ALL PROGRAMMABLE

Vivado Design Flow

Vivado 2013.4 Version

This material exempt per Department of Commerce license exception TSU

© Copyright 2014 Xilinx

Objectives

> After completing this module, you will be able to:

- Explain how the design analysis features of the Vivado IDE can help in FPGA design development
- List the main features of the Vivado IDE
- Describe the Vivado IDE Design flow
- Introduce the scripted Vivado IDE design flows

Outline

- > Vivado IDE Features and Benefits
- > Vivado Design Suite Introduction
- > Vivado Design Flow
- **>** Summary

Vivado IDE Solution

> Interactive design and analysis

 Timing analysis, connectivity, resource utilization, timing constraint analysis, and entry

> RTL development and analysis

- Elaboration of HDL
- Hierarchical exploration
- Schematic generation
- > XSIM simulator integration
- > Synthesis and implementation in one package
- I/O pin planning
 - Interactive rule-based I/O assignment

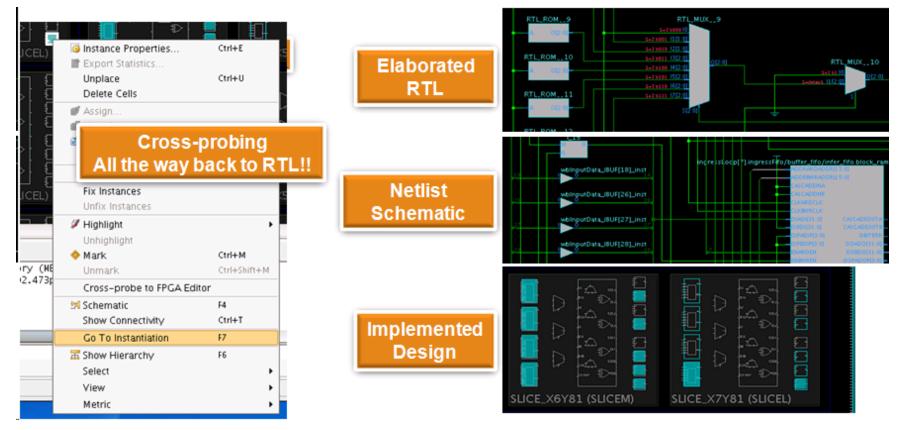
File Edit Flow Tools Window	Layout View Help	Q- Search commands
🯕 📄 🖩 i in 🕫 🐂 🐂 🗙 🤇	🕅 🕨 🐮 🚰 🥝 🧐 🐝 ∑ 🤪 😬 Default Layout 💿 🗶 🔖 🔭	E Implementation Complet
Flow Navigator «	Implemented Design - constrs_2 xc7k70tfbg676-2 (active)	3
🖫 🖨	Netlist _ 🗆 🗠 × 🔽 Project Summary 🗙 🐼 Device 🗙	□ Ŀ ×
 Project Hanager Project Settings Project Settings Add Sources Simulation Simulation Settings Run Simulation Netlist Analysis Performance Open Synthesized Design Implementation Run Implementation Settings Run Implementation Implementation Constraints Gent Timing Constraints Report Timing Summary Report Clock Networks 	Image: Second	
🛐 Report Clock Interaction	Design Runs	- D & X
Report DRC Report Noise Report Utilization Report Utilization Report Power Program and Debug Bitstream Settings Generate Bitstream	Q Name Part Constraints Strategy Impl_1 xc7k70tfbg676-2 constrs_2 Vivado Implementat Impl_1 impl_2 impl_2 impl_2	ion Defaults (Vivado Implementation 201
Open Hardware Session		
Launch iMPACT	· · · · · · · · · · · · · · · · · · ·) ∢

Hierarchical Design Analysis and Implementation Environment

Vivado's Visualization Feature

> Visualize and debug your design at any flow stage

- Cross-probing between netlist/schematic/RTL



Tcl Features

> Tcl Console enables the designer to actively query the design netlist

> Full Tcl scripting support in two design flows

- Project-based design flow provides easy project management by the Vivado IDE
- Non-project batch design flow enables entire flow to be executed in memory
- Journal and log files can be used for script construction
- > Tcl scripting is covered in the last module of the course



