ELEC 4200 Lab#0 Tutorial

Objectives(1)

- In this Lab exercise, we will design and implement a 2-to-1 multiplexer (MUX), using Xilinx *Vivado* tools to create a verilog model of the design, verify the model, and implement the model in a Field Programmable Gate Array (FPGA)
- The behavior of the multiplexer will be implemented in two different ways:
  - 1) Logic equations, which should be familiar from Digital Logic Circuits.
  - 2) Behavioral description, in which we specify the desired input/output behavior and allow the *Vivado* synthesis tool to implement the required logic.

Objectives(2)

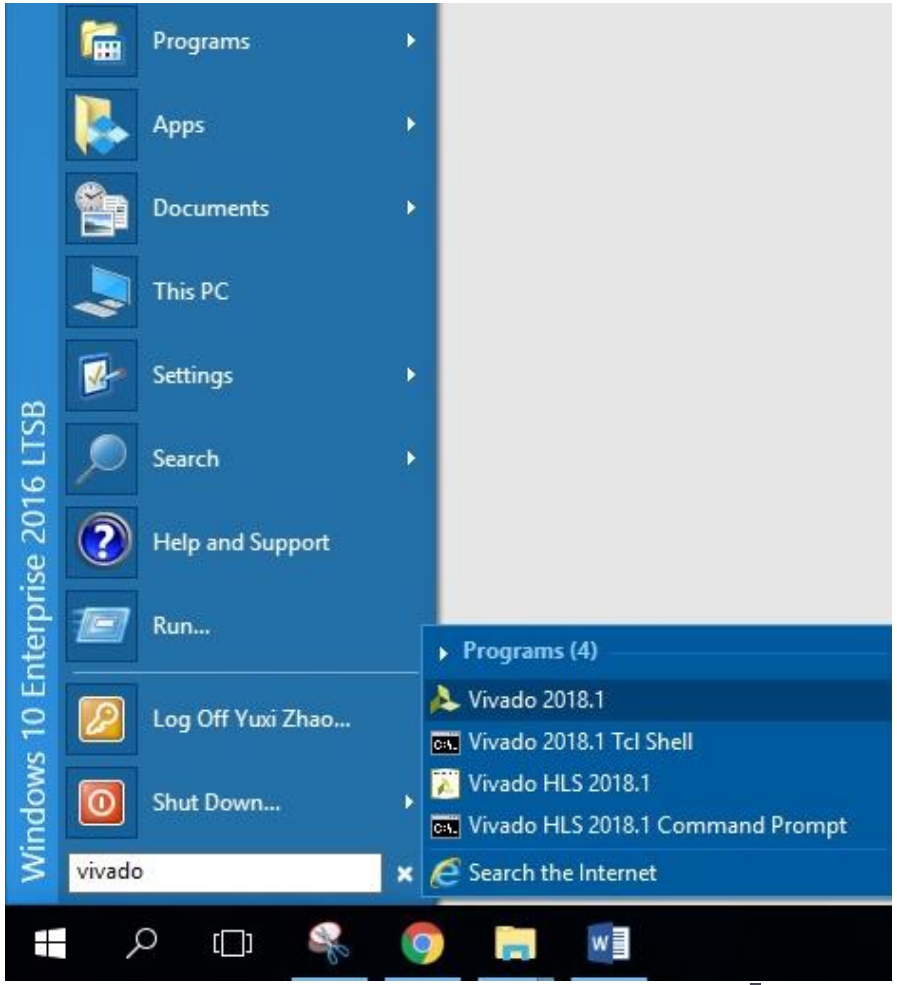
- Vivado simulator will be used to simulate and verify the design, with model debugging and correction performed as needed.
- After the design is fully verified, we will use *Vivado* to synthesize the design into an Artix-7 FPGA, generating a configuration data file which will then be downloaded onto a Diligent *Nexus 4 DDR* circuit board, where the implemented design will be verified on the hardware using switches to stimulate the circuit and LEDs to observe the outputs.
- If you have no idea what any of this means, don’t worry. That is why this is a tutorial!

Warning:

- You will need to actually read the instructions.
- If you attempt to just look at the pictures, you will miss steps, make mistakes, encounter errors, etc.
- When you ask for help with problems resulting from not reading the directions, your GTA will make fun of you before telling you to read the directions.
- You have been warned.

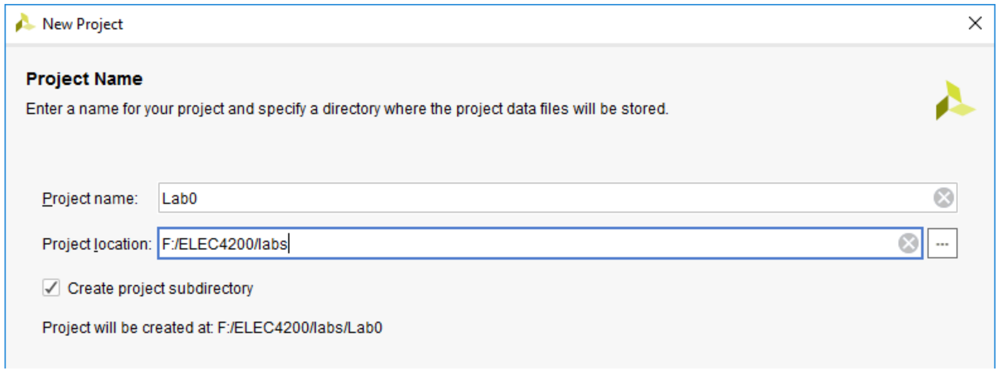
New Project (1)

- Click the start menu, and type "Vivado." Then look for "Vivado 2018.2" (or the latest installed version) and click on it.



New Project (2)

- Launch the *New Project Wizard* by clicking “Create New Project” on the *Quick Start* page, or “New Project” in the “File” dropdown menu.
- Click “Next” on the first screen to produce the *New Project* window.



New Project

Project Name

Enter a name for your project and specify a directory where the project data files will be stored.

Project name: Lab0

Project location: F:/ELEC4200/labs

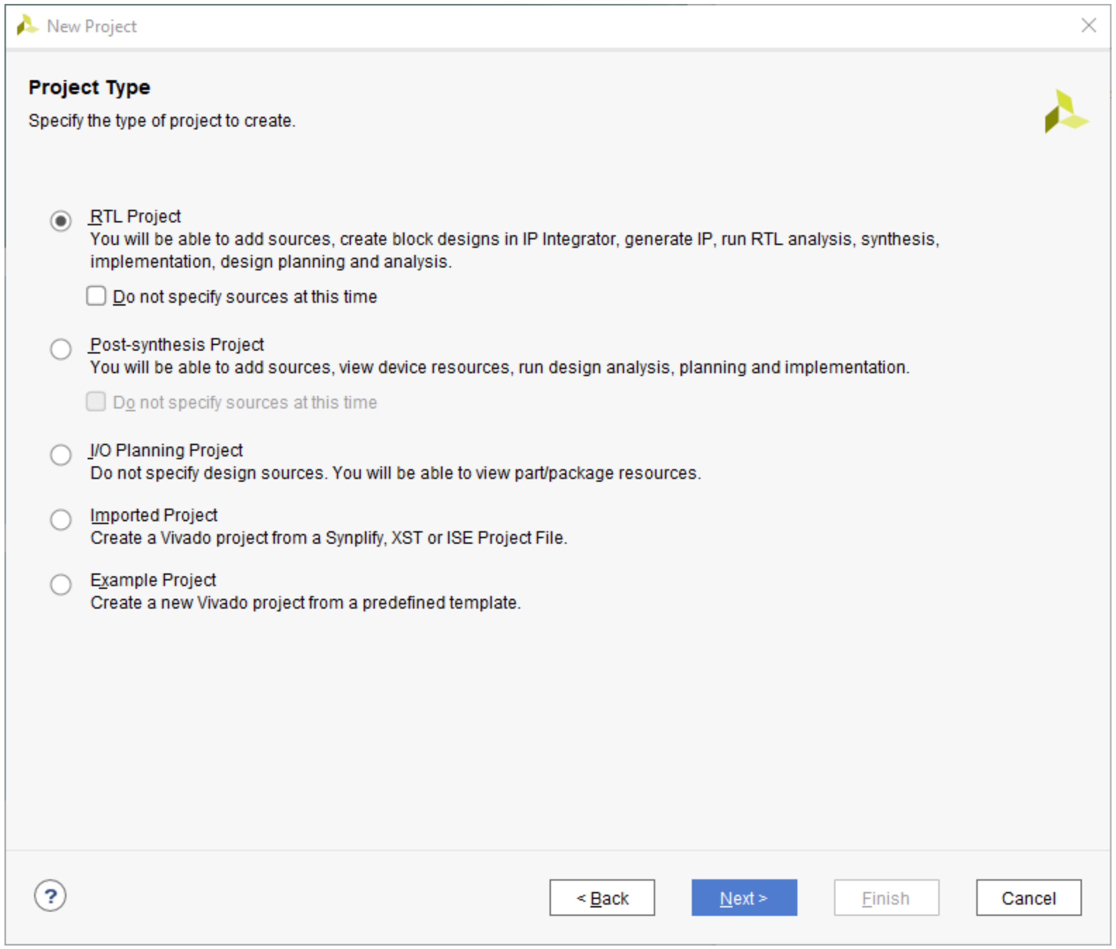
✓ Create project subdirectory

Project will be created at: F:/ELEC4200/labs/Lab0

- Enter the project name and location in the corresponding boxes.
- Note: It is HIGHLY recommended that while working in the lab your projects be located on an external flash drive or the C: drive (NOT your H: drive). If your project is located on the H: drive one of the later synthesis steps will ALWAYS fail! If using the C: drive you can use the “C:\TEMP\” directory.
- Give the project a descriptive name, like “LAB0” and click “Next”.

New Project (3)

- Select “RTL Project” as the project type and click “Next”.



New Project

Project Type

Specify the type of project to create.

[✓ ●] RTL Project
You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis.
☐ Do not specify sources at this time

○ Post-synthesis Project
You will be able to add sources, view device resources, run design analysis, planning and implementation.
☐ Do not specify sources at this time

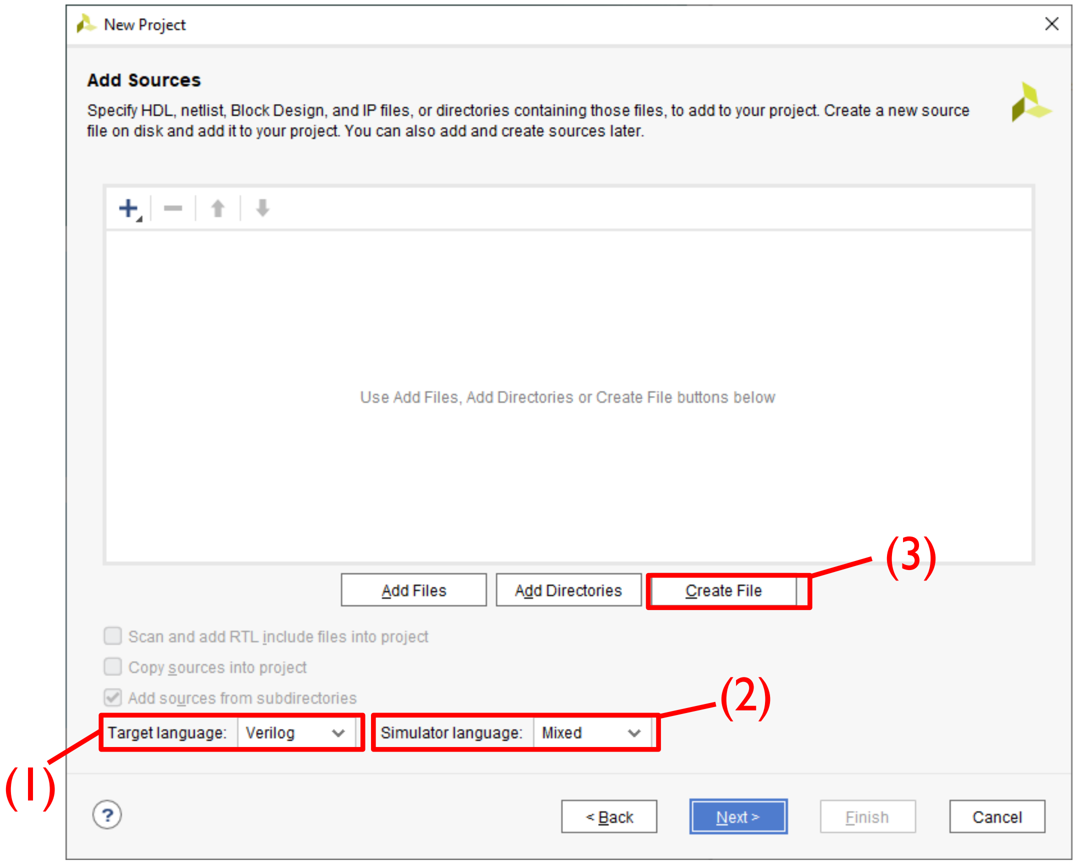
○ I/O Planning Project
Do not specify design sources. You will be able to view part/package resources.

○ Imported Project
Create a Vivado project from a Synplify, XST or ISE Project File.

○ Example Project
Create a new Vivado project from a predefined template.

New Project (4)

- In the *Add Sources* window, set the “Target Language” to verilog and the “Simulator language” to Mixed.
- After setting the language options, click “Create File”



New Project

Add Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those files, to add to your project. Create a new source file on disk and add it to your project. You can also add and create sources later.

Use Add Files, Add Directories or Create File buttons below

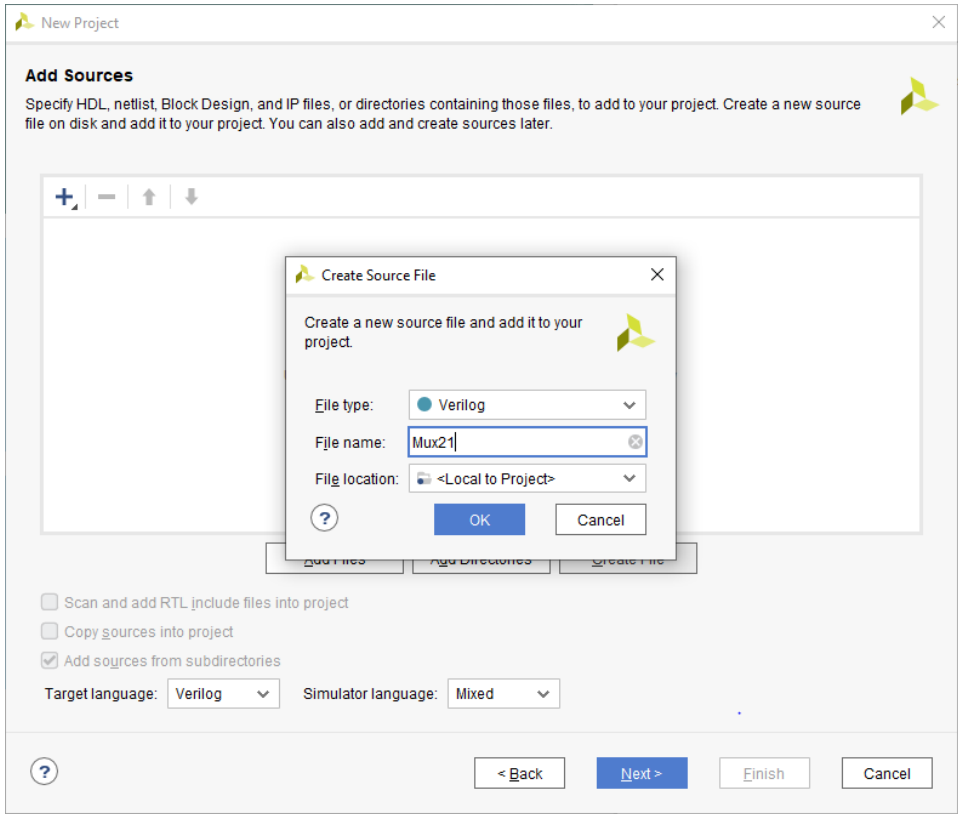
- [ ] Scan and add RTL include files into project
- [ ] Copy sources into project
- [x] Add sources from subdirectories

Target language: Verilog

Simulator language: Mixed

New Project (5)

- In the Create Source File dialog box, make sure the “File type” is set to Verilog and enter the name of your file.
  - Again, it is always a good idea to use a descriptive name, in this case *Mux21* because this is a 2-1 multiplexor.
- Click “OK” to close the dialog box and then “Next”.



