Digital IC Digital Design and Verification

Lab Manual # 08 – FPGA Prototyping

Release: 1.0

Date: 07-July-2024

NUST Chip Design Centre (NCDC), Islamabad, Pakistan

Revision History

Revision Number	Revision Date	Revision By	Nature of Revision	Approved By
1.0	07/07/2024	Saad Khan	Complete Manual	-

Contents

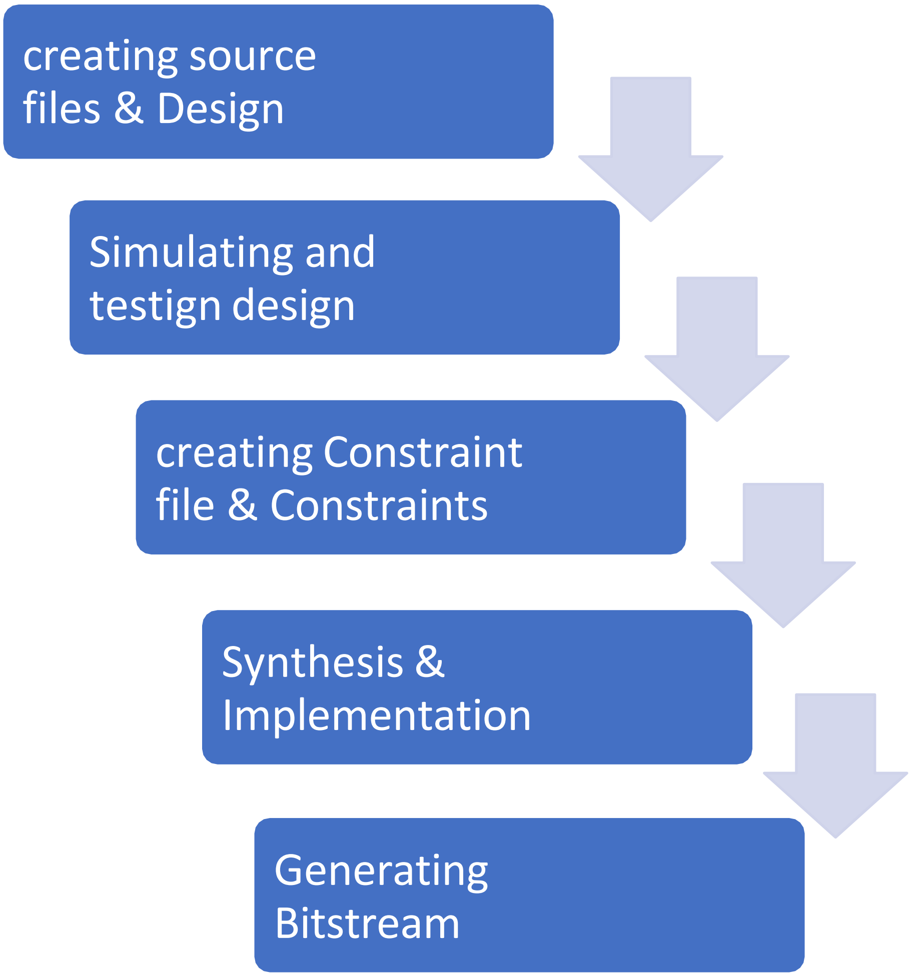
Overview

This handout will guide you through using Vivado 2019 to synthesize, implement, analyze, and generate a bitstream for a 4x1 multiplexer. You will also learn how to connect the bitstream to a NEXUS A7 FPGA board. Additionally, you'll receive home tasks to further your understanding.

Tools

- Xilinx Vivado

Overflow:



creating source files & Design

Simulating and testign design

creating Constraint file & Constraints

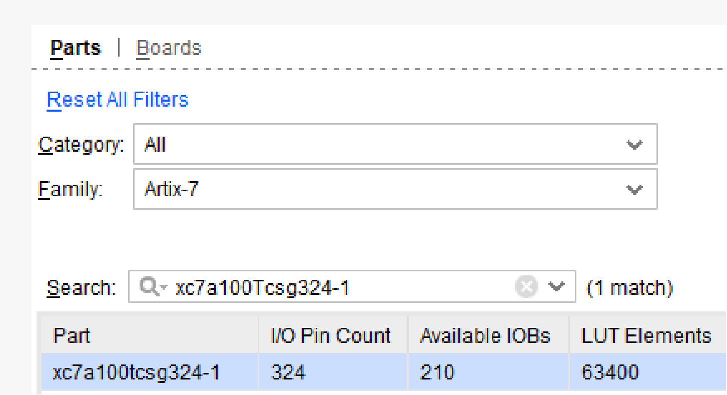
Synthesis & Implementation

Generating Bitstream

Step-by-Step Guide:

Opening Vivado 2019

1.1. Launch Vivado 2019 from your computer.
1.2. Create a new project: File > Project > New.
1.3. Follow the wizard to name your project and choose its location.
1.4. Select RTL Project, in Project Type Menu.
1.5. Check, “Do not Specify sources at this time”.
1.6. Press Next.
1.7. Select Family as “Artix 7” board as **“xc7a100Tcsg324-1”** FPGA Board.