Outline

- > Vivado IDE Features and Benefits
- > Vivado Design Suite Introduction
- > Vivado Design Flow
- **>** Summary

Typical vs Vivado Design Flow

Interactive IP plug-n-play environment

– AXI4, IP_XACT

Common constraint language (XDC) throughout flow

- Apply constraints at any stage

Reporting at any stage

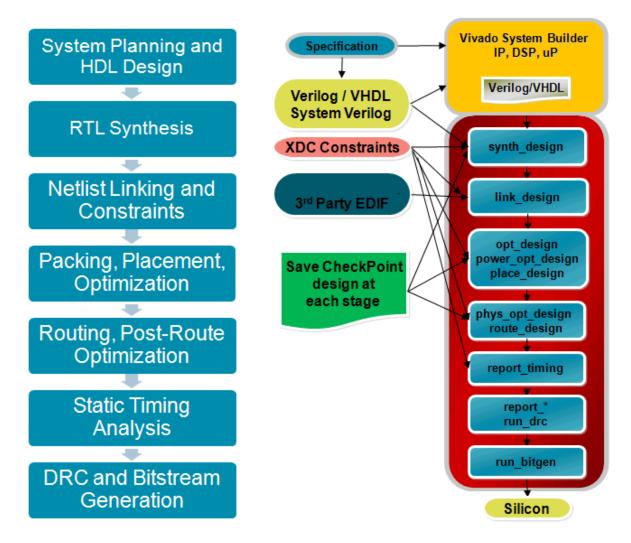
Robust Tcl API

Common data model throughout the flow

- "In memory" model improves speed
- Generate reports at all stages

Save checkpoint designs at any stage

- Netlist, constraints, place and route results



What is a Netlist?

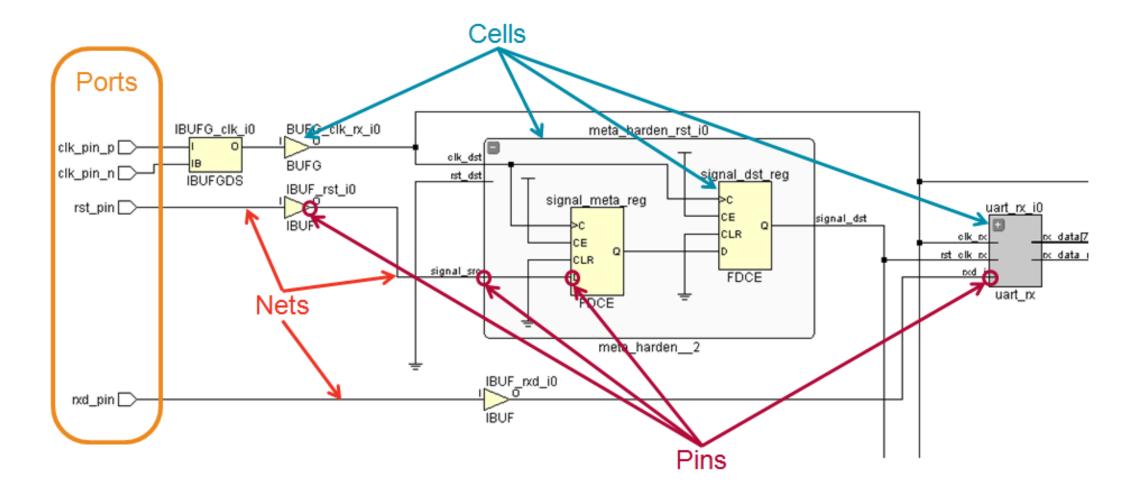
A netlist is a description of your design