New Project

Add Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those files, to add to your project. Create a new source file on disk and add it to your project. You can also add and create sources later.

Create Source File

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

File type: Verilog

File name: Mux21

File location: <Local to Project>

OK | Cancel

Scan and add RTL include files into project

Copy sources into project

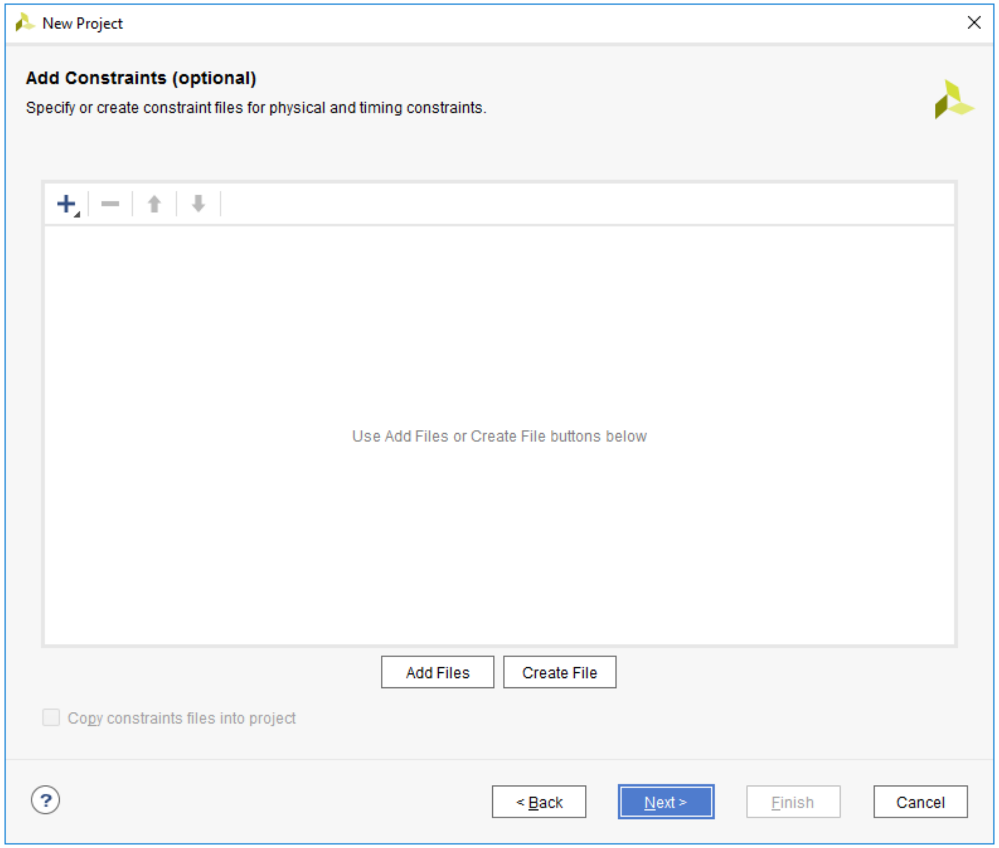
Add sources from subdirectories ✓

Target language: Verilog   Simulator language: Mixed

< Back | Next > | Finish | Cancel

New Project (6)

- We do not have any constraints yet, but in future projects, you may import and edit existing constraints files to save time. So click “Next” for now.



New Project

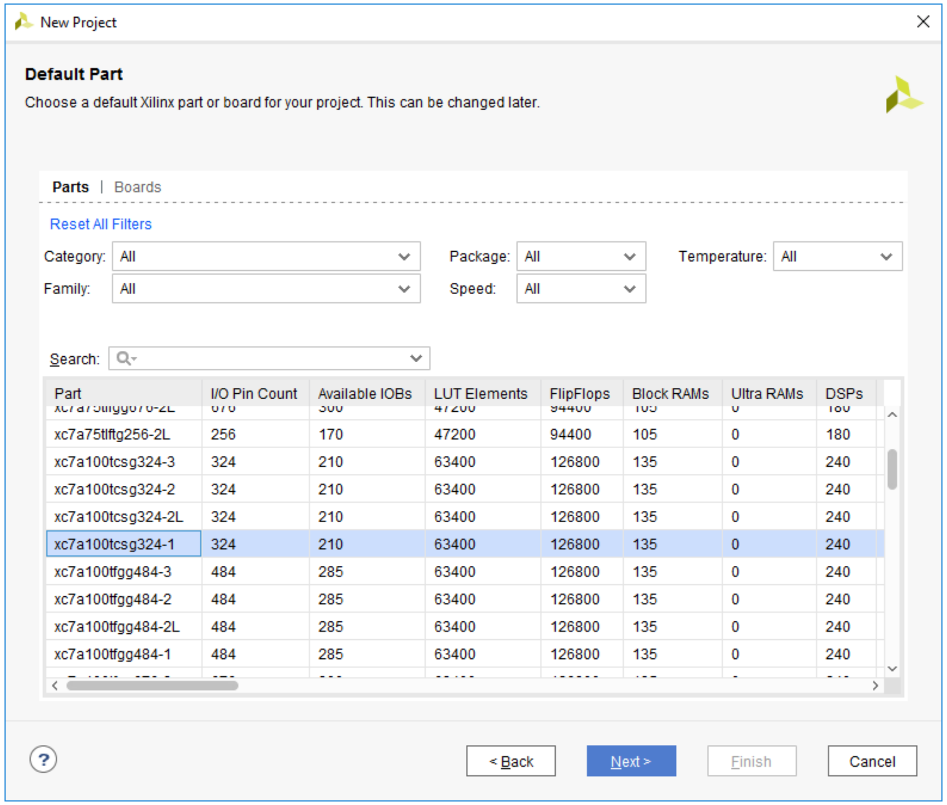
Add Constraints (optional)

Specify or create constraint files for physical and timing constraints.

Use Add Files or Create File buttons below

New Project (7)

- Here we need to select our device. You can use the search function, filters, or just scroll until you find our device: **XC7A100tcsg324-1**.
  - This FPGA is from the Xilinx Artix-7 family (**XC7A100T**).
  - The device is contained in a 324-pin **csg324** package.
  - The speed grade of the part is "**-1**".
- Click "Next"



Default Part

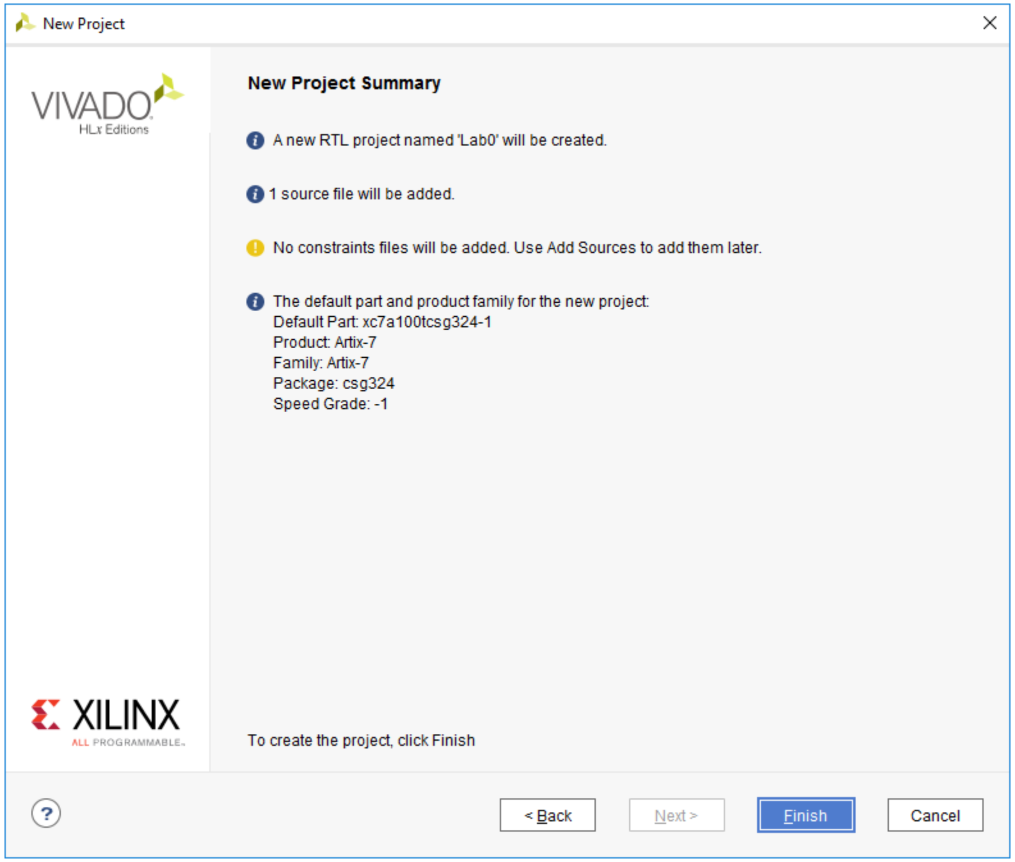
Choose a default Xilinx part or board for your project. This can be changed later.

Category: All
Package: All
Temperature: All
Family: All
Speed: All
Search:

Part	I/O Pin Count	Available IOBs	LUT Elements	FlipFlops	Block RAMs	Ultra RAMs	DSPs
xc7a75tfgg676-2L	676	300	47200	94400	105	0	180
xc7a75tlftg256-2L	256	170	47200	94400	105	0	180
xc7a100tcsg324-3	324	210	63400	126800	135	0	240
xc7a100tcsg324-2	324	210	63400	126800	135	0	240
xc7a100tcsg324-2L	324	210	63400	126800	135	0	240
xc7a100tcsg324-1	324	210	63400	126800	135	0	240
xc7a100tfgg484-3	484	285	63400	126800	135	0	240
xc7a100tfgg484-2	484	285	63400	126800	135	0	240
xc7a100tfgg484-2L	484	285	63400	126800	135	0	240
xc7a100tfgg484-1	484	285	63400	126800	135	0	240

New Project (8)

- The New Project Summary window lists information selected in the previous screens.
  - If necessary, use the “Back” button to return to previous screens to make changes/corrections.
- Click “Finish” to open the Project Manager window.



New Project Summary

A new RTL project named 'Lab0' will be created.

1 source file will be added.

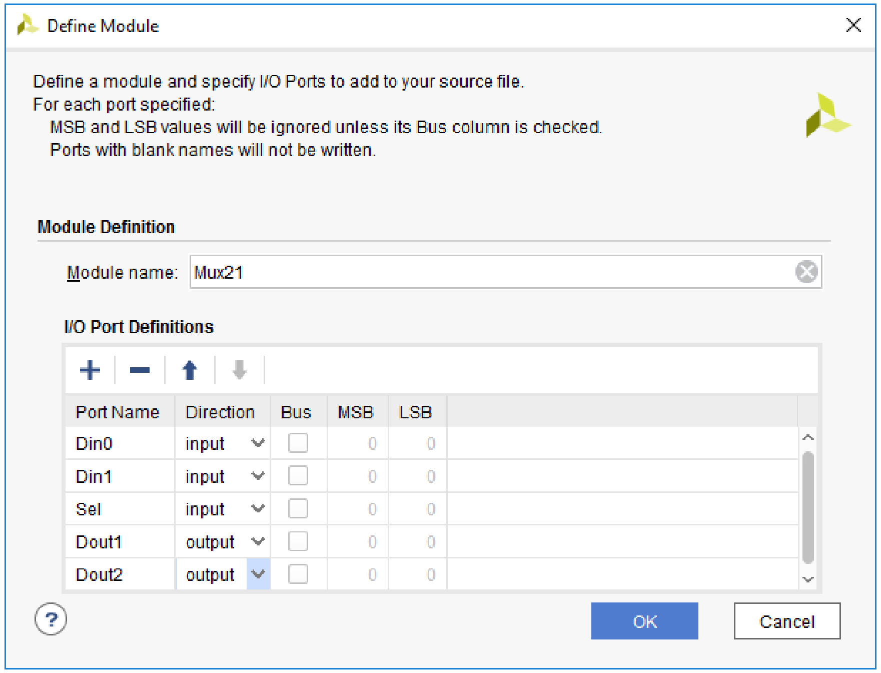
No constraints files will be added. Use Add Sources to add them later.

The default part and product family for the new project:
Default Part: xc7a100tcsg324-1
Product: Artix-7
Family: Artix-7
Package: csg324
Speed Grade: -1

To create the project, click Finish

Create the Verilog model

- Since we chose to create a new verilog file, *Vivado* will automatically launch a wizard to assist in creating the entity and architecture structures that comprise a verilog model. This can all be done by hand (and you can edit all of this later), but there is no reason not to take advantage of the wizard utility.
- Leave the Entity and Architecture names as their defaults.
- Create three inputs (Direction "input"): *Din0*, *Din1*, *Sel*
- Create two outputs (Direction "output"): *Dout1, Dout2*
- Click "OK"



Define Module

Define a module and specify I/O Ports to add to your source file.
For each port specified:
MSB and LSB values will be ignored unless its Bus column is checked.
Ports with blank names will not be written.

Module Definition

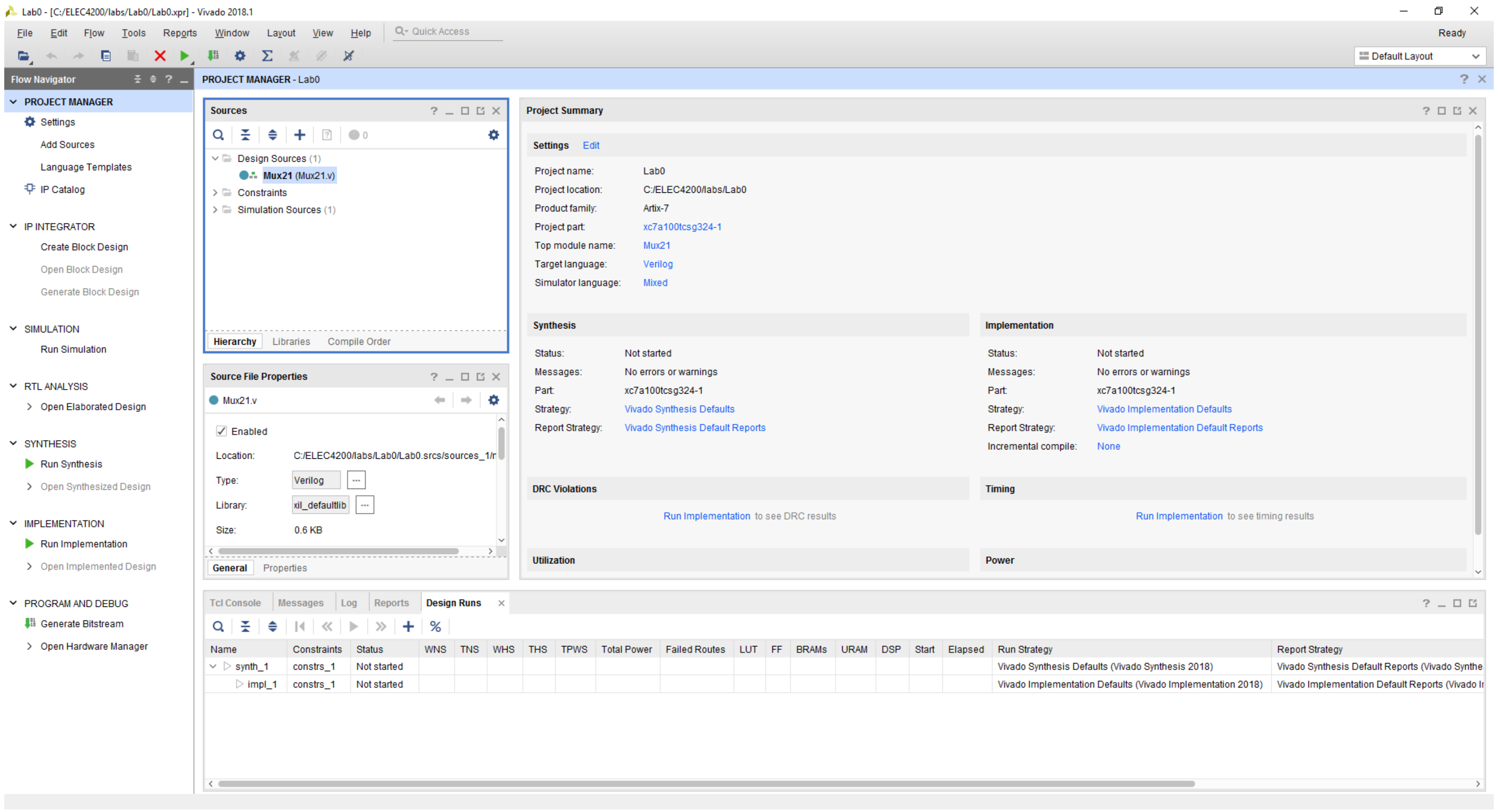
Module name: Mux21

I/O Port Definitions

Port Name	Direction	Bus	MSB	LSB
Din0	input	☐	0	0
Din1	input	☐	0	0
Sel	input	☐	0	0
Dout1	output	☐	0	0
Dout2	output	☐	0	0