Category: All

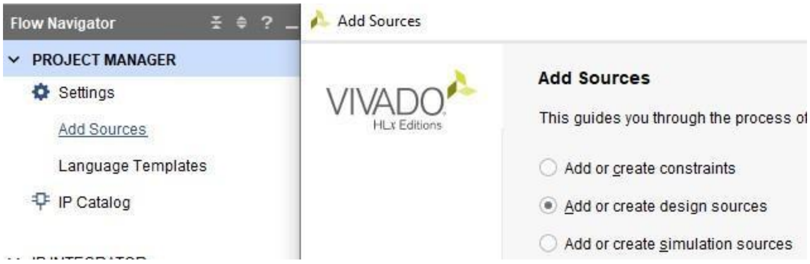
Family: Artix-7

Search: xc7a100Tcsg324-1 (1 match)

Part	I/O Pin Count	Available IOBs	LUT Elements
xc7a100tcsg324-1	324	210	63400

Create the design:

2.1. Add Design source file

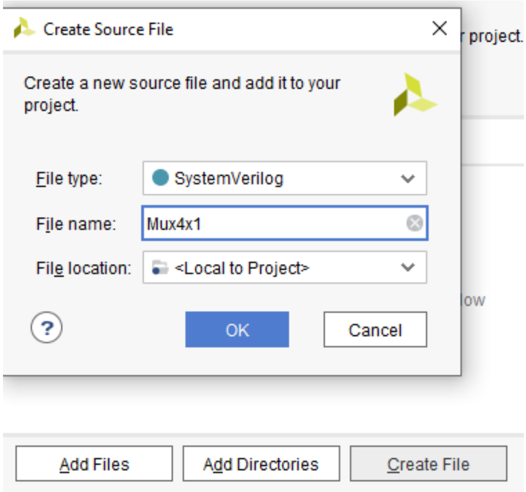


Add Sources

This guides you through the process of

○ Add or create constraints
● Add or create design sources
○ Add or create simulation sources

2.2. Create a “Mux4x1” Module, change file type to System Verilog.



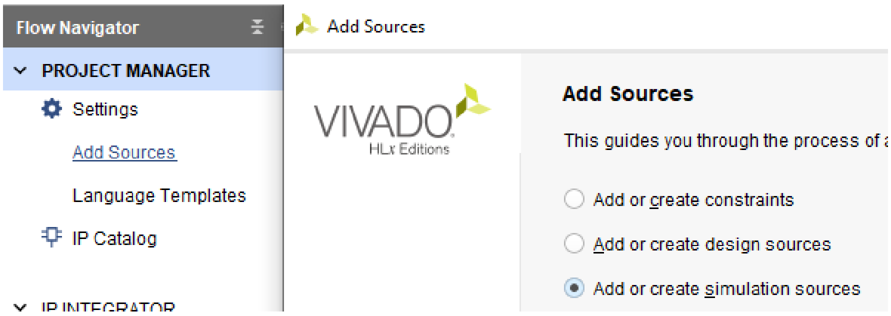
Create a new source file and add it to your project.

File type: SystemVerilog

File name: Mux4x1

File location: <Local to Project>

2.3. Add Simulation Source file.

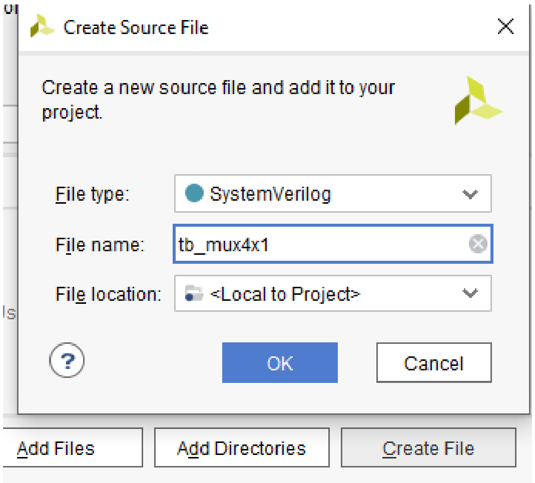


Add Sources

This guides you through the process of a

○ Add or create constraints
○ Add or create design sources
● Add or create simulation sources

2.4. Create a testbench, “tb_mux4x1”, change file type to System Verilog.



Create a new source file and add it to your project.