Consists of cells, pins, port and nets

- Cells are design objects
 - Instances of user modules/entities
 - Instances of library elements (BELs)
 - LUTs, FF, RAMs, DSP cells, etc...
 - Generic technology representations of hardware functions
 - Black boxes
- Pins are connection points on cells
- Ports are the top level ports of your design
- Nets make connections between pins and from pins to ports



Netlist Objects

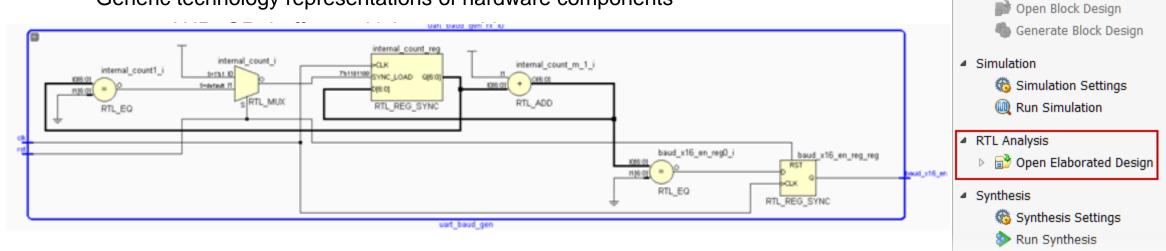


Elaborated Design



> Representation of the design before synthesis

- Interconnected netlist of hierarchical and generic technology cells
 - Instances of modules/entities
 - Generic technology representations of hardware components



Flow Navigator

▲ IP Integrator

Project Manager

Project Settings
Add Sources

👫 Create Block Design

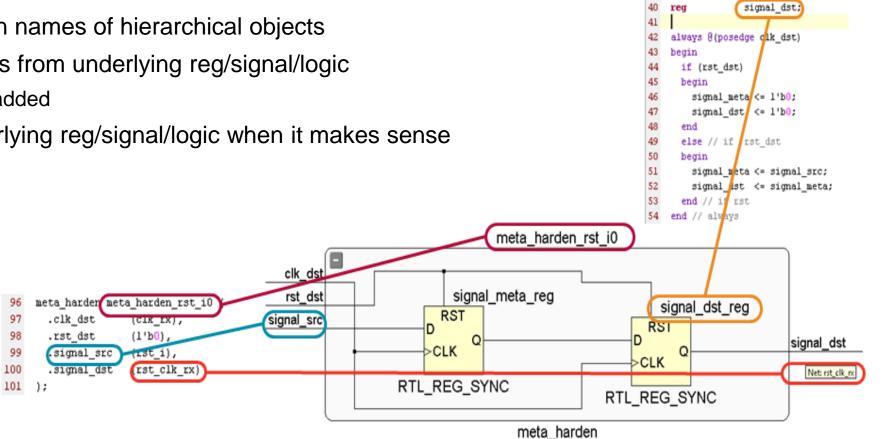
IP Catalog

🔍 🔀 🍃

Object Names in Elaborated Design

Object names are extracted from RTL

- Instance and pin names of hierarchical objects
- Inferred flip-flops from underlying reg/signal/logic
 - Suffix _reg is added
- Nets from underlying reg/signal/logic when it makes sense



39

req

signal neta;

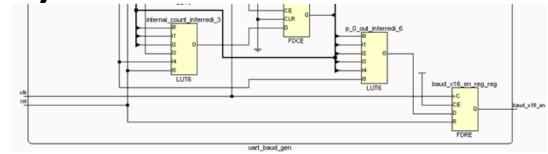
Synthesized Design

> Accessed through the Flow Navigator by selecting Open Synthesized Design

> Representation of the design after synthesis

- Interconnected netlist of hierarchical and basic elements (BELs)
 - Instances of modules/entities
 - Basic elements
 - LUTs, flip-flops, carry chain elements, wide MUXes
 - Block RAMs, DSP cells
 - Clocking elements (BUFG, BUFR, MMCM, …)
 - I/O elements (IBUF, OBUF, I/O flip-flops)