Edit Verilog Model(1)

- Open your new verilog file from the Sources window (double click on the Design Source name, *Mux21*)



Source File Properties: Mux21.v

☑ Enabled
Location: C:/ELEC4200/labs/Lab0/Lab0.srcs/sources_1/n
Type: Verilog
Library: xil_defaultlib
Size: 0.6 KB

Project Summary

Settings Edit

Project name:	Lab0
Project location:	C:/ELEC4200/labs/Lab0
Product family:	Artix-7
Project part:	xc7a100tcsg324-1
Top module name:	Mux21
Target language:	Verilog
Simulator language:	Mixed

Synthesis

Status:	Not started
Messages:	No errors or warnings
Part:	xc7a100tcsg324-1
Strategy:	Vivado Synthesis Defaults
Report Strategy:	Vivado Synthesis Default Reports

Implementation

Status:	Not started
Messages:	No errors or warnings
Part:	xc7a100tcsg324-1
Strategy:	Vivado Implementation Defaults
Report Strategy:	Vivado Implementation Default Reports
Incremental compile:	None

DRC Violations

Run Implementation to see DRC results

Timing

Run Implementation to see timing results

Utilization

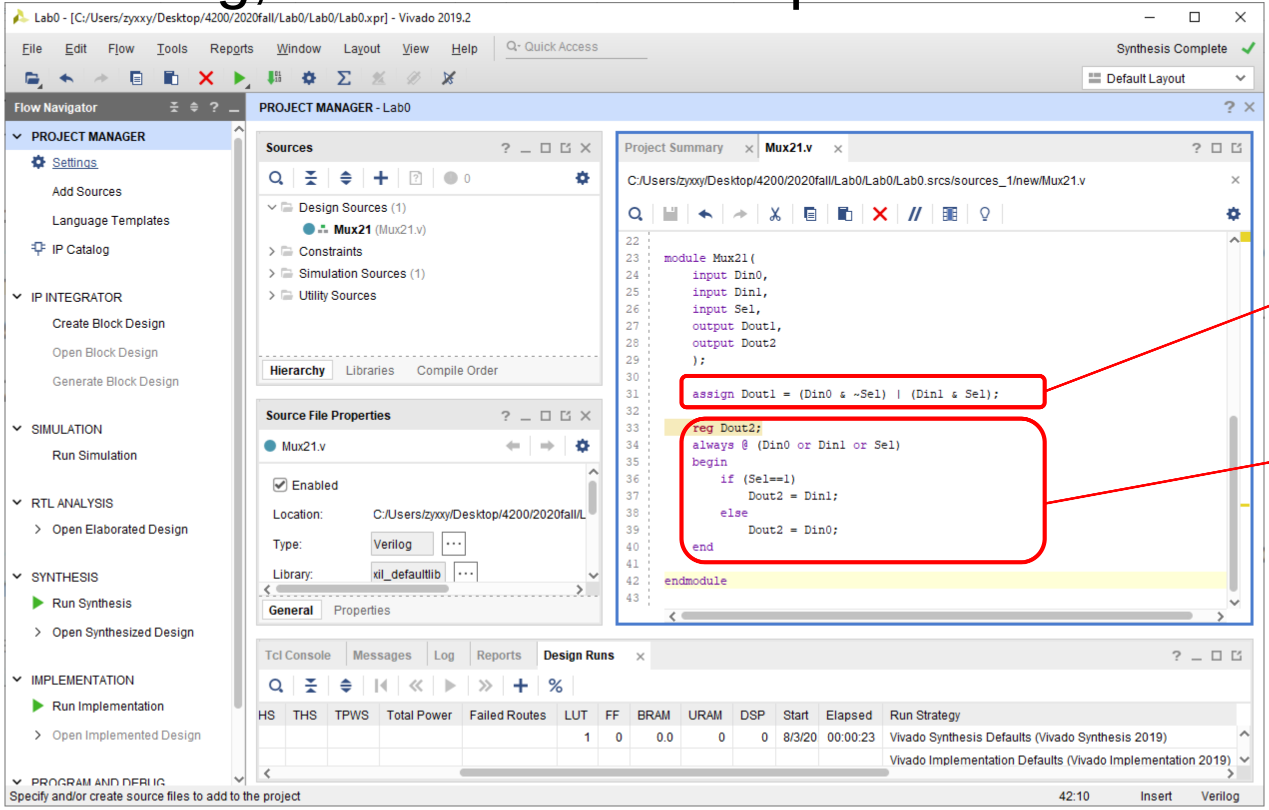
Power

Design Runs

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAMs	URAM	DSP	Start	Elapsed	Run Strategy	Report Strategy
synth_1	constrs_1	Not started															Vivado Synthesis Defaults (Vivado Synthesis 2018)	Vivado Synthesis Default Reports (Vivado Synthe
impl_1	constrs_1	Not started															Vivado Implementation Defaults (Vivado Implementation 2018)	Vivado Implementation Default Reports (Vivado In

Edit Verilog Model(2)

- Scroll down and add the code in the red rectangles between the Begin and End statements of the architecture section of the model.
- After saving, we can then set up the simulator.



Lab0 - [C:/Users/zyxxy/Desktop/4200/2020fall/Lab0/Lab0/Lab0.xpr] - Vivado 2019.2

C:/Users/zyxxy/Desktop/4200/2020fall/Lab0/Lab0/Lab0.srcs/sources_1/new/Mux21.v

```
module Mux21(
    input Din0,
    input Din1,
    input Sel,
    output Dout1,
    output Dout2
    );

    assign Dout1 = (Din0 & ~Sel) | (Din1 & Sel);

    reg Dout2;
    always @ (Din0 or Din1 or Sel)
    begin
        if (Sel==1)
            Dout2 = Din1;
        else
            Dout2 = Din0;
    end

endmodule
```

HS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP	Start	Elapsed	Run Strategy
					1	0	0.0	0	0	8/3/20	00:00:23	Vivado Synthesis Defaults (Vivado Synthesis 2019)
												Vivado Implementation Defaults (Vivado Implementation 2019)

Specify and/or create source files to add to the project

Mux models:

1st statement: logic equation

2nd statement: “behavior”

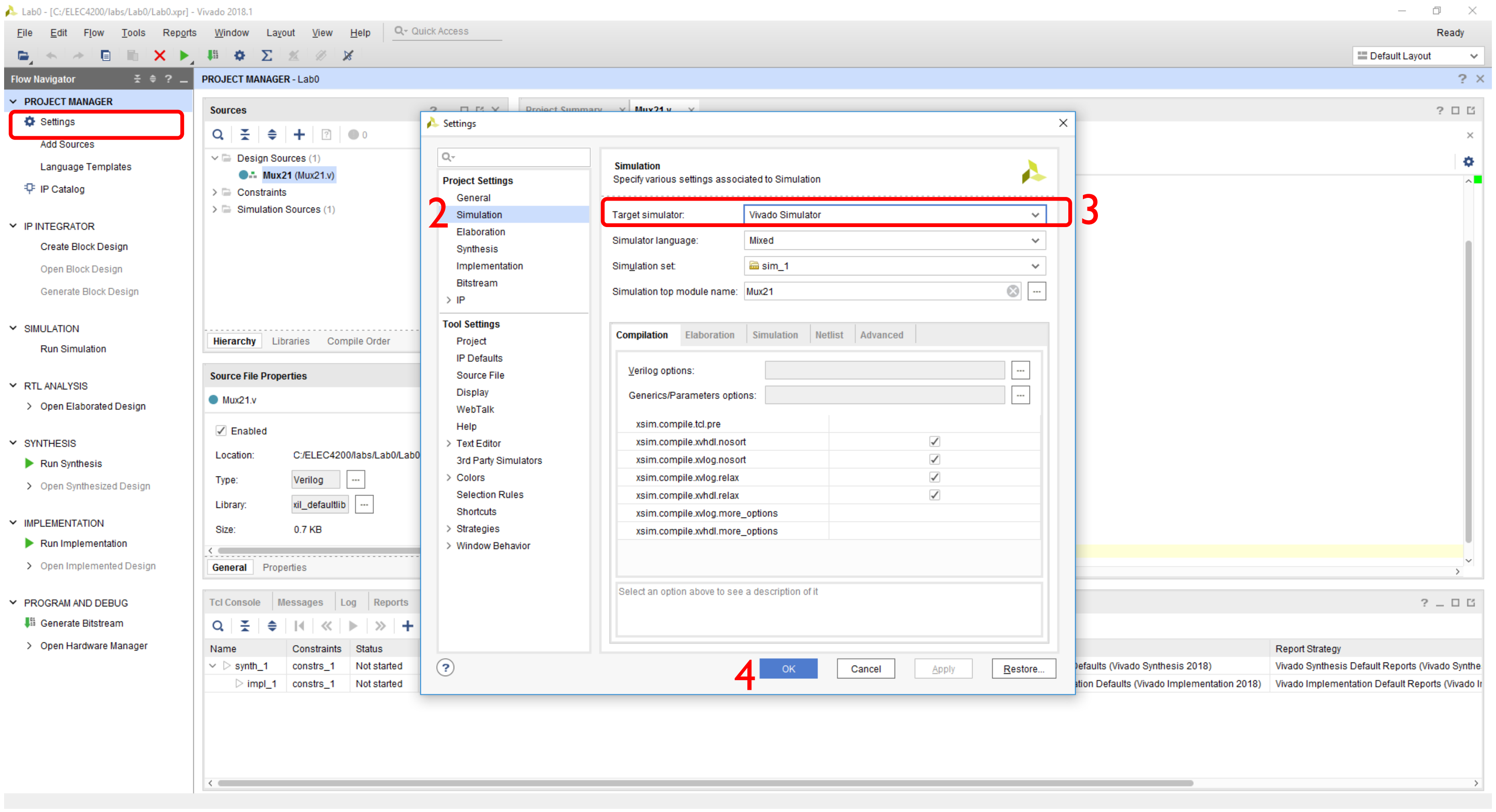
Simulator Setup(1)

- Xilinx *Vivado* integrates support for its own simulator, as well as several other commercial simulation tools. Therefore, you must select *Vivado simulator* from a list of simulators.

- As long as you use the same computer week to week, this setting should only have to be made this one time. However, if you ever get errors from *Vivado* about not being able to find the simulator, check this setting first.

Simulator Setup(2)

1. In the Project Manager pane, select *Project Settings*
2. In the Project Settings window, select *Simulation*
3. Select *Vivado Simulator* in the Target simulator drop-down menu.
4. Click OK to close the window.



Simulation
Specify various settings associated to Simulation

Target simulator: Vivado Simulator
Simulator language: Mixed
Simulation set: sim_1
Simulation top module name: Mux21

Compilation | Elaboration | Simulation | Netlist | Advanced

Verilog options:
Generics/Parameters options:

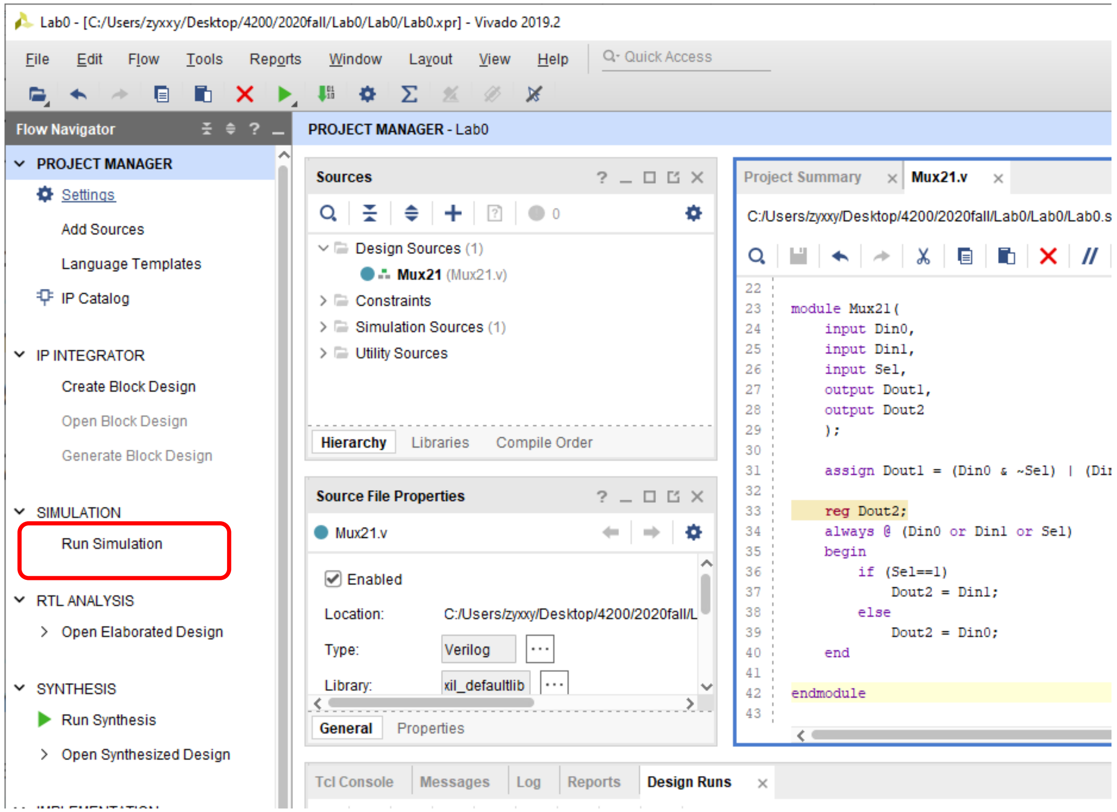
xsim.compile.tcl.pre	
xsim.compile.xvhdl.nosort	✓
xsim.compile.xvlog.nosort	✓
xsim.compile.xvlog.relax	✓
xsim.compile.xvhdl.relax	✓
xsim.compile.xvlog.more_options	
xsim.compile.xvhdl.more_options	

Select an option above to see a description of it

Name	Constraints	Status		Report Strategy
synth_1	constrs_1	Not started	efaults (Vivado Synthesis 2018)	Vivado Synthesis Default Reports (Vivado Synthe
impl_1	constrs_1	Not started	tion Defaults (Vivado Implementation 2018)	Vivado Implementation Default Reports (Vivado I

Simulation(1)

- This tutorial describes only how to perform a basic simulation, but its is recommended that you experiment with its many capabilities, as you may find ways it can help you debug in later lab sessions.
- Click on “Run Simulation" in the Flow Navigator, and then Run “Behavioral Simulation” in the popup menu.