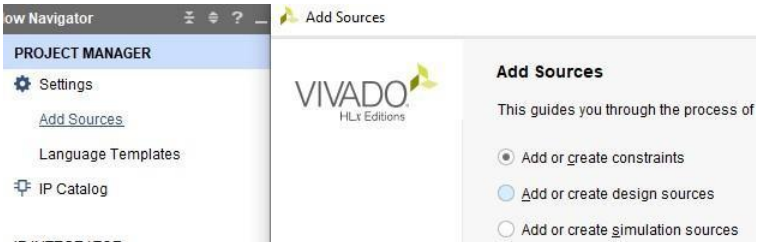
File type: SystemVerilog

File name: tb_mux4x1

File location: <Local to Project>

Add CLK Constraint:

1. Add a constrain Source file.

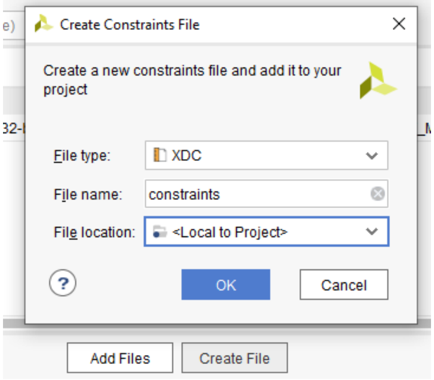


Add Sources

This guides you through the process of

● Add or create constraints
○ Add or create design sources
○ Add or create simulation sources

2. Create the Constraint file.



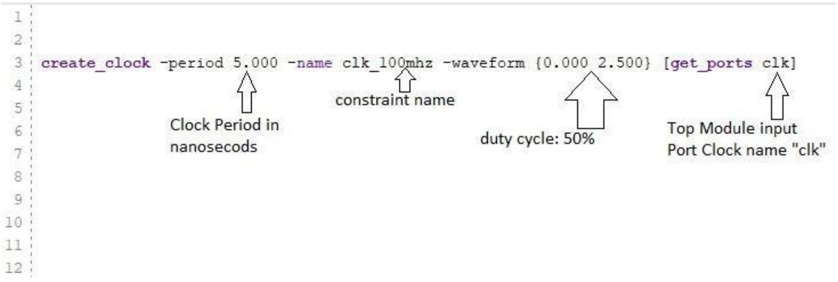
Create a new constraints file and add it to your project

File type: XDC

File name: constraints

File location: <Local to Project>

3. Write the clock constraint.

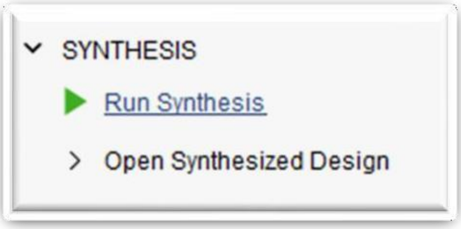


```
create_clock -period 5.000 -name clk_100mhz -waveform {0.000 2.500} [get_ports clk]
```

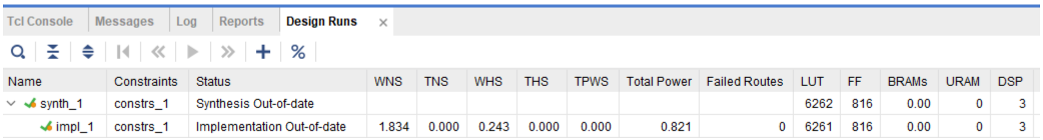
Clock Period in nanosecods
constraint name
duty cycle: 50%
Top Module input Port Clock name "clk"

Synthesizing the Design:

4.1. Click on Run Synthesis.



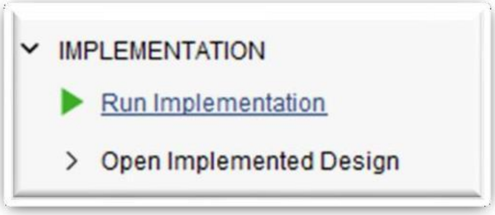
4.2. Review the synthesis report under Design Runs Tab, to ensure there are no errors.



