place_design)		
🗋 Vivado Implementation Log	4/10/14 8:10 AM	20.4 KB
Pre-Placement Incremental		
- 🗋 IO Report	4/10/14 8:09 AM	75.6 KB
Clock Utilization Report	4/10/14 8:09 AM	9.0 KB
🛛 🕒 Utilization Report	4/10/14 8:09 AM	8.2 KB
🖳 📄 Control Sets Report	4/10/14 8:09 AM	6.3 KB
Incremental Reuse Report		
Route Design (route_design)		
🖳 🗋 Vivado Implementation Log	4/10/14 8:10 AM	20.4 KB
🗎 WebTalk Report		
🗠 📄 DRC Report	4/10/14 8:10 AM	7.2 KB
🗠 📄 Power Report	4/10/14 8:10 AM	9.4 KB
🖳 🗎 Route Status Report	4/10/14 8:10 AM	0.6 KB
		15 7 10
📃 Tcl Console 🗋 🔎 Messages 🗌 💐 Lo	g) 📑 Reports 🗌 🖏 De	sign Runs

```
2510. Instantiated Netlists
26
271. Slice Logic
28 -----
29
30 +---
                           31 | Site Type | Used | Loced | Available | Util% |
32 +----+
33 | Slice LUTs | 3027 | 0 | 63400 | 4.77 |

        34
        LUT as Logic
        2899
        0
        63400
        4.57
        35

        35
        LUT as Memory
        128
        0
        19000
        0.67

36 | LUT as Distributed RAM | 128 | 0 |
                                               I I
37 | LUT as Shift Register | 0 | 0 |
                                                   1
                                                           1
38 | Slice Registers | 850 | 0 | 126800 | 0.67 |
39 | Register as Flip Flop | 850 |
                                       0 | 126800 | 0.67 |
                                            126800 | 0.00 |

        40
        Register as Latch
        0

        41
        F7 Muxes
        73

                                       0 |
41 | F7 Muxes
                                       0 |
                                              31700 | 0.23 |
                                            31700 | 0.23 |
15850 | 0.20 |
                                     0 1
                           | 32 |
42 | F8 Muxes
43 +----+
44
```

Figure 28. Viewing utilization report



Generate the Bitstream and Verify Functionality

Step 5

- **5-1.** Connect the board and power it ON. Generate the bitstream, open a hardware session, and program the FPGA.
- **5-1-1.** Make sure that the Micro-USB cable is connected to the JTAG PROG connector (next to the power supply connector).
- **5-1-2.** Make sure that the JP3 is set to select USB power.



Figure 29. Board connection

- **5-1-3.** Power **ON** the switch on the board.
- **5-1-4.** Click on the **Generate Bitstream** entry under the *Program and Debug* tasks of the *Flow Navigator* pane.

The bitstream generation process will be run on the implemented design. When the process is completed a *Bitstream Generation Completed* dialog box with three options will be displayed.

Bitstream Generation Completed			
Bitstream Generation successfully completed.			
Next			
O View Reports			
Open Hardware Manager			
C Launch iMPACT			
Don't show this dialog again			
OK Cancel			

Figure 30. Bitstream generation



www.xilinx.com/support/university

This process will have generated a **AHBLITE_SYS.bit** file under **impl_1** directory in the **lab1.runs** directory.

5-1-5. Select the Open Hardware Manager option and click OK.

The Hardware Manager window will open indicating "unconnected" status.

5-1-6. Click on the Open a new hardware target link.

You can also click on the Open recent target link if the board was already targeted before.

Hardware Session - unconnected	
(i) No hardware target is open. Open recent target	Open a new hardware target

Figure 31. Opening new hardware target

- 5-1-7. Click Next to see the Vivado CSE Server Name form.
- **5-1-8.** Click **Next** with the localhost port selected.

The JTAG cable which uses the Xilinx_tcf should be detected and identified as a hardware target. It will also show the hardware devices detected in the chain.