Location: C:/Users/zyxxy/Desktop/4200/2020fall/L

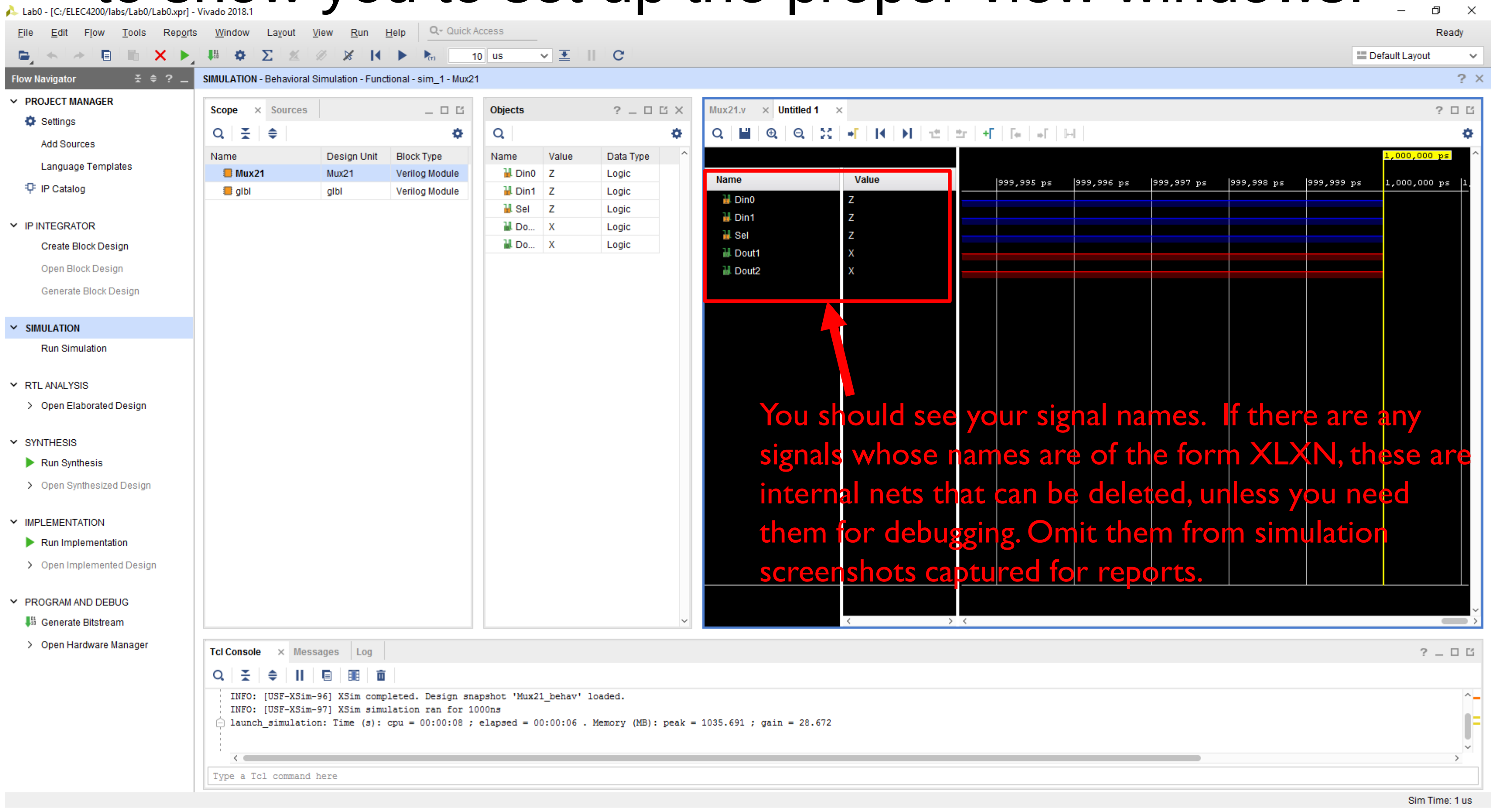
Type: Verilog

Library: xil_defaultlib

```
module Mux21(
    input Din0,
    input Din1,
    input Sel,
    output Dout1,
    output Dout2
    );

    assign Dout1 = (Din0 & ~Sel) | (Di

    reg Dout2;
    always @ (Din0 or Din1 or Sel)
    begin
        if (Sel==1)
            Dout2 = Din1;
        else
            Dout2 = Din0;
    end

endmodule
```

Simulation(2)

- When the simulator is finished launching you should see the following window. If you do not, call the GTA to show you to set up the proper view windows.



Name	Design Unit	Block Type
Mux21	Mux21	Verilog Module
glbl	glbl	Verilog Module

Name	Value	Data Type
Din0	Z	Logic
Din1	Z	Logic
Sel	Z	Logic
Do...	X	Logic
Do...	X	Logic

Name	Value
Din0	Z
Din1	Z
Sel	Z
Dout1	X
Dout2	X

You should see your signal names. If there are any signals whose names are of the form XLXN, these are internal nets that can be deleted, unless you need them for debugging. Omit them from simulation screenshots captured for reports.

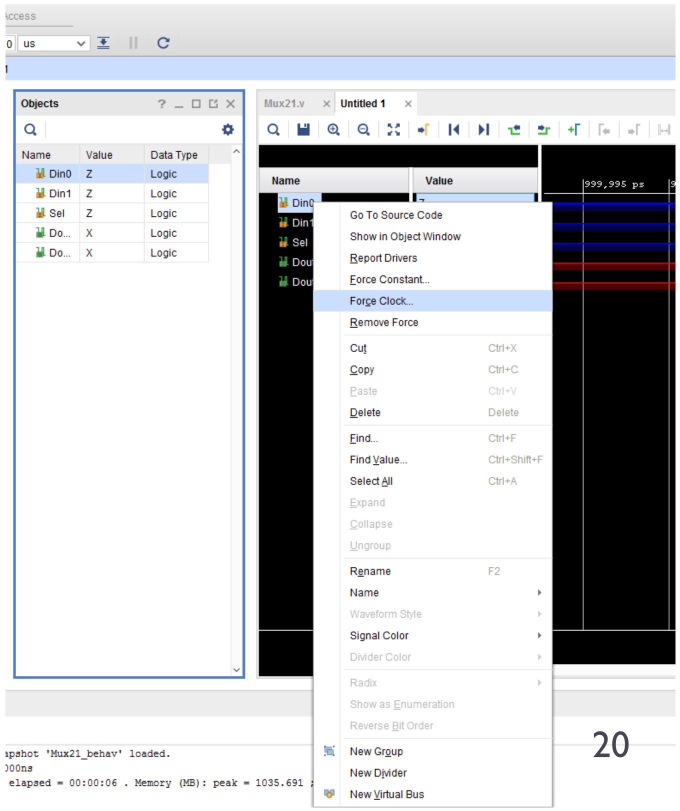
```
INFO: [USF-XSim-96] XSim completed. Design snapshot 'Mux21_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:08 ; elapsed = 00:00:06 . Memory (MB): peak = 1035.691 ; gain = 28.672
```

Type a Tcl command here

Simulation(3)

- Design verification requires that you stimulate the inputs and observe the outputs. You are to stimulate the inputs with all possible input combinations and observe each output to verify its correctness.
- This is called "exhaustive testing". It is suitable for a simple circuit (such as the mux) but is not practical for large circuits with many inputs.

- Right click on one of your input signals (ex: Din0) and select "Force clock" from the context menu.

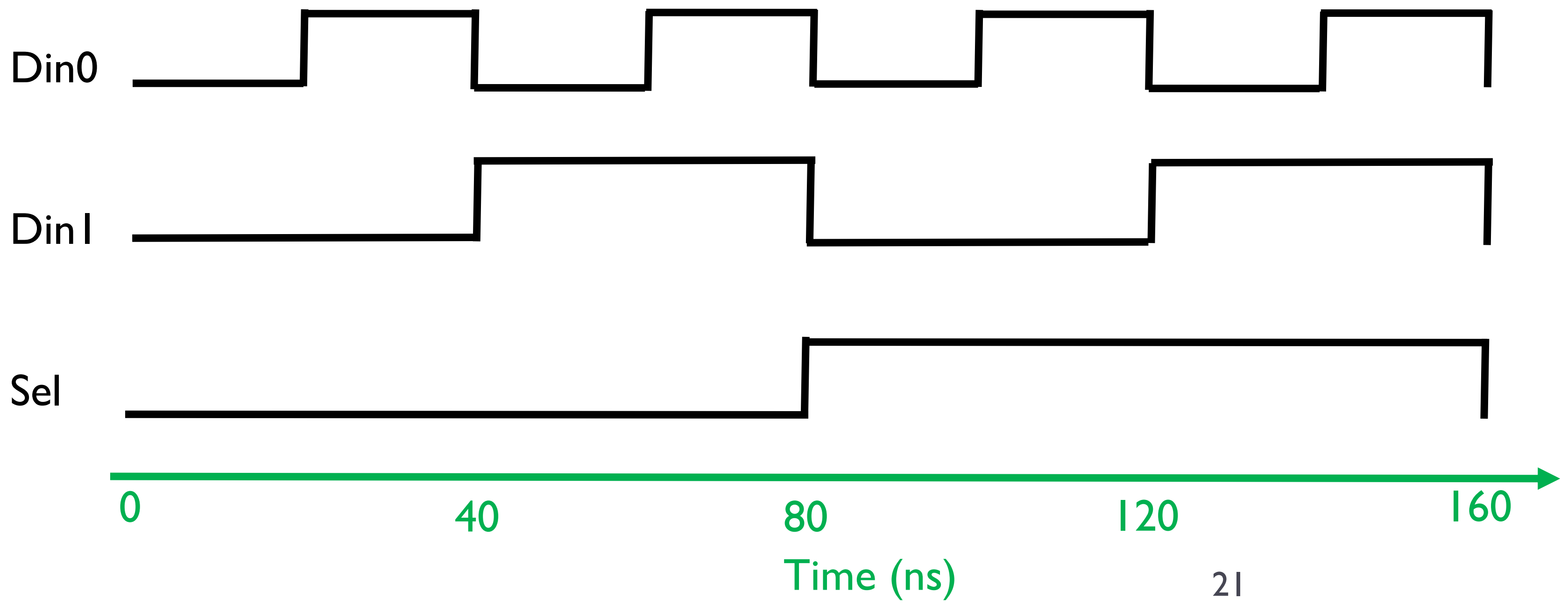


Name	Value	Data Type
Din0	Z	Logic
Din1	Z	Logic
Sel	Z	Logic
Do...	X	Logic
Do...	X	Logic

```
apshot 'Mux21_behav' loaded.
)00ns
 elapsed = 00:00:06 . Memory (MB): peak = 1035.691 ;
```

Simulation(4)

- For the 2-1 MUX we have 3 inputs (Din0, Din1, and Sel) where each input can be either logic high (1) or logic low (0).
- There are 2^3=8 possible input combinations. While we could stimulate each combination individually, an easier approach would be to use *the simulator's* clock stimulator to generate the input timing diagrams below.

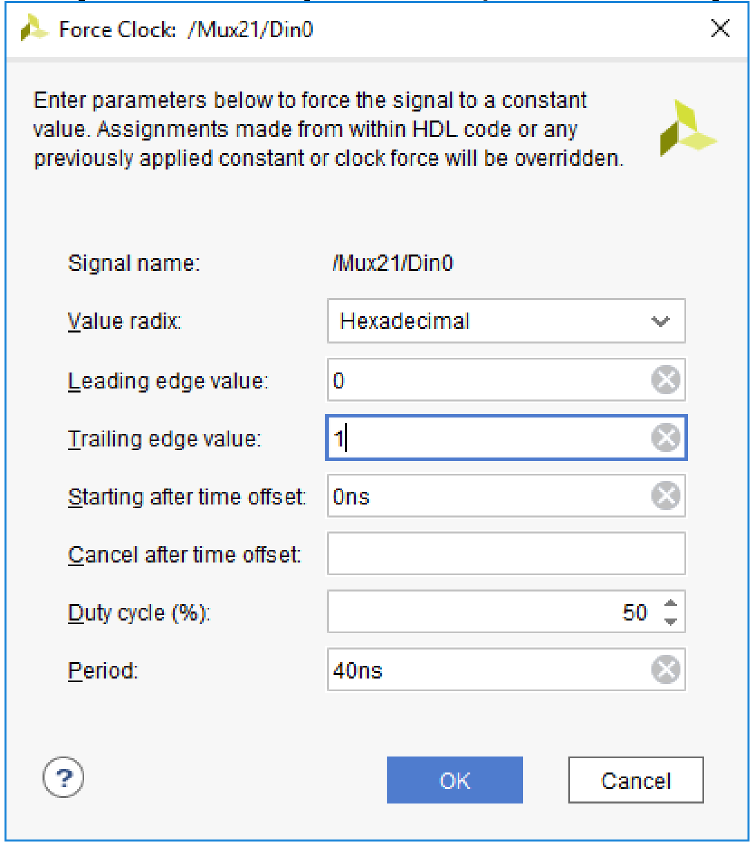


Simulation(5)

- The right side of the Stimulators dialog box should now show the clock parameters.
- You can set the starting value (0 or 1), adjust the offset value to start the clock, adjust the period, and adjust the duty cycle

- Set the options as shown on the right to create a waveform that starts low and repeats every 40ns with a 50% duty cycle.

- Click "Apply"


Force Clock: /Mux21/Din0

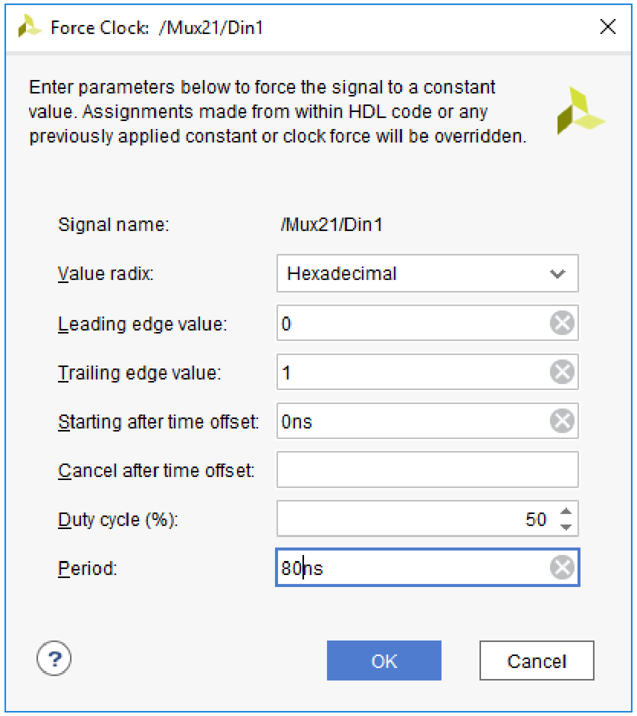
Enter parameters below to force the signal to a constant value. Assignments made from within HDL code or any previously applied constant or clock force will be overridden.

Signal name:	/Mux21/Din0
Value radix:	Hexadecimal
Leading edge value:	0
Trailing edge value:	1
Starting after time offset:	0ns
Cancel after time offset:	
Duty cycle (%):	50
Period:	40ns

Simulation(6)

- Repeat the clock procedure for Din1 and Sel with the following periods. Click "Apply" after each signal is added.
  - Din1: 80ns period
  - Sel: 160ns period

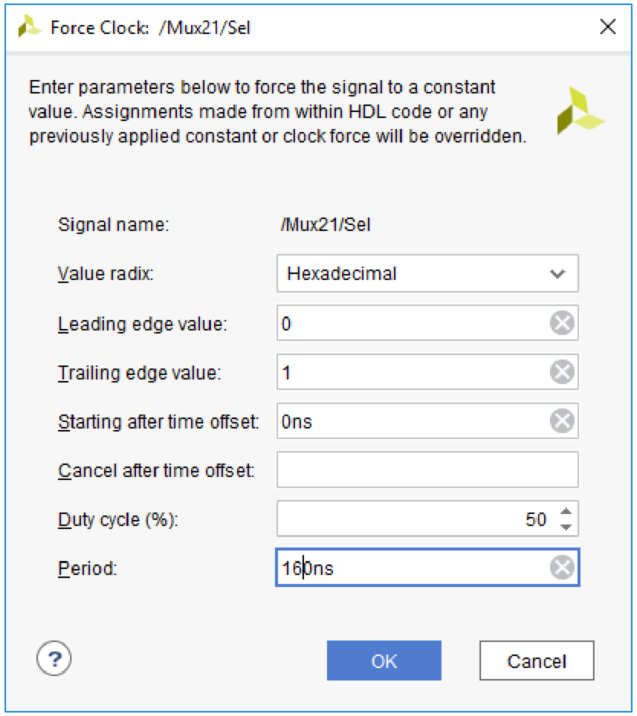
- Click "Close" when you are finished.