Object names are the same as names in the elaborated netlist when possible



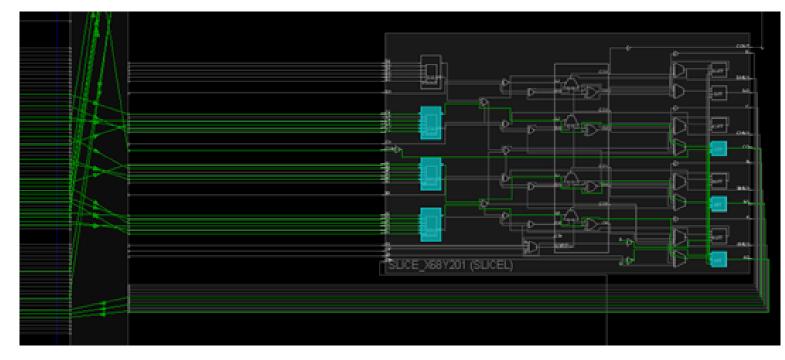


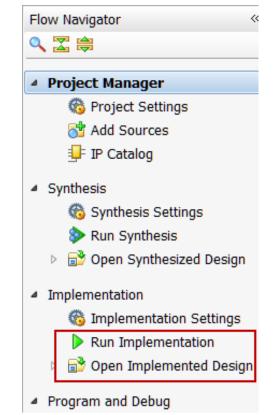
Implemented Design

> Accessed through the Flow Navigator by selecting Open Implemented Design

> Representation of the design during and after the implementation process

- Structurally similar to the Synthesized Design
- Cells have locations, and nets are mapped to specific routing channels





Project Data

> All project data is stored in a project_name directory containing the following

- project_name.xpr file: Object that is selected to open a project (Vivado IDE project file)
- project_name.runs directory: Contains all run data
- project_name.srcs directory: Contains all imported local HDL source files, netlists, and XDC files
- project_name.data directory: Stores floorplan and netlist data

Journal and Log Files

Journal file (vivado.jou)

- Contains just the Tcl commands executed by the Vivado IDE

Log file (vivado.log)

 Contains all messages produced by the Vivado IDE, including Tcl commands and results, info/warning/error messages, etc.

Location

- Linux: directory where the Vivado IDE is invoked
- Windows via icon: %APPDATA%\Xilinx\Vivado or C:\Users\<user_name>\AppData\Roaming\Xilinx\Vivado
- Windows via command line: directory where the Vivado IDE is invoked
- From the GUI
 - Select File > Open Log File
 - Select File > Open Journal File

Outline

- > Vivado IDE Features and Benefits
- > Vivado Design Suite Introduction
- > Vivado Design Flow
- **>** Summary

Getting Started Jump Page

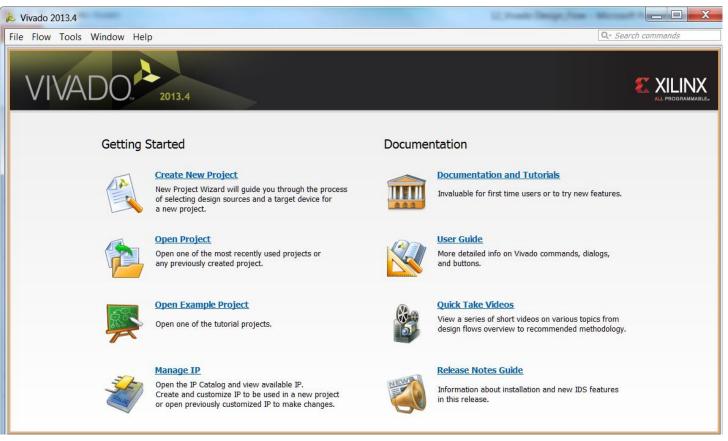
> Links to create new or open existing projects

> Open example projects

- Used for tutorials

Documentation links

- Invokes PDF viewer for
 - Release notes
 - User guide
 - Methodology guide
 - Tutorials