Select Hardware Target Select a hardware target from the list of available targets on the Vivado CSE Server (vcse_server). Hardware Targets Type Port ESN Image: Select a hardware target from the list of available targets on the Vivado CSE Server (vcse_server). Hardware Targets Type Port ESN Image: Server is the point of the point is the point of the p	🚴 Open New Hardware Target	
Hardware Targets Type Port ESN Image: state sta	Select Hardware Target Select a hardware target from the list of available targets on the Vivado CSE Server (vcse_server).	>
Type Port ESN Image: state st	Hardware Targets	
Wilinx_tcf Digilent/210274992934A Hardware Devices Name ID Code IR Length XC7A100T_0 03631093 Server: Incalhost:60001 Version:	Type Port ESN	
Hardware Devices Name ID Code IR Length	xilinx_tcf Digilent/210274992934A	
Hardware Devices Name ID Code IR Length Image: Normal Server: localbost:60001 Version: 20		
Hardware Devices Name ID Code IR Length		
Hardware Devices Name ID Code IR Length Image: Server: localhost:60001_Version: 20		
Hardware Devices Name ID Code IR Length Image: Server: localbost:60001 Version: 20		
Name ID Code IR Length > XC7A100T_0 03631093 6	Hardware Devices	
XC7A100T_0 03631093 6 Server: localhost:60001_Version: 20	Name ID Code IR Length	
Server: localhost:60001_Version: 20	XC7A100T_0 03631093 6	
Server: localhost:60001_Version: 20		
	Server: localhost:60001_Version: 20	
		_
< Back Next > Finish Cancel	< Back Next > Finish Cancel]

Figure 32. New hardware target detection

5-1-9. Click Next twice and then Finish.

The Hardware Session status changes from Unconnected to the server name and the device is highlighted. Also notice that the Status indicates that it is not programmed.

www.xilinx.com/support/university



Hardware Manager - localhost/xilinx_tcf/Digilent/210274992934A			
(i) There are no debug cores. Program device	e <u>Refresh device</u>		
Hardware	_ 🗆 🖻 ×		
< 🔀 🖨 🛃 ▶ ▶ ■			
Name	Status		
□· localhost (1)	Connected		
xilinx_tcf/Digilent/210274992934A	Open		
XC7A100T_0 (0) (active)	Not programmed		
-			
•	•		
Hardware 💡 Templates			

Figure 33. Opened hardware session

5-1-10. Select the device and verify that the AHBLITE_SYS.bit is selected as the programming file in the General tab.



Figure 34. Programming file

5-1-11. Right-click on the device and select *Program Device…* or click on the *Program device* > *XC7A100T_0* link to program the target FPGA device.



Hardware Manager - localhost/xilinx_tcf/Digilent/210274992934A					
There are no debug cores. Program device <u>Refresh device</u>					
Hardware _ 🗆 🖻 🗵				Debug Probes _	
				🔍 🛣 🖨 🖪	
Name		Status			
Connected					
xlinx_tcf/Digilent/210274992934A Open					
	Hardware Device Pro				
	Assign Program			ng File	
Hardware 🖓 Templates	٠	Program Device			
Hardware Device Properties	Refresh Device				
← → <u>©</u> <u></u> ∞ XC7A100T_0		Export to Spreads	hee	t	

Figure 35. Selecting to program the FPGA

5-1-12. Click **OK** to program the FPGA.

The DONE light will light when the device is programmed. You may see LEDs toggling if the SW15 switch position is OFF (down position).

- **5-1-13.** Turn ON the SW15 switch (up position) which will put the CPU in a reset state and you won't see the LEDs toggling.
- **5-1-14.** Turn OFF the SW15 switch which will put the CPU in a run state and you will see the LEDs toggling.
- **5-1-15.** When satisfied, power **OFF** the board.
- **5-1-16.** Close the hardware session by selecting **File > Close Hardware Manager**.
- **5-1-17.** Click **OK** to close the session.
- **5-1-18.** Close the **Vivado** program by selecting **File > Exit** and click **OK**.

Conclusion

The Vivado software tool can be used to perform a complete design flow. The project was created using the supplied source files (HDL model and user constraint file). A behavioral simulation using the provided testbench was done to verify the model functionality. The model was then synthesized, implemented, and a bitstream was generated. The functionality was verified in hardware using the generated bitstream.

