Xilinx Vivado 2018.2 Design Tool Guide (Synthesis, Implementation & Programming FPGA)

Logic and Digital System Design – CS 303

Sabanci University

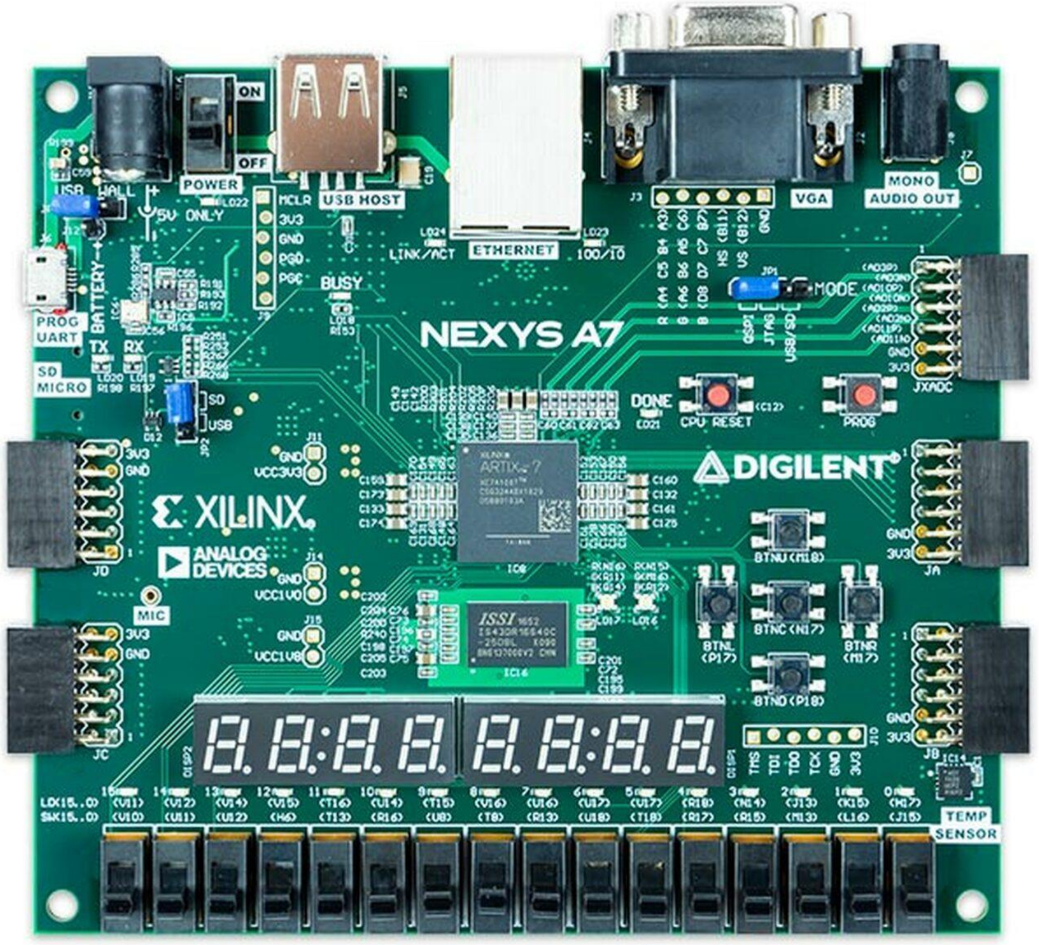
Fall 2021

Synthesis, Implementation & Programming FPGA

- This guide shows you how to perform synthesis/implementation operations for your design and read the synthesis/implementation reports.
- It shows how to generate a bit file to program FPGA.