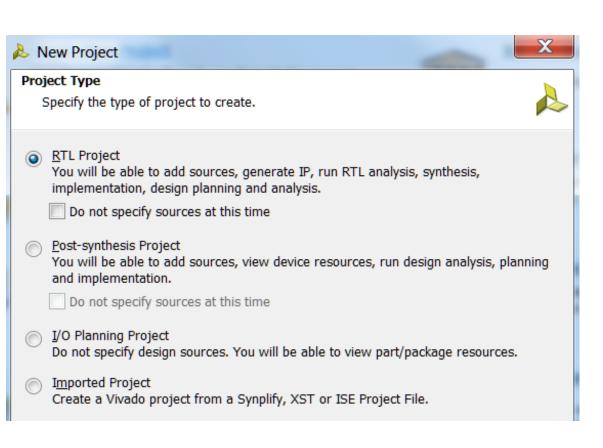


New Project Creation Wizard

Four different types of project can be created

- RTL
 - Front-to-back
- Post-synthesis
 - EDIF or NGC
- I/O planning
 - For early pin testing
 - No design sources
- Import Project existing project from Project Navigator, XST, and Synplify Projects

Next, add source files, constraints files, and select an FPGA



Creating a Project with HDL Sources and/or Synthesized Design

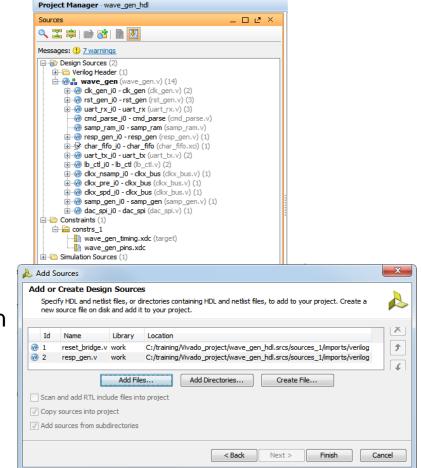
> Define the project name and location

Select source files in RTL project creation

- All recognized source files, Verilog, VHDL, in the directory and subdirectories, can be added
- Select post-synthesized netlist in Post-synthesized project creation
 - All synthesized files in the directory and subdirectories, are added

Select constraint files

- One or more constraints files including IP specific and top-level can be added
- > Select the family and target device, or pre-defined board
- Reference original existing files or import and copy them into the project



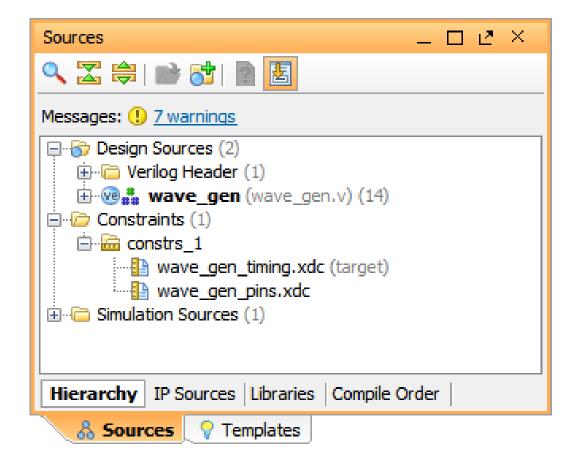
Constraints File Management

Constraint sets are a collection of XDC files

- A project can contain multiple constraint sets but all must be of the same type
- For a constraints set to be applied, it must be set to "active"
- Any constraint set can be made active by rightclicking and selecting Make Active

Target XDC

- The XDC file in a constraint set to which NEW constraints are written
- Modified existing constraints are written back to the original constraint file



Project Manager

- Used to manage sources, customize IP, and view project details in the Project Summary
- > Flow Navigator