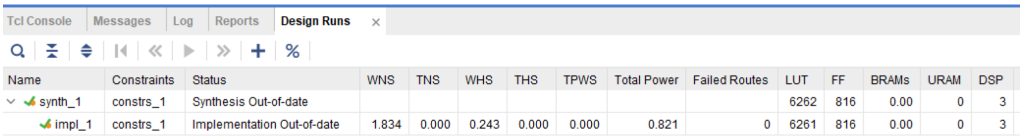
Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAMs	URAM	DSP
✓ synth_1	constrs_1	Synthesis Out-of-date								6262	816	0.00	0	3
✓ impl_1	constrs_1	Implementation Out-of-date	1.834	0.000	0.243	0.000	0.000	0.821	0	6261	816	0.00	0	3

Implementing the Design

5.1. Click on Run Implementation.



5.2. Review the implementation report for any issues.

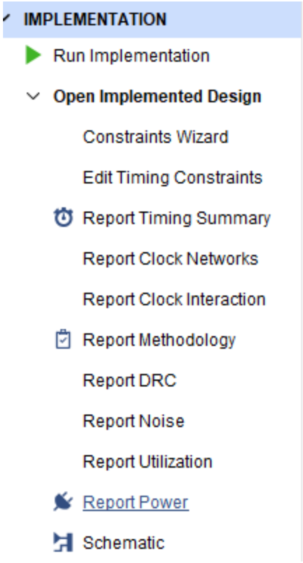


Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAMs	URAM	DSP
✓ synth_1	constrs_1	Synthesis Out-of-date								6262	816	0.00	0	3
✓ impl_1	constrs_1	Implementation Out-of-date	1.834	0.000	0.243	0.000	0.000	0.821	0	6261	816	0.00	0	3

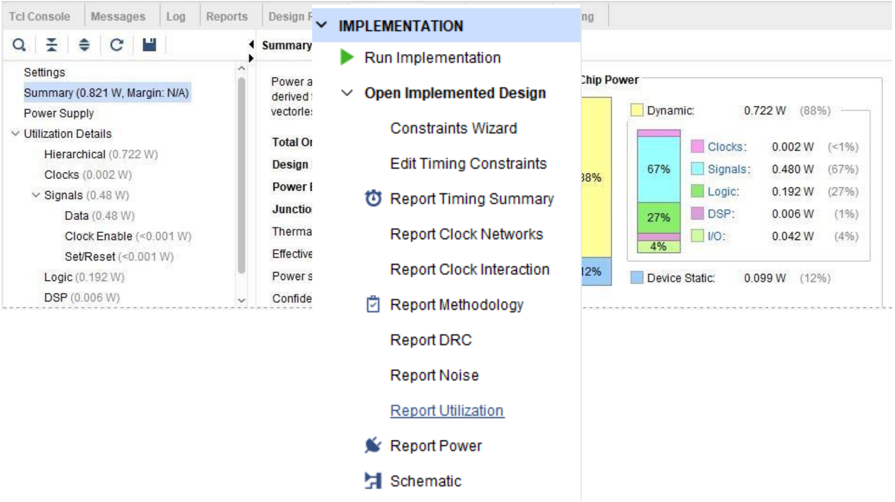
Analyzing Power & Area Analysis

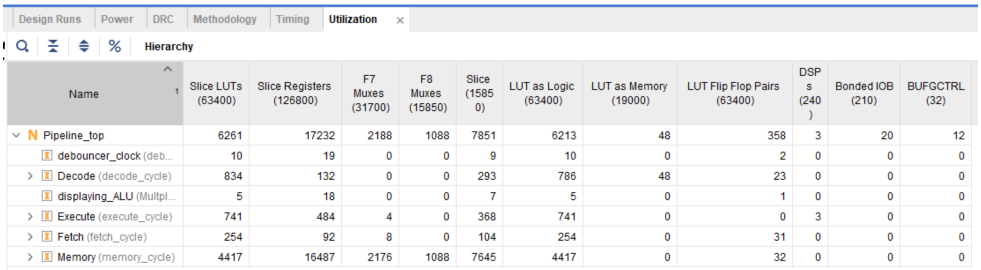
6.1. After implementation, navigate to the Reports section, Under Implementation.