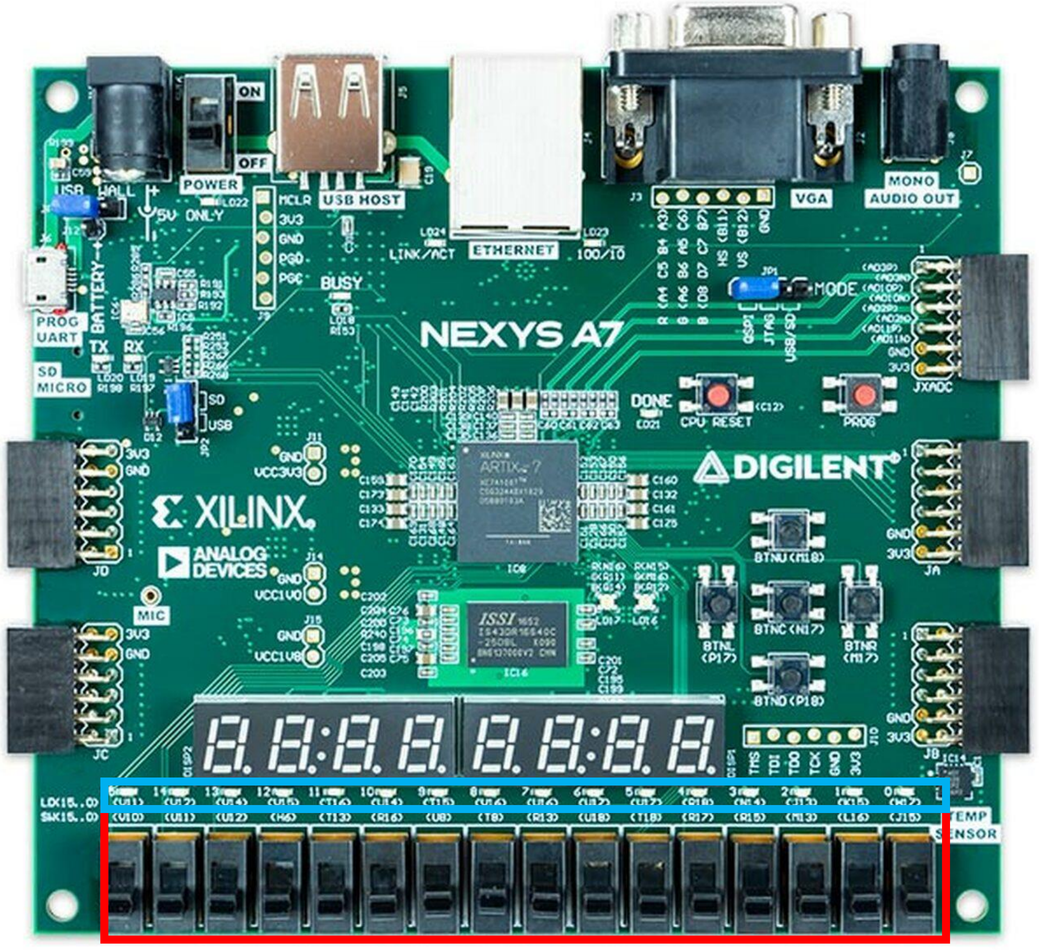
Nexys A7 – 100T FPGA Board

- You will use Nexys A7 – 100T FPGA board to implement your design.
- It will be distributed in the lab sessions and will be taken back after you demonstrate your work on it.