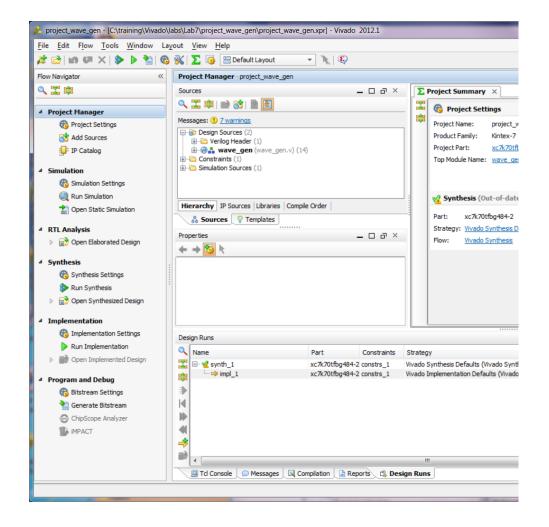
Sources view

- Hierarchical display of sources
- IP Sources and Libraries view
 - HDL and netlists including references to library and location
- Gives access to constraints file

Project Summary

 Gives access to device utilization (resources), timing summary, and strategy information

Tcl Console, Messages, Compilation, Reports, and Design Runs



Vivado Design Flow- 25

Project Settings

> General settings

- Select device

- Target HDL language
- Top module name
- Language options
- Other settings are covered in their respective modules

Project Settings	X
90	General
<u>G</u> eneral	Name: my_and2_test
	Project device: 🔷 xc7a100tcsg324-1 (active)
Simulation	Target language: Verilog 🔹
	Top module name: system_wrapper 📀 📼
Synthesis	Language Options
	Verilog options: verilog_version=Verilog 2001
Implementation	Generics/Parameters:
1010	Top library:
<u>B</u> itstream	Loop count: 1,000 🤤
Īb	

Flow Navigator – RTL Project

Configure project sources

- Add HDL source files, constraints files, simulation files, block designs

> IP Integrator

- Create, open, generate a block design

Run Simulation

- XSIM simulator included
- Behavioral, post-synthesis, post-implementation

RTL Analysis

- Open Elaborated Design button: Loads the elaborated RTL design

> Run Synthesis

- Timing driven
- Open Synthesized Design button: Loads synthesized netlist



XILINX > ALL PROGRAMMABLE.

Flow Navigator – RTL Project

> Run Implement button: Runs implementation tools

- link, opt, power_opt, place, phys_opt, and route
- Open Implemented Design button: Loads implemented design

Program and Debug: Launches programming and debugging tools

- Open hardware session to program the FPGA
- Also can use iMPACT program if installed to program the FPGA

	Flow Navigator «
	Sec. 2 (1)
	Project Manager
	🊳 Project Settings
•	👌 Add Sources
5	🕂 IP Catalog
	▲ IP Integrator
	Open Synthesized Design
	 Implementation Implementation Settings
	Run Implementation
	 Den Implemented Design
	Program and Debug
	🊳 Bitstream Settings
	🔚 Generate Bitstream
	💕 Open Hardware Session
	🛞 Launch iMPACT

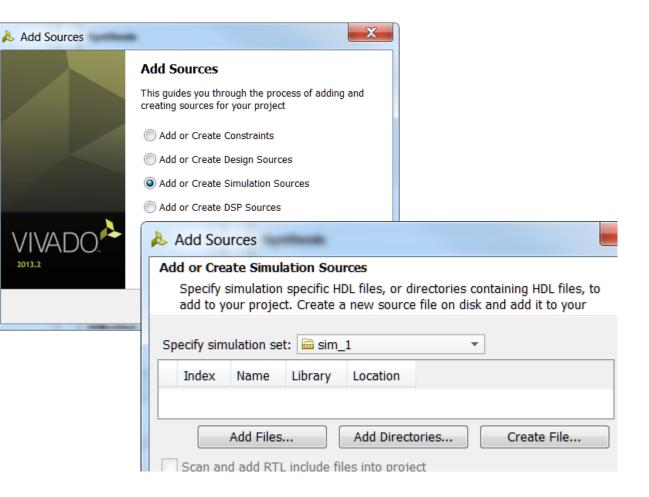
Simulation

- The Vivado simulator, XSIM, supports RTL, netlist, and timing simulation
- Sraphical waveform display
- Toolbar buttons for adding markers, measuring delays, and zooming
- Buses can be expanded to view individual signals
- Dividers can be inserted to visually isolate groups of related signals
- By default, the top-level signals are displayed