Force Clock: /Mux21/Din1

Enter parameters below to force the signal to a constant value. Assignments made from within HDL code or any previously applied constant or clock force will be overridden.

Signal name:	/Mux21/Din1
Value radix:	Hexadecimal
Leading edge value:	0
Trailing edge value:	1
Starting after time offset:	0ns
Cancel after time offset:	
Duty cycle (%):	50
Period:	80ns



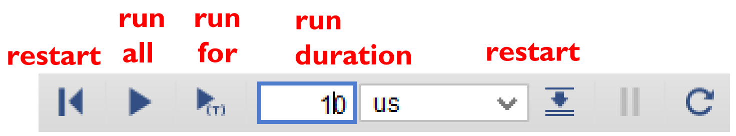
Force Clock: /Mux21/Sel

Enter parameters below to force the signal to a constant value. Assignments made from within HDL code or any previously applied constant or clock force will be overridden.

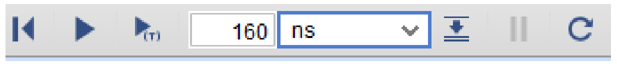
Signal name:	/Mux21/Sel
Value radix:	Hexadecimal
Leading edge value:	0
Trailing edge value:	1
Starting after time offset:	0ns
Cancel after time offset:	
Duty cycle (%):	50
Period:	160ns

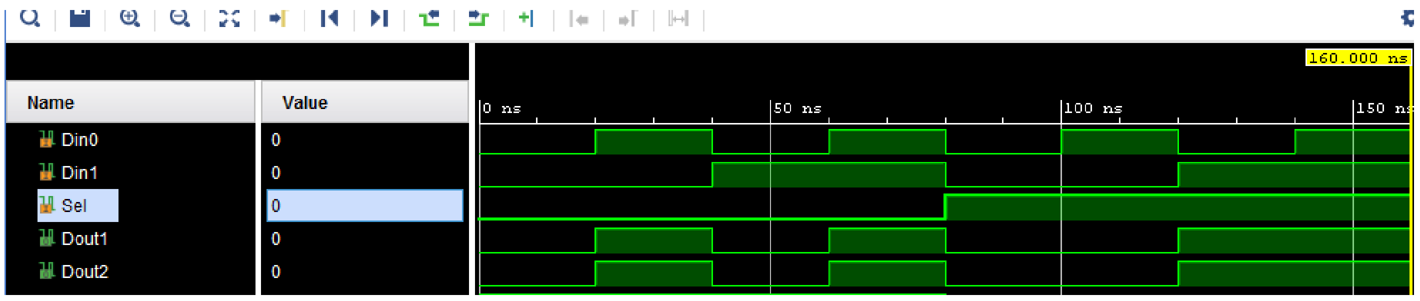
Simulation(7)

- Now we need to run the simulation using the following simulation controls.



- Whenever you begin a new simulation you should clear the waveforms window. Do this by clicking the "Restart Simulation" button or by typing "restart" into the Console Window.
- Run the simulation for 160ns by typing 160 ns into the "Run Duration" box and then click "Run For". Your output should look similar to this. Use the zoom controls as needed to make the waveform fill the window.





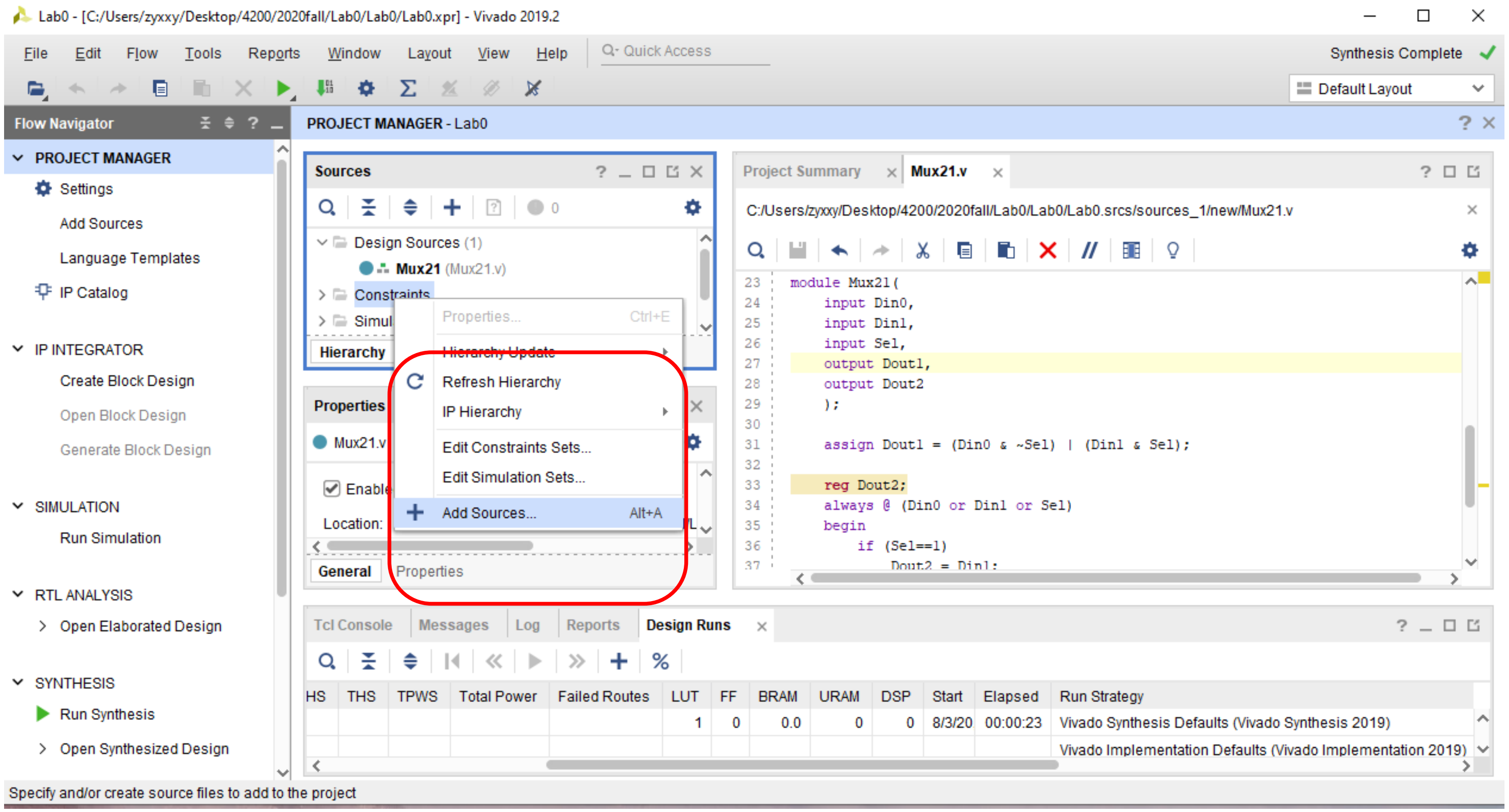
Name	Value
Din0	0
Din1	0
Sel	0
Dout1	0
Dout2	0

Simulation(8)

- Examine your results and ensure that they accurately match the operation of a 2-1 MUX for both Dout1 and Dout2.
- If you observe any incorrect results during simulation, go back and debug your circuit. Do not proceed any further until your simulation produces the correct results.
- If time allows you may want to experiment with some of the other “types” in the Stimulators window so that you can become familiar with their operation and know how to use them for future labs.
- If your final circuit does not work and it is traced it back to an error in simulation, the GTA reserves the right to laugh at you publicly in front of your classmates.

Add Constraints (1)

- We need to define a “constraints file” that defines which signals (Din, Dout, Sel) go to which pins on the FPGA.
- Right click on the “Constraints” folder in the project manager and select “Add Sources”.



C:/Users/zyxxy/Desktop/4200/2020fall/Lab0/Lab0/Lab0.srcs/sources_1/new/Mux21.v

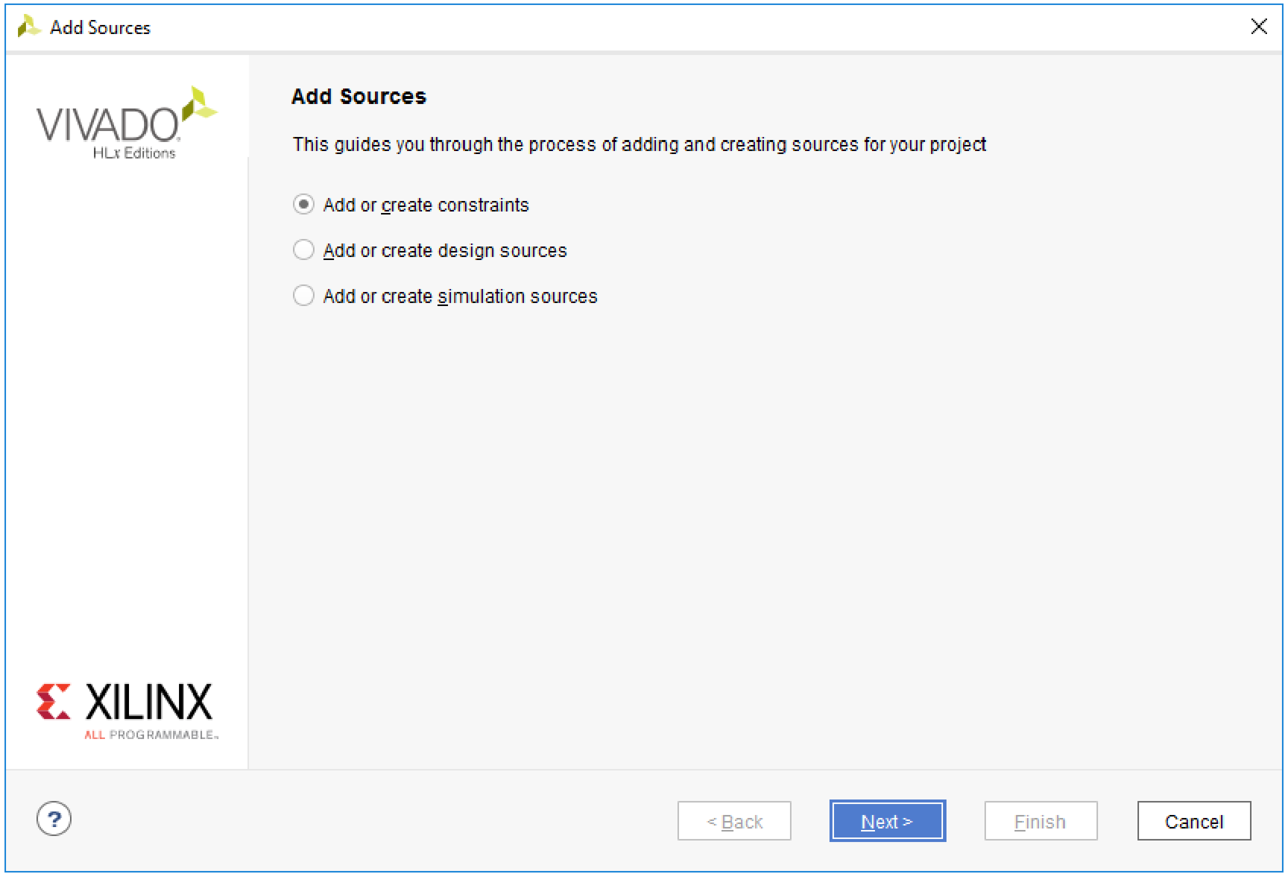
```
module Mux21(
    input Din0,
    input Din1,
    input Sel,
    output Dout1,
    output Dout2
    );

    assign Dout1 = (Din0 & ~Sel) | (Din1 & Sel);

    reg Dout2;
    always @ (Din0 or Din1 or Sel)
    begin
        if (Sel==1)
            Dout2 = Din1;
```

HS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP	Start	Elapsed	Run Strategy
					1	0	0.0	0	0	8/3/20	00:00:23	Vivado Synthesis Defaults (Vivado Synthesis 2019)
												Vivado Implementation Defaults (Vivado Implementation 2019)

Specify and/or create source files to add to the project

Add Constraints (2)

- This opens the "Add Sources Wizard". Make sure "Add or create constraints" is selected and click "Next".



Add Sources

This guides you through the process of adding and creating sources for your project

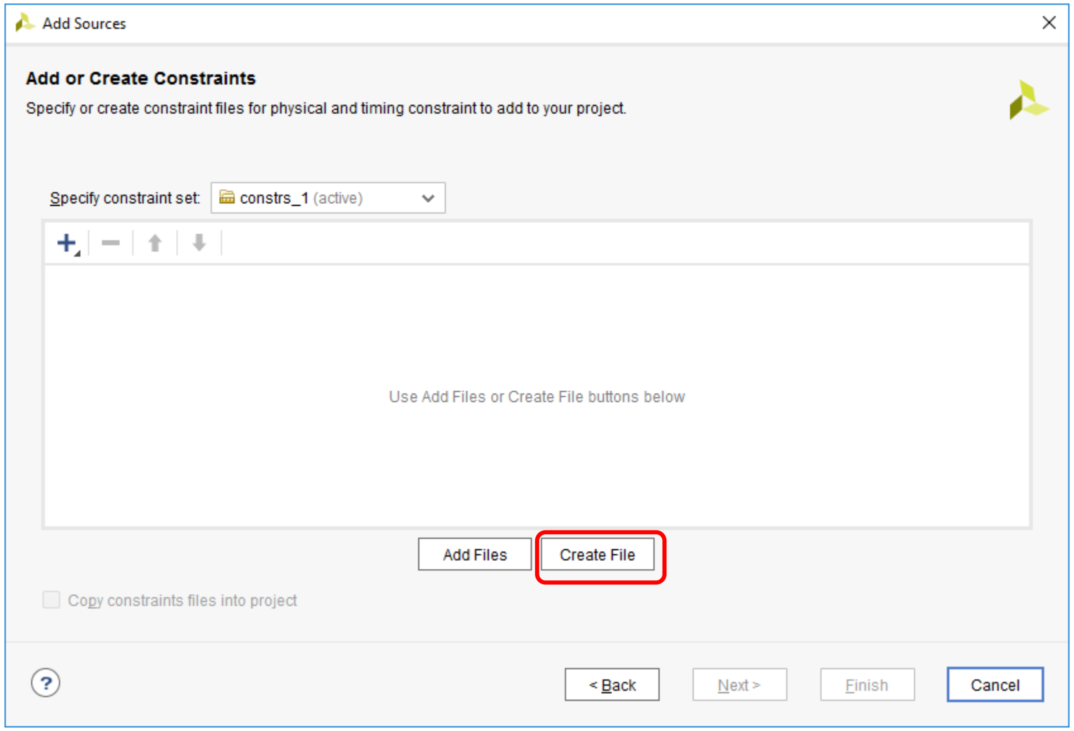
[✓ Add or create constraints]

Add or create design sources

Add or create simulation sources

Add Constraints (3)

- Click "Create File" and in the dialog box that opens, name your constraints file. As always, use a descriptive name, like "LAB0_Constraints". Then click "OK".



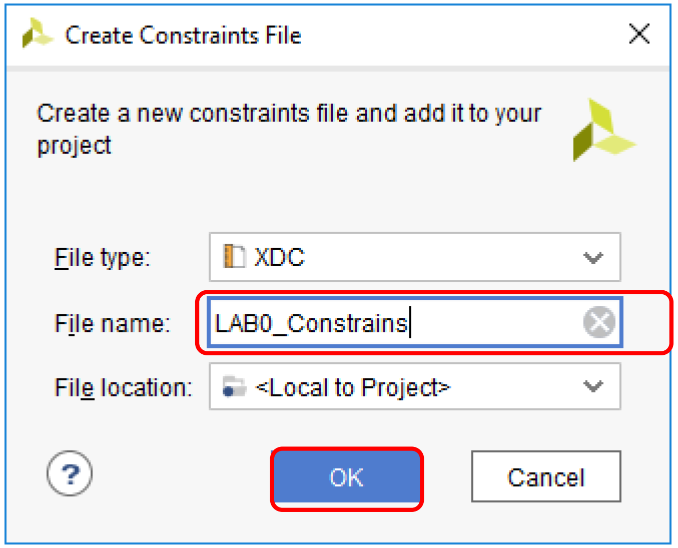
Add Sources

Add or Create Constraints

Specify or create constraint files for physical and timing constraint to add to your project.