6.2. Generate a power analysis report: Reports > Report Power.



6.3. Generate an area analysis report: Reports > Report Utilization.

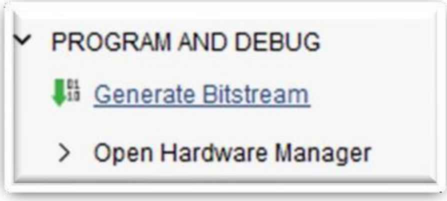




Name	Slice LUTs (63400)	Slice Registers (126800)	F7 Muxes (31700)	F8 Muxes (15850)	Slice (15850)	LUT as Logic (63400)	LUT as Memory (19000)	LUT Flip Flop Pairs (63400)	DSPs (240)	Bonded IOB (210)	BUFGCTRL (32)
Pipeline_top	6261	17232	2188	1088	7851	6213	48	358	3	20	12
debouncer_clock (deb...	10	19	0	0	9	10	0	2	0	0	0
Decode (decode_cycle)	834	132	0	0	293	786	48	23	0	0	0
displaying_ALU (Multpl...	5	18	0	0	7	5	0	1	0	0	0
Execute (execute_cycle)	741	484	4	0	368	741	0	0	3	0	0
Fetch (fetch_cycle)	254	92	8	0	104	254	0	31	0	0	0
Memory (memory_cycle)	4417	16487	2176	1088	7645	4417	0	32	0	0	0

Generating Bitstream

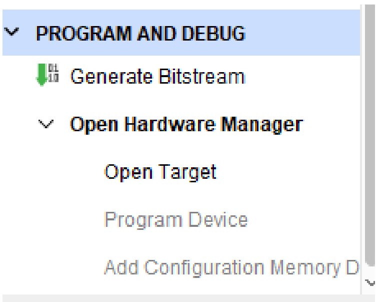
7.1. Click on Generate Bitstream.



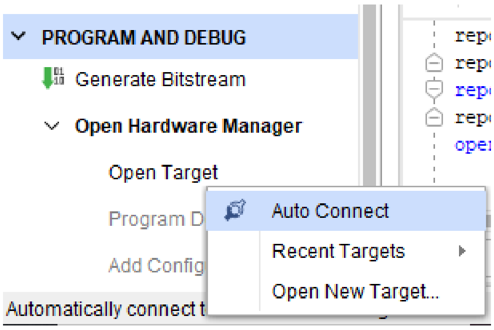
7.2. Once the bitstream is generated, ensure there are no errors in the process.

Connecting to the NEXUS A7 FPGA Board.

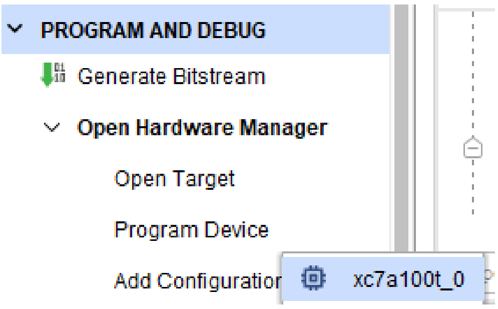
8.1. Connect your NEXUS A7 FPGA board to your computer via USB.
8.2. Open the Hardware Manager in Vivado.



8.3. Click Open Target and select Auto Connect.



8.4. Program the FPGA with your bitstream: Program Device > xc7a35t_0.



Rubrics for FPGA Prototyping

Criteria	Description	Marks Distribution(Out of 15)
Synthesis and Implementation		
Successful Synthesis	Design synthesizes without Warnings, zero errors	1
Successful Implementation	Design implementation without Warnings, zero errors	1
Hardware Utilization		
Minimal Hardware Used	Less hardware resources utilized	2
Area Efficiency	Less area consumed by the design	2
Warnings and Errors		
Minimal Warnings	Fewer warnings during synthesis and implementation	2
No Errors	No errors during synthesis and implementation	1
Power Consumption		
Low Power Usage	Less power consumed by the design	2
Testbench and Verification		
Functional Verification	Testbench verifies all functionality correctly	1
Documentation and Submission		
Proper Documentation	Detailed and clear documentation provided	1
Correct Submission Format	Submitted as a zip folder with the correct naming convention	1
VIVA		5

Submission Instructions

Submit your complete project folder as a zip file following the naming convention: **DLD-Lab5-FPGA-Prototyping-StudentName-TaskX**

Ensure your submission includes:

- All source files (.sv)
- Testbench files
- Synthesis and implementation reports
- Power and area analysis reports
- Bitstream file (.bit)
- Documentation (README, design description, etc.)

Example Naming

If your name is John Doe and you are submitting Task 2, the zip file should be named: **DLD-Lab5-FPGA-Prototyping-JohnDoe-Task2**

By adhering to this rubric and submission format, you will be evaluated on your design efficiency, correctness, and the quality of your documentation and submission.