Edit Flow Simulation Tools	Window Layout View	Help									Q + Search commands
😂 🕼 🖉 🗙 🗞 🍋 🎦	📢 🕨 🕅 🕞 🚾 💵 🛛 🌀	8 🕺 🗵 🗔	😬 Default Layou	it 👻 🎉 🔍	E)					Synthesis and Imp	plementation Out-of-Date more
Navigator «	Simulation - wave_gen_	behav - wave_ge	n								
Z 🖨	Scopes		@ ~ ×	Objects		_ 0 a ×	🔄 Untit	led 1* ×			00
		F(x) Ch		16 16 16 16 16	18 1		₹				1,000.000 n
Project Manager	Name	Design Unit	Dia da Tara	Name	Value	Data Ta					
Project Settings			Block Type	a construction of the second sec		Data Typ	*	Name	Value	0 ns	500 ns
Add Sources	e- 📒 wave_gen	wave_gen	Verilog Mo 🔺	- U dk_pin_p	Z	Logic A	Q- 1	ī_dk_pin_p	z		
IP Catalog	⊕ Gk_gen_i0	dk_gen rst_gen	Verilog Mo	- U rst_pin	Z	Logic Logic	2	dk_pin_n	7		
-	⊕ guart_rx_i0	uart_rx	Verilog Mo	1 rxd_pin	Z	Logic	3		4		
🐣 IP Packager		cmd parse	Verilog Mo	- U txd_pin	X	Logic		rst_pin	Z		
	E samp_ram_i0	samp_ram	Verilog Mo	- U lb_sel_pin	z	Logic	4	🔤 📊 rxd_pin	Z		
mulation	resp_gen_i0	resp_gen	Verilog Mo	- U spi_ck_pin	0	Logic	5	txd_pin	x		
🔞 Simulation Settings	- Greap_gen_io	char_fifo	Verilog Mo	- U spi_mosi_pin	x	Logic E	- C	 b_sel_pin	z		
(Run Simulation		uart_tx	Verilog Mo	- la dac_cs_n_pin	x	Logic	2 °				
~	⊕ 🔒 lb_ctl_i0	lb_ctl	Verilog Mo	- 4 dac_dr_n_pin	x	Logic		spi_dk_pin	U		
📸 Open Static Simulation	🕀 📒 clkx_nsamp_i0	dkx_bus	Verilog Mo	🗄 📑 led_pins[7:0]	XXXXXXXXX	Array	8	👷 spi_mosi_pin	x		
	🕀 📒 dkx_pre_i0	dkx_bus	Verilog Mo	- la rst_i	x	Logic	9	-l _m dac_cs_n_pin	x		
L Analysis	🕒 🧧 dkx_spd_i0	dkx_bus	Verilog Mo	- 1a rxd i	X	Logic	10	dac_dr_n_pin	x		
Open Elaborated Design	🕀 📒 samp_gen_i0	samp_gen	Verilog Mo	- 1% txd o	x	Logic	. 11	🛄 🚟 led_pins[7:0]	X000000X		200000000
	🕀 🔛 dac_spi_i0	dac_spi	Verilog Mo	- 1 b_sel_i	x	Logic			*******		
nthesis			•	- 1 spi_ck_o	0	Logic	12	irst_i	x		
🚯 Synthesis Settings	Scope & Sour	rec		- 1 spi_mosi_o	x	Logic	13	le: rxd_i	x		
	an Scope too Sour			- la dac_cs_n_o	х	Logic	14	We txd_o	x		
Run Synthesis	: Simulation Scope Propertie	s	×	🗈 🍕 led_o[7:0]	XXXXXXXXXX	Array	15]∲ lb_sel_i	v		
Open Synthesized Design	← → 100 k			- llg dk_rx	0	Logic	16		2		
				13 dk_tx	0	Logic		ik spi_dk_o	Q		
plementation	wave_gen			- 1 clk_samp	0	Logic	17	le spi_mosi_o	x		
Careful and the second second second]	- 1 en_ck_samp	X	Logic	18		х		
Model Implementation Settings	Name: /wave_gen	1		- 1 clock_locked	0	Logic	19		XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX		200000000
Run Implementation	Design Unit: wave_gen			- 1% rst_clk_rx	1	Logic	20	- We dk_rx	6		
Open Implemented Design	Block Type: Verilog Mod	tule		- 1% rst_clk_tx	1	Logic			v		
and open implemented beingin			project wave gen	- 1% rst_clk_samp	1	Logic	21	iktx	0		
ogram and Debug	File: <u>c:/training/</u>	vivado/labs/lab7/	project wave gen	- 1% rxd_clk_rx	х	Logic	22		0		
				- 1 rx_data_rdy	х	Logic	- 12		<u>.</u>		
🚳 Bitstream Settings				🕀 🌱 rx_data[7:0]	XXXXXXXXXX	Array					
Cenerate Bitstream				1 send_char_val	X	Logic 🚽					
ChipScope Analyzer	۰ III.		•	4	III	•		<.	+	► <	
IMPACI	Td Console										_ 0 7
	Vivado Simul	ator does no	t vet support	tracing of parame	ters.						
	WARNING: Beh		ls for indepen	dent clock FIFO o	configurat	ions do not m	nodel synd	chronization delays. The	behavioral models a	are functionally cor	rect, and will represent t
				Design snapshot				derajor inc	bendvibidi modelo (ite ranooronarry oor	reout and will represent
				26 . Memory (MB)				19			
		11mc (0). (Hemory (HD)	· peak -	0/01/07/ / gui					
	×										
	<										۲
	Type a Tcl com	mand here									
	Tcl Console										
	TCI COIISOle	messages									