Specify constraint set: constrs_1 (active)

Use Add Files or Create File buttons below



Create Constraints File

Create a new constraints file and add it to your project

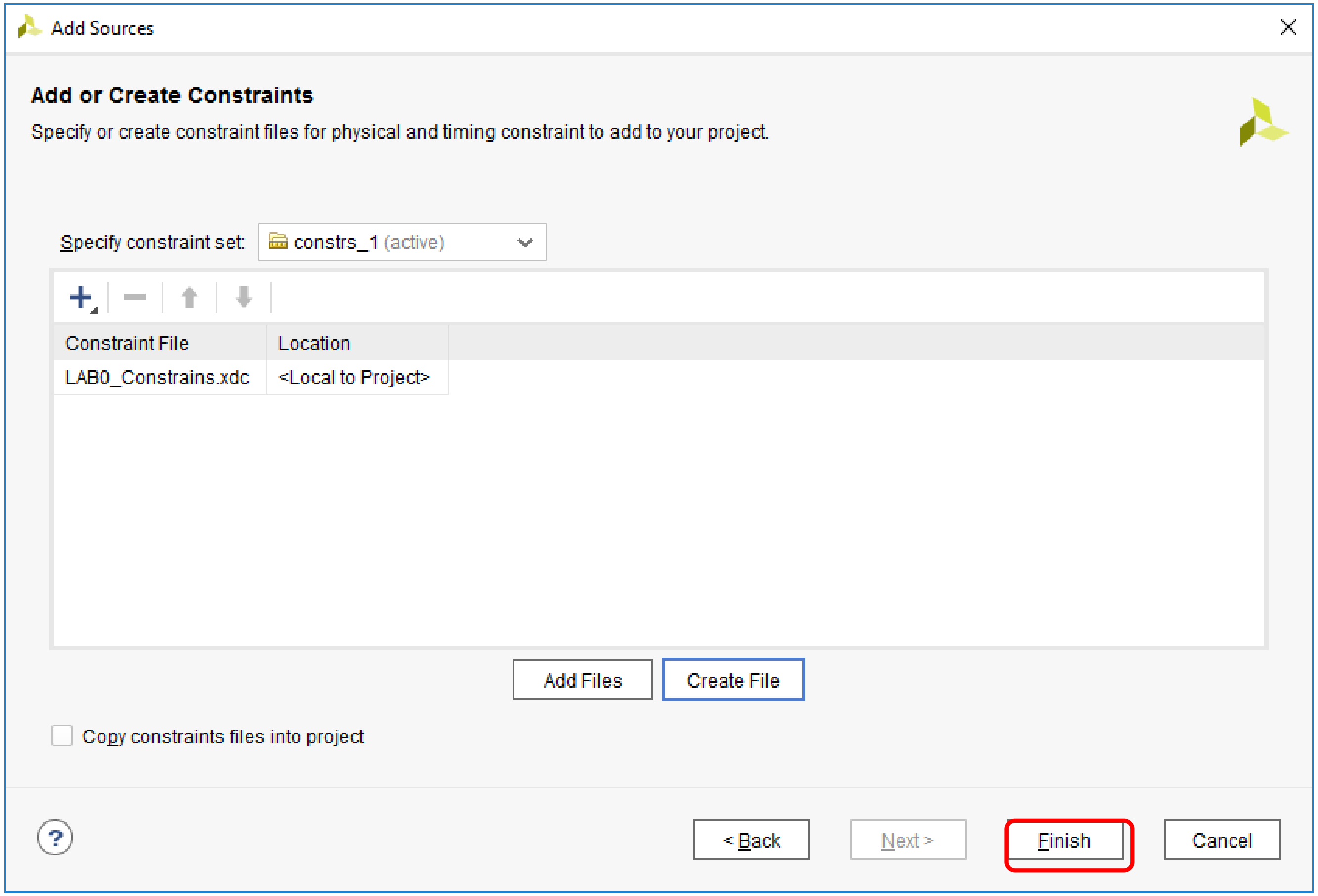
File type: XDC

File name: LAB0_Constrains

File location: <Local to Project>

Add Constraints (4)

- Click "Finish" to add your file to the project.



Add Sources

Add or Create Constraints

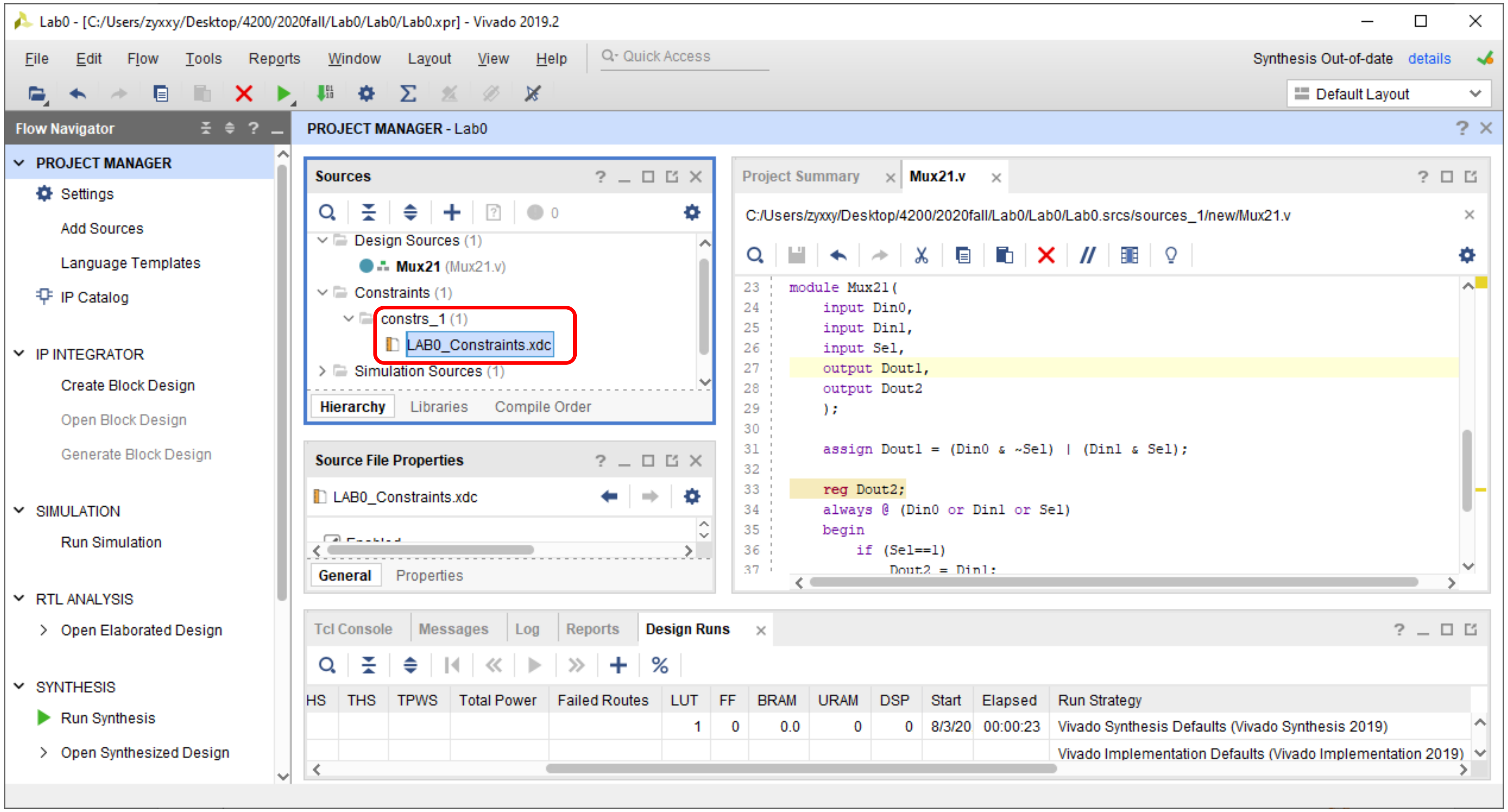
Specify or create constraint files for physical and timing constraint to add to your project.

Specify constraint set: constrs_1 (active)

Constraint File	Location
LAB0_Constrains.xdc	<Local to Project>

Add Constraints (5)

- Open your newly added file by double clicking on it from the Project Manager.



Lab0 - [C:/Users/zyxxy/Desktop/4200/2020fall/Lab0/Lab0/Lab0.xpr] - Vivado 2019.2

```
Design Sources (1)
    Mux21 (Mux21.v)
Constraints (1)
    constrs_1 (1)
        LAB0_Constraints.xdc
Simulation Sources (1)
```

Source File Properties: LAB0_Constraints.xdc

C:/Users/zyxxy/Desktop/4200/2020fall/Lab0/Lab0.srcs/sources_1/new/Mux21.v

```
module Mux21(
    input Din0,
    input Din1,
    input Sel,
    output Dout1,
    output Dout2
    );

    assign Dout1 = (Din0 & ~Sel) | (Din1 & Sel);

    reg Dout2;
    always @ (Din0 or Din1 or Sel)
    begin
        if (Sel==1)
            Dout2 = Din1;
```

HS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP	Start	Elapsed	Run Strategy
					1	0	0.0	0	0	8/3/20	00:00:23	Vivado Synthesis Defaults (Vivado Synthesis 2019)
												Vivado Implementation Defaults (Vivado Implementation 2019)

Add Constraints (6)

- Add the following lines in the constraints file, renaming the ports to match those in your design. Save after you are done.

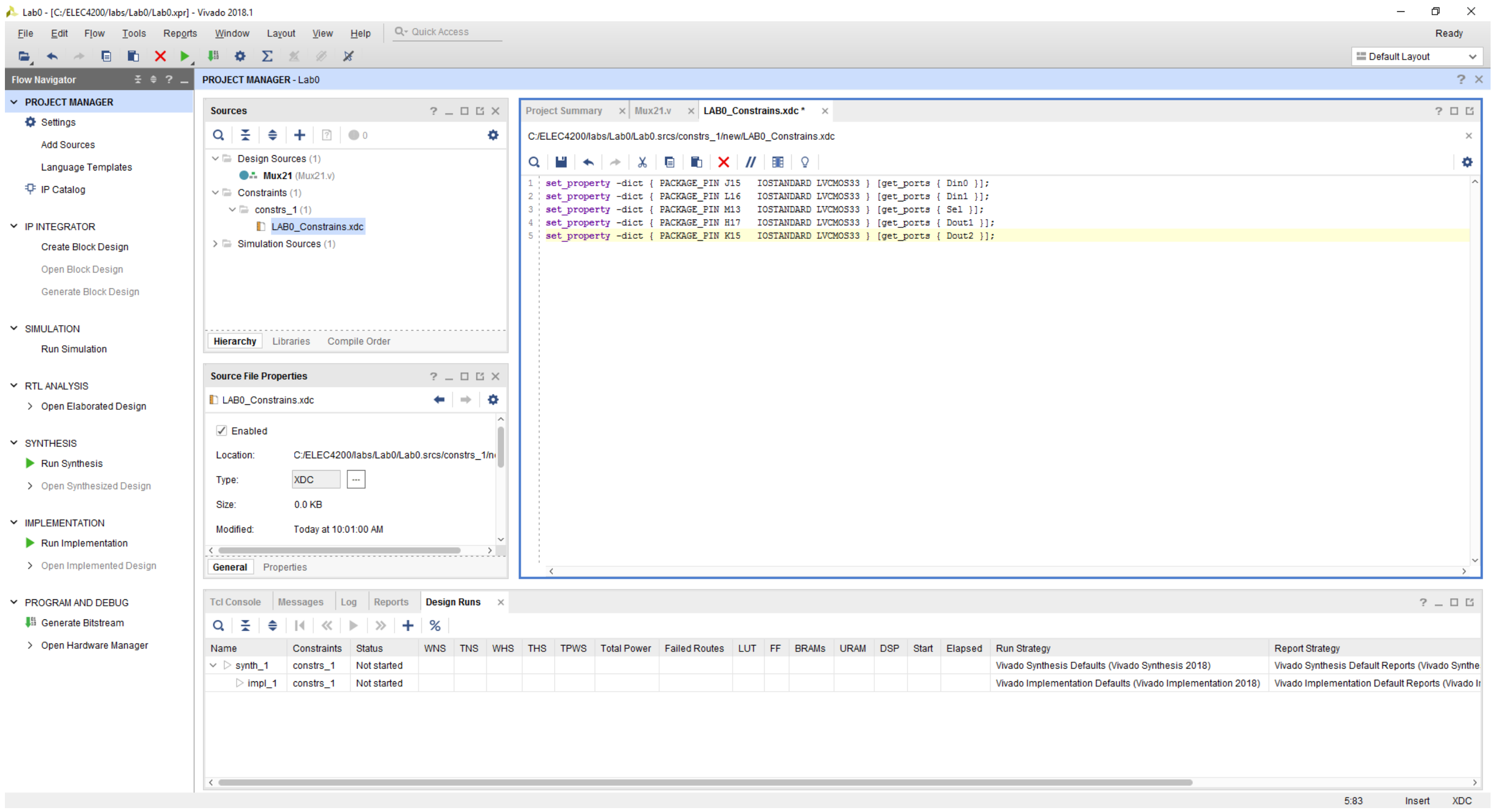
set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMOS33 } [get_ports { **Din0** }];

set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { **Din1** }];

set_property -dict { PACKAGE_PIN M13   IOSTANDARD LVCMOS33 } [get_ports { **Sel** }];

set_property -dict { PACKAGE_PIN H17   IOSTANDARD LVCMOS33 } [get_ports { **Dout1** }];

set_property -dict { PACKAGE_PIN K15   IOSTANDARD LVCMOS33 } [get_ports { **Dout2** }];



C:/ELEC4200/labs/Lab0/Lab0.srcs/constrs_1/new/LAB0_Constrains.xdc

```
set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMOS33 } [get_ports { Din0 }];
set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { Din1 }];
set_property -dict { PACKAGE_PIN M13   IOSTANDARD LVCMOS33 } [get_ports { Sel }];
set_property -dict { PACKAGE_PIN H17   IOSTANDARD LVCMOS33 } [get_ports { Dout1 }];
set_property -dict { PACKAGE_PIN K15   IOSTANDARD LVCMOS33 } [get_ports { Dout2 }];
```

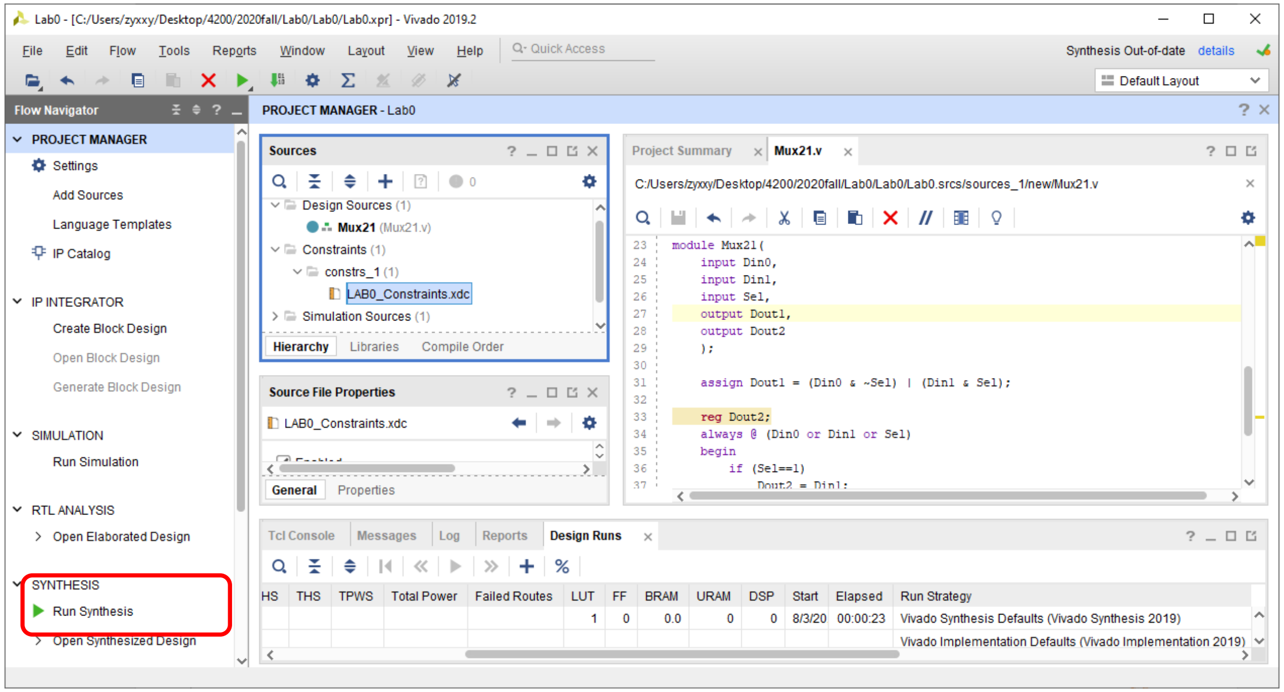
LAB0_Constrains.xdc

✓ Enabled	
Location:	C:/ELEC4200/labs/Lab0/Lab0.srcs/constrs_1/n
Type:	XDC
Size:	0.0 KB
Modified:	Today at 10:01:00 AM

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAMs	URAM	DSP	Start	Elapsed	Run Strategy	Report Strategy
synth_1	constrs_1	Not started															Vivado Synthesis Defaults (Vivado Synthesis 2018)	Vivado Synthesis Default Reports (Vivado Synthe
impl_1	constrs_1	Not started															Vivado Implementation Defaults (Vivado Implementation 2018)	Vivado Implementation Default Reports (Vivado I

Implement the design(1)

- Now you are going to have *Vivado* “synthesize” your design into generic digital logic components.
- In the Flow Navigator, click “Run Synthesis”.
- This may take several minutes to run, so be patient.
- A dialog box will open on completion, either select “Run Implementation” or hit “Cancel”.