Adding Testbench Files

- > In the Flow Navigator, click Add Sources
- Select Add or Create Simulation Sources option and click Next
- Click on Add Files..., Add directories... buttons if the testbench file is already available or click on Create File... button to create a new testbench file
- Select a file type- Verilog, Verilog Header, SystemVerilog, or VHDL
- Browse to an existing testbench or enter a filename to create



Simulation Settings

- From the Flow Navigator, click Simulation Settings
- Allows selection of compilation and simulation properties
- Additional options can be entered via More Compilation Options field in Compilation tab and More Simulation Options field in Simulation tab
- Refer to the Vivado Design Suite Simulation Guide (UG900) for more information

<u></u>	Simulation
W	
<u>G</u> eneral	Target simulator: Vivado Simulator 💌
<u> </u>	Simulation set:
Simulation	Simulation top module name: two_digits_counter_on_2_7segment_dis 💿 📻
	Clean up simulation files
Synthesis	Compilation Simulation Netlist Advanced
	Verilog options:
Implementation	Generics/Parameters options:
1010	debug typical
Bitstream	mt auto
-	SDF Delay Sdfmax
<u>-</u>	rangecheck
IP	Enable fast simulation models
	More Compilation Options
	Select an option above to see a description of it

Outline

- > Vivado IDE Features and Benefits
- > Vivado Design Suite Introduction
- > Vivado Design Flow
- **>** Summary

Summary

> Features and benefits of the Vivado IDE include

- Performance predictability
- Design analysis features that speed a designer's ability to gain timing closure
- Tcl features (commands) that make scripting easier and powerful

> Vivado tools use a common data model throughout the FPGA design process

- This yields runtime and memory resource benefits to the user

> Vivado tools support scripting in non-project batch and project-based design flows

- Vivado tools support the use of Tcl for all commands

> Vivado tools use a common constraint language (XDC) throughout the design process

- This enables synthesis optimization significantly better than the ISE software

Pushbutton flows for most designs