Lab0 - [C:/Users/zyxxy/Desktop/4200/2020fall/Lab0/Lab0/Lab0.xpr] - Vivado 2019.2

Synthesis Out-of-date details

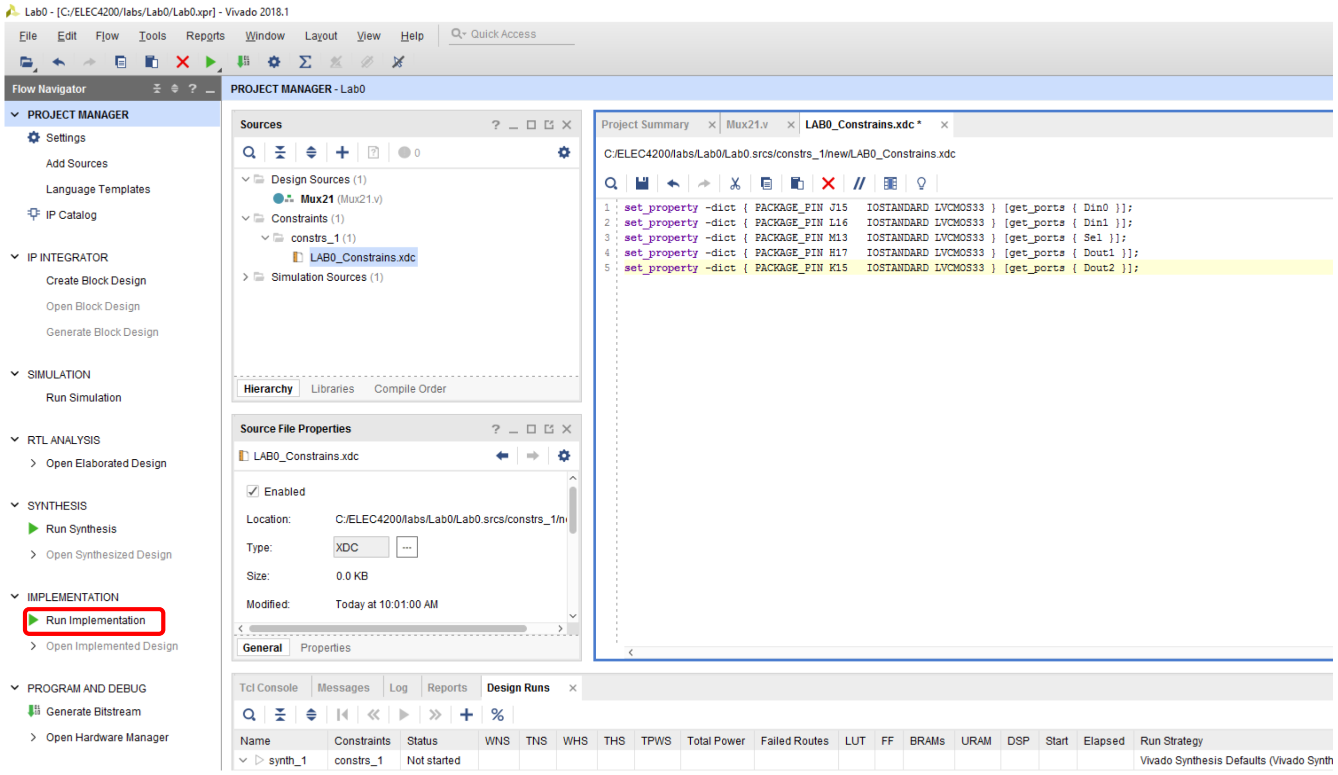
C:/Users/zyxxy/Desktop/4200/2020fall/Lab0/Lab0/Lab0.srcs/sources_1/new/Mux21.v

```
module Mux21(
    input Din0,
    input Din1,
    input Sel,
    output Dout1,
    output Dout2
    );

    assign Dout1 = (Din0 & ~Sel) | (Din1 & Sel);

    reg Dout2;
    always @ (Din0 or Din1 or Sel)
    begin
        if (Sel==1)
            Dout2 = Din1;
```

HS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP	Start	Elapsed	Run Strategy
					1	0	0.0	0	0	8/3/20	00:00:23	Vivado Synthesis Defaults (Vivado Synthesis 2019)
												Vivado Implementation Defaults (Vivado Implementation 2019)

Implement the design (2)

- Click on “Run Implementation”. This maps the synthesized design to the hardware, and routes I/O ports to the pins specified in the constraints file.
- Again, this will take several minutes to run.
- A dialog box will open on completion, either select “Generate Bitstream” or hit “Cancel”.



Source File Properties: LAB0_Constrains.xdc

✓ Enabled	
Location:	C:/ELEC4200/labs/Lab0/Lab0.srcs/constrs_1/n
Type:	XDC
Size:	0.0 KB
Modified:	Today at 10:01:00 AM

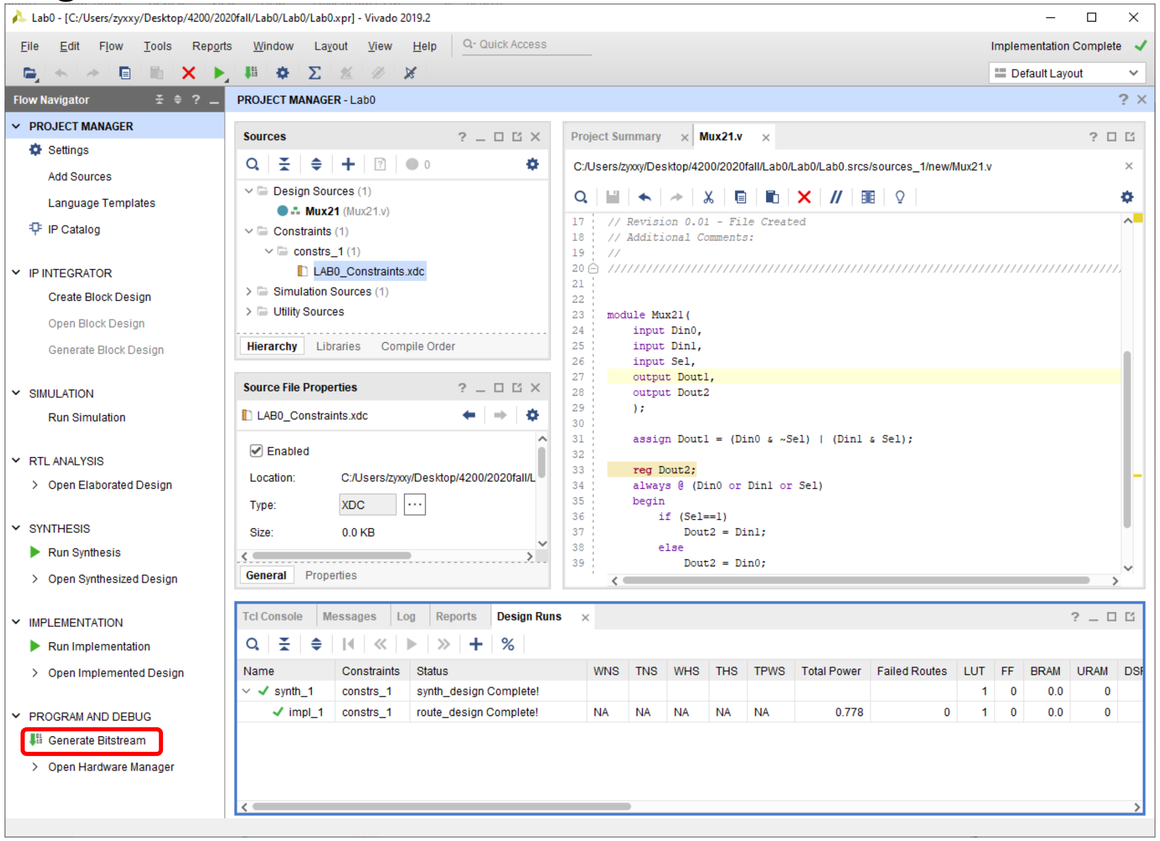
C:/ELEC4200/labs/Lab0/Lab0.srcs/constrs_1/new/LAB0_Constrains.xdc

```
set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMOS33 } [get_ports { Din0 }];
set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { Din1 }];
set_property -dict { PACKAGE_PIN M13   IOSTANDARD LVCMOS33 } [get_ports { Sel }];
set_property -dict { PACKAGE_PIN H17   IOSTANDARD LVCMOS33 } [get_ports { Dout1 }];
set_property -dict { PACKAGE_PIN K15   IOSTANDARD LVCMOS33 } [get_ports { Dout2 }];
```

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAMs	URAM	DSP	Start	Elapsed	Run Strategy
synth_1	constrs_1	Not started															Vivado Synthesis Defaults (Vivado Synth

Generate Bitstream

- With the design now implemented, it is time to generate the bitstream, or the file that will be downloaded to the FPGA.
- Click “Generate Bitstream”
- A dialog box will open on completion, either select “Open Hardware Manager” or hit “Cancel”.



LAB0_Constraints.xdc

☑ Enabled

Location: C:/Users/zyxxy/Desktop/4200/2020fall/L

Type: XDC

Size: 0.0 KB

C:/Users/zyxxy/Desktop/4200/2020fall/Lab0/Lab0/Lab0.srcs/sources_1/new/Mux21.v

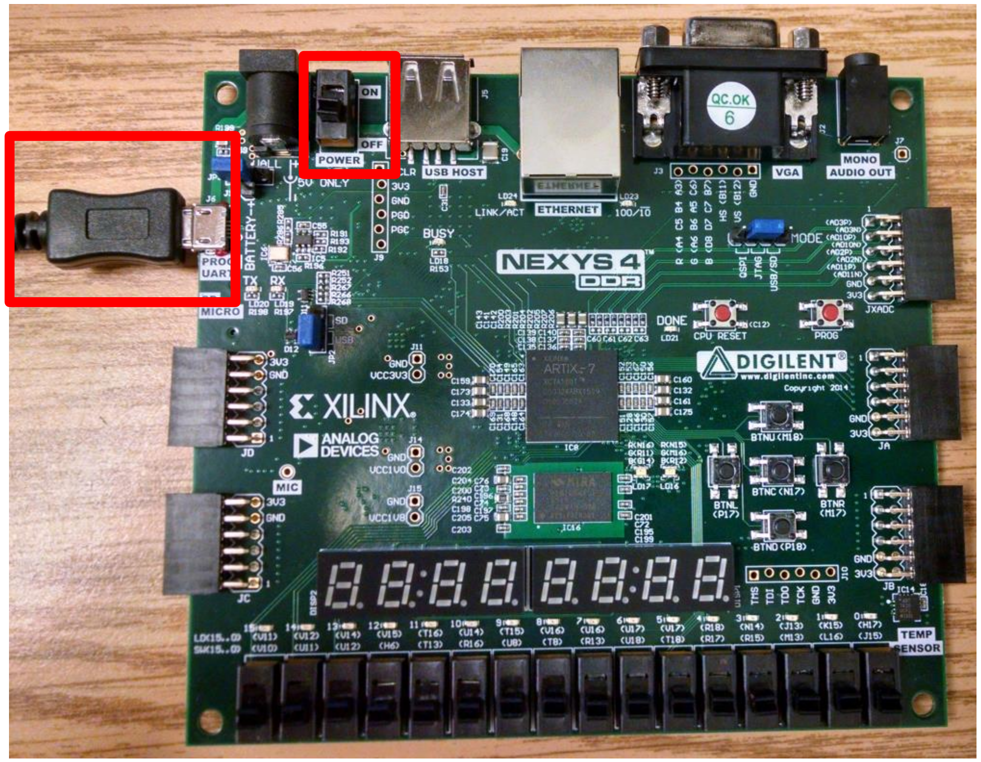
```
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////////////////////////////////////////////////


module Mux21(
    input Din0,
    input Din1,
    input Sel,
    output Dout1,
    output Dout2
    );

    assign Dout1 = (Din0 & ~Sel) | (Din1 & Sel);

    reg Dout2;
    always @ (Din0 or Din1 or Sel)
    begin
        if (Sel==1)
            Dout2 = Din1;
        else
            Dout2 = Din0;
```

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP
✓ synth_1	constrs_1	synth_design Complete!								1	0	0.0	0	
✓ impl_1	constrs_1	route_design Complete!	NA	NA	NA	NA	NA	0.778	0	1	0	0.0	0	

Download to Hardware(1)

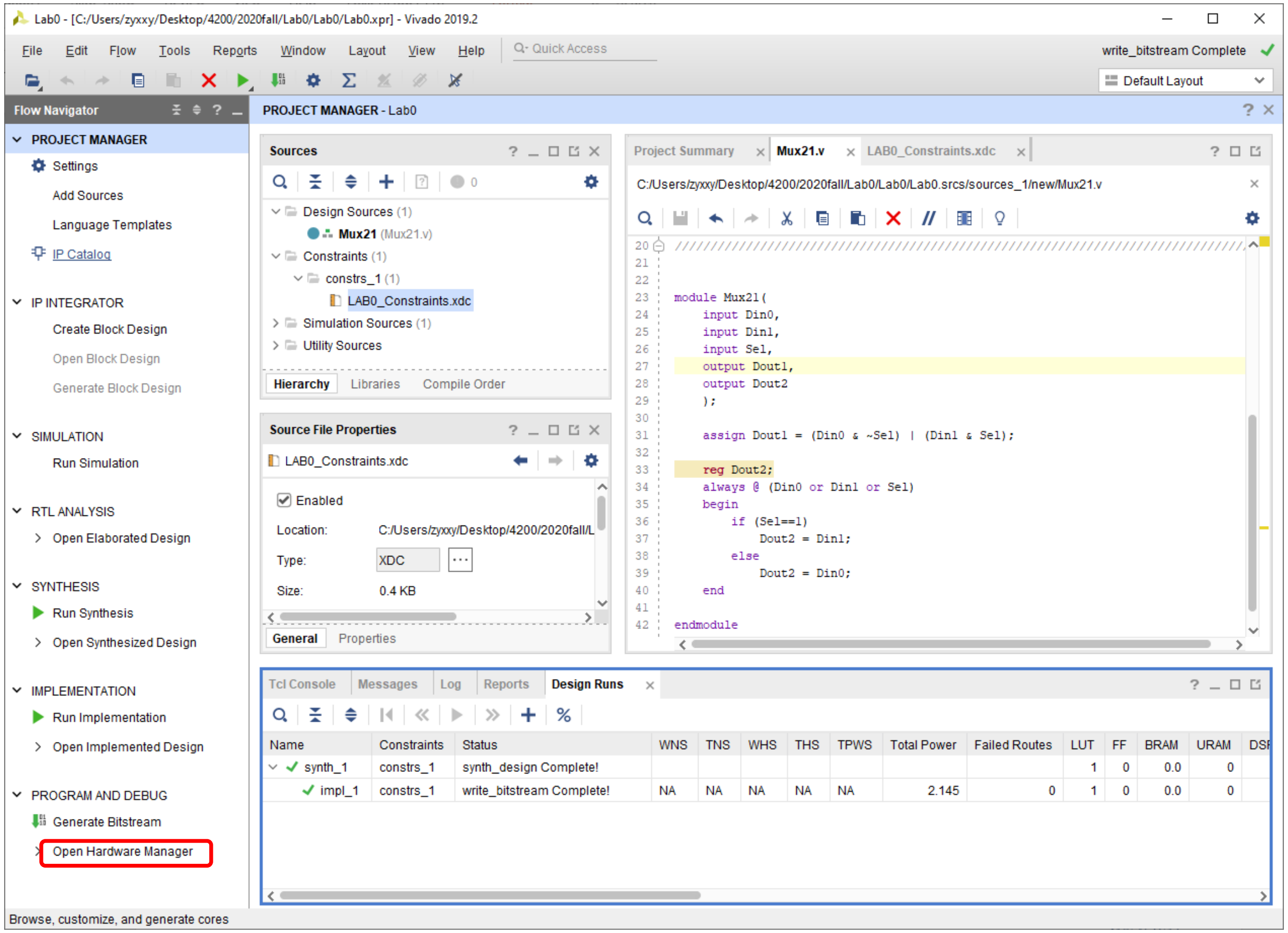
- Plug in the USB cable between a PC USB port and the USB port on the FPGA board to access the JTAG programming module.

- Flip the board power switch from OFF to ON



Download to Hardware (2)

- Click on “Hardware Manager”



Lab0 - [C:/Users/zyxxy/Desktop/4200/2020fall/Lab0/Lab0/Lab0.xpr] - Vivado 2019.2

File Edit Flow Tools Reports Window Layout View Help

write_bitstream Complete ✓

Default Layout

Flow Navigator

PROJECT MANAGER
- Settings
- Add Sources
- Language Templates
- IP Catalog

IP INTEGRATOR
- Create Block Design
- Open Block Design
- Generate Block Design

SIMULATION
- Run Simulation

RTL ANALYSIS
- Open Elaborated Design

SYNTHESIS
- Run Synthesis
- Open Synthesized Design

IMPLEMENTATION
- Run Implementation
- Open Implemented Design

PROGRAM AND DEBUG
- Generate Bitstream
- Open Hardware Manager

PROJECT MANAGER - Lab0

Sources

- Design Sources (1)
  - Mux21 (Mux21.v)
- Constraints (1)
  - constrs_1 (1)
    - LAB0_Constraints.xdc
- Simulation Sources (1)
- Utility Sources

Hierarchy | Libraries | Compile Order

Source File Properties

LAB0_Constraints.xdc

☑ Enabled

Location:	C:/Users/zyxxy/Desktop/4200/2020fall/L
Type:	XDC
Size:	0.4 KB

General | Properties

Project Summary | Mux21.v | LAB0_Constraints.xdc

C:/Users/zyxxy/Desktop/4200/2020fall/Lab0/Lab0/Lab0.srcs/sources_1/new/Mux21.v

```
//////////////////////////////////////////////////////////////////////////////


module Mux21(
    input Din0,
    input Din1,
    input Sel,
    output Dout1,
    output Dout2
    );

    assign Dout1 = (Din0 & ~Sel) | (Din1 & Sel);

    reg Dout2;
    always @ (Din0 or Din1 or Sel)
    begin
        if (Sel==1)
            Dout2 = Din1;
        else
            Dout2 = Din0;
    end

endmodule
```

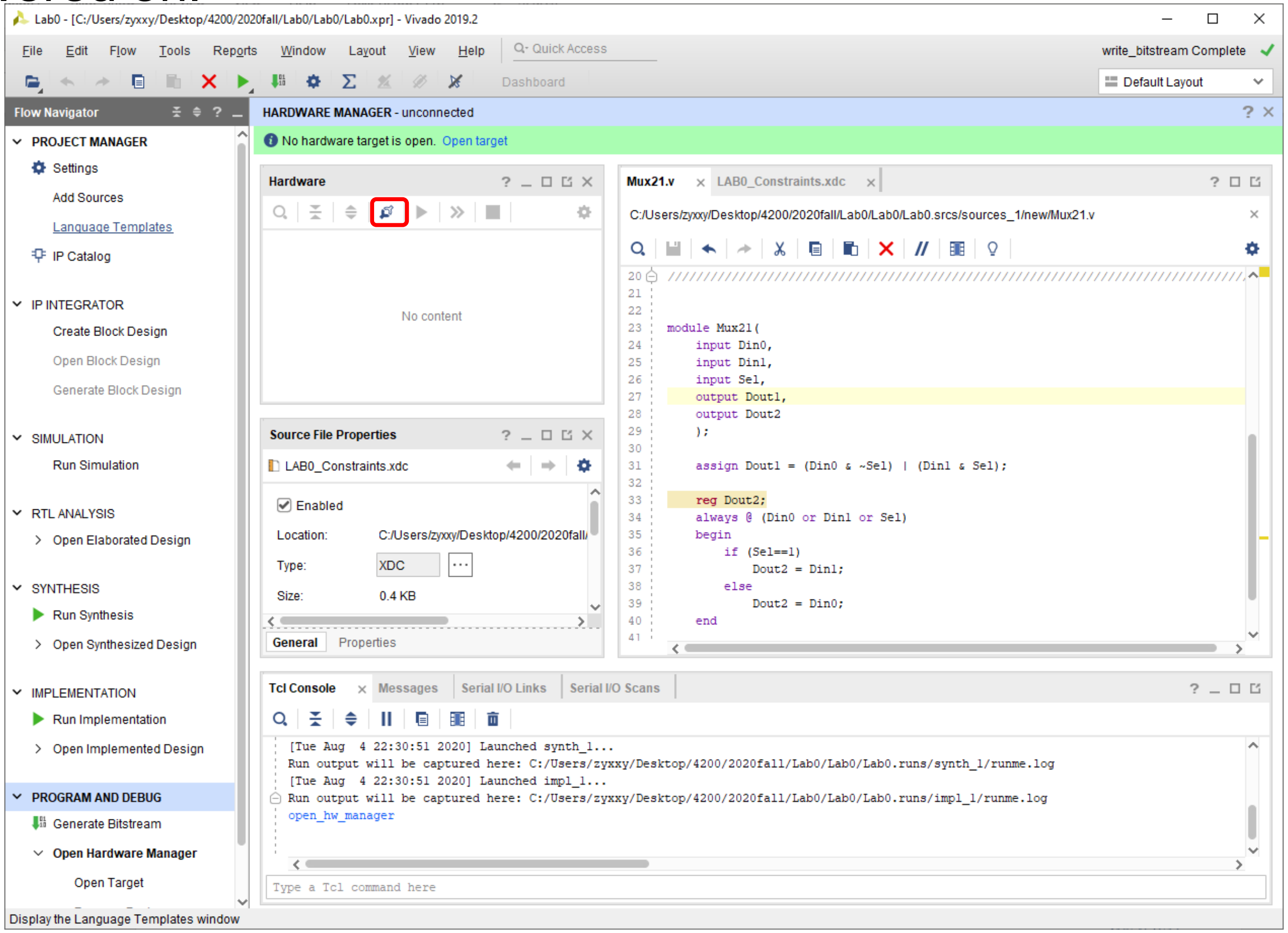
Tcl Console | Messages | Log | Reports | Design Runs

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP
✓ synth_1	constrs_1	synth_design Complete!								1	0	0.0	0	
✓ impl_1	constrs_1	write_bitstream Complete!	NA	NA	NA	NA	NA	2.145	0	1	0	0.0	0	

Browse, customize, and generate cores

Download to Hardware (3)

- Click “Auto Connect” to connect to your Nexys 4 board.
- If you encounter any errors, verify that the board is plugged in and powered on.



No hardware target is open. Open target

C:/Users/zyxxy/Desktop/4200/2020fall/Lab0/Lab0/Lab0.srcs/sources_1/new/Mux21.v

```
//////////////////////////////////////////////////////////////////////////////////


module Mux21(
    input Din0,
    input Din1,
    input Sel,
    output Dout1,
    output Dout2
    );

    assign Dout1 = (Din0 & ~Sel) | (Din1 & Sel);

    reg Dout2;
    always @ (Din0 or Din1 or Sel)
    begin
        if (Sel==1)
            Dout2 = Din1;
        else
            Dout2 = Din0;
    end
```

LAB0_Constraints.xdc

☑ Enabled

Location:	C:/Users/zyxxy/Desktop/4200/2020fall/
Type:	XDC
Size:	0.4 KB

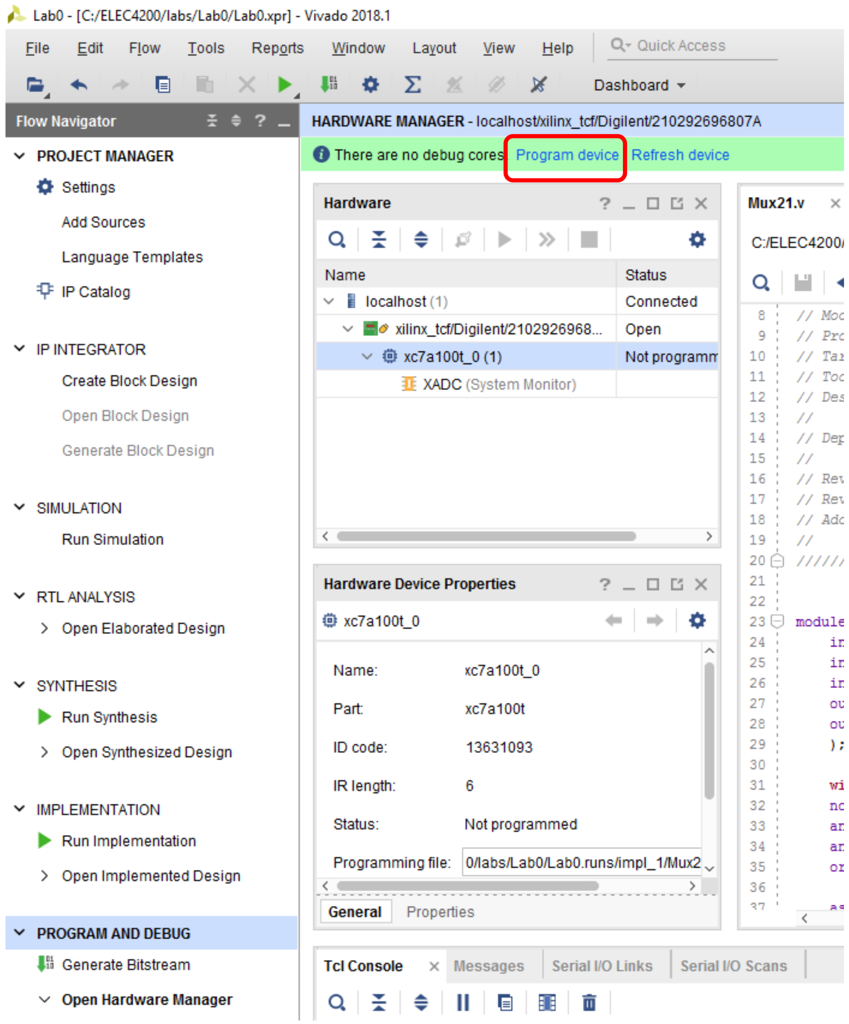
```
[Tue Aug  4 22:30:51 2020] Launched synth_1...
Run output will be captured here: C:/Users/zyxxy/Desktop/4200/2020fall/Lab0/Lab0/Lab0.runs/synth_1/runme.log
[Tue Aug  4 22:30:51 2020] Launched impl_1...
Run output will be captured here: C:/Users/zyxxy/Desktop/4200/2020fall/Lab0/Lab0/Lab0.runs/impl_1/runme.log
open_hw_manager
```

Type a Tcl command here

Display the Language Templates window

Download to Hardware(4)

- Click “Program Device” and select xc7a100t_0 from the list.



Lab0 - [C:/ELEC4200/labs/Lab0/Lab0.xpr] - Vivado 2018.1

HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210292696807A

There are no debug cores

Name	Status
localhost (1)	Connected
xilinx_tcf/Digilent/2102926968...	Open
xc7a100t_0 (1)	Not programm
XADC (System Monitor)	

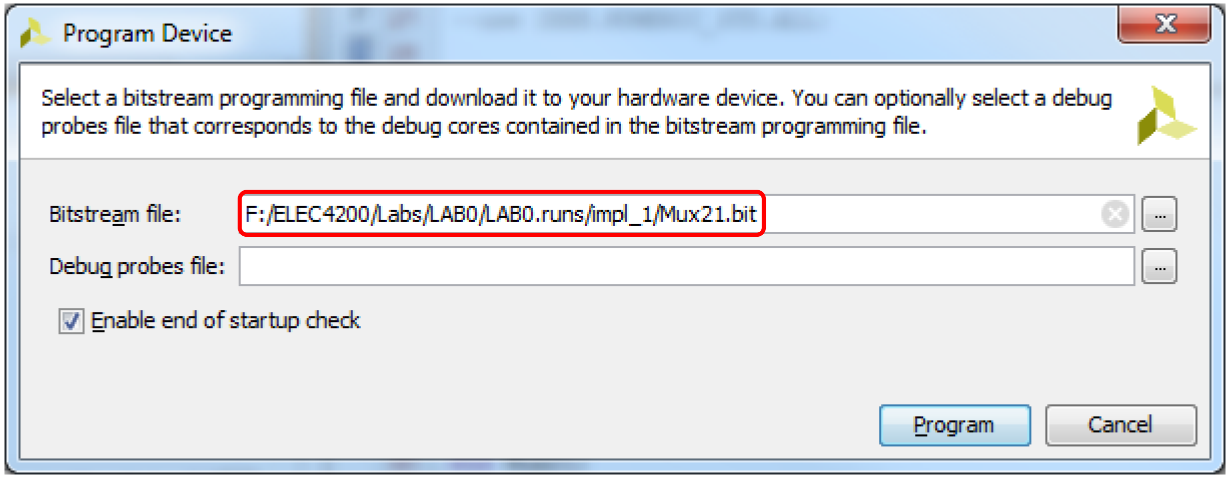
xc7a100t_0

Name:	xc7a100t_0
Part:	xc7a100t
ID code:	13631093
IR length:	6
Status:	Not programmed
Programming file:	0/labs/Lab0/Lab0.runs/impl_1/Mux2

C:/ELEC4200/

Download to Hardware(5)

- In the dialog box, make sure the correct bit file is selected.
- Leave the “Debug probes file” box empty for now.
- Click “Program”



Program Device

Select a bitstream programming file and download it to your hardware device. You can optionally select a debug probes file that corresponds to the debug cores contained in the bitstream programming file.

Bitstream file: F:/ELEC4200/Labs/LAB0/LAB0.runs/impl_1/Mux21.bit

Debug probes file:

☑ Enable end of startup check

Download to Hardware(6)

- Verify correct operation of the circuit using the switches and observe the output on the LEDs. If you encounter any bugs, attempt to figure out what went wrong before asking for help.
- Ensure that you apply all possible input combinations, as you did in simulation, and verify that the outputs match the simulation results.
- After verifying the circuit is correct, call the GTA over and demonstrate the circuit.

Clean up

- Turn off the board, unplug the USB cable, put them back in their box, and return the box to the shelf.
- Close Vivado, any other open programs, and save all files.
- Don’t forget to log out of your machine and take any USB drives with you.