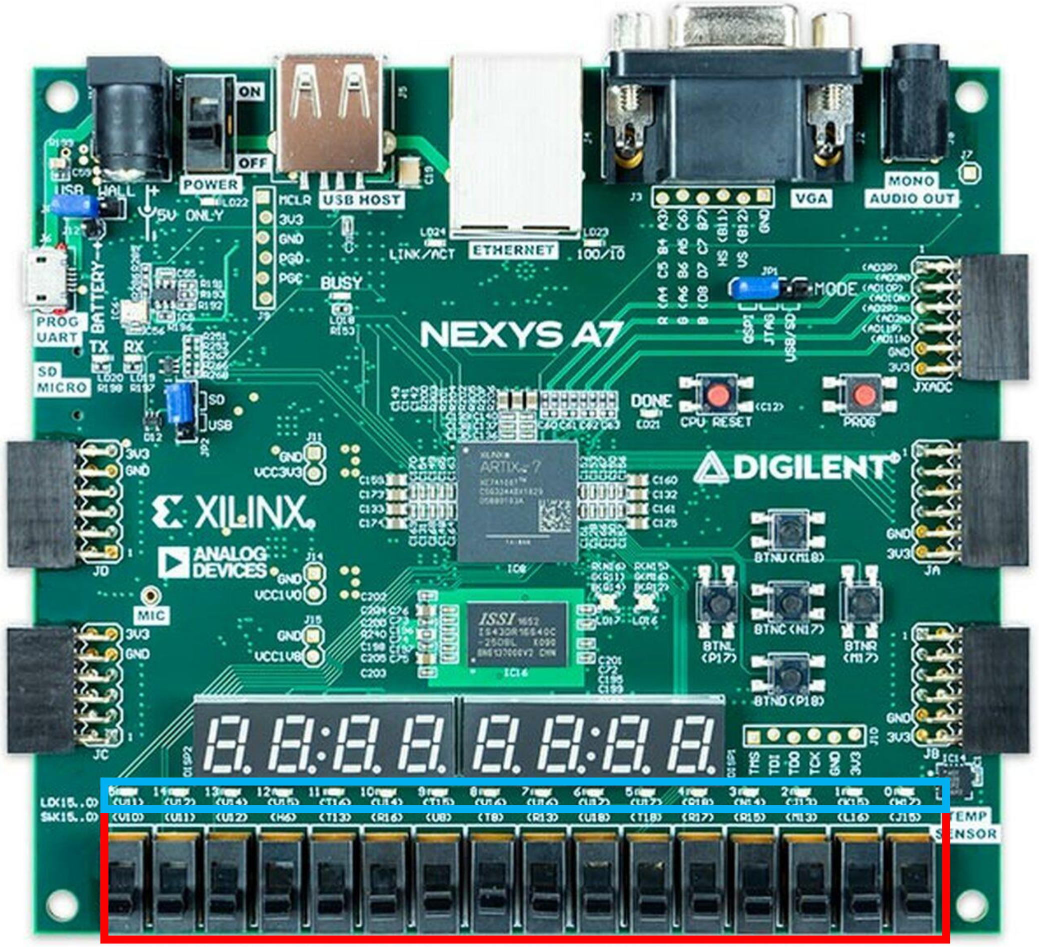
Nexys A7 – 100T FPGA Board

- 16 Switches and 16 LEDs
- Physical pins for switches: J15, L16 etc.
- Physical pins for LEDs: H17, K15 etc.
- Pins represent the components on the FPGA.
- J15 => Switch 0
- H17 => LED 0



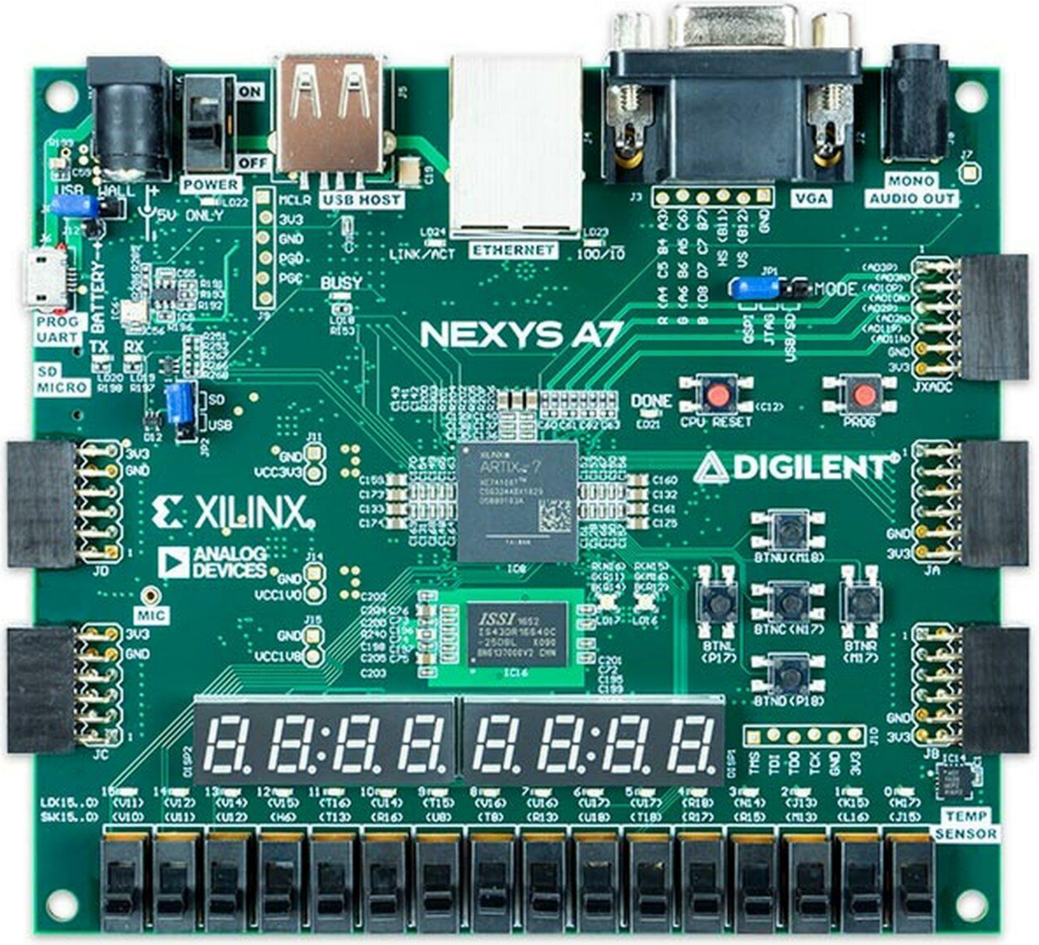
Nexys A7 – 100T FPGA Board

- Switches can be used for driving input values.
- LEDs can be used for observing output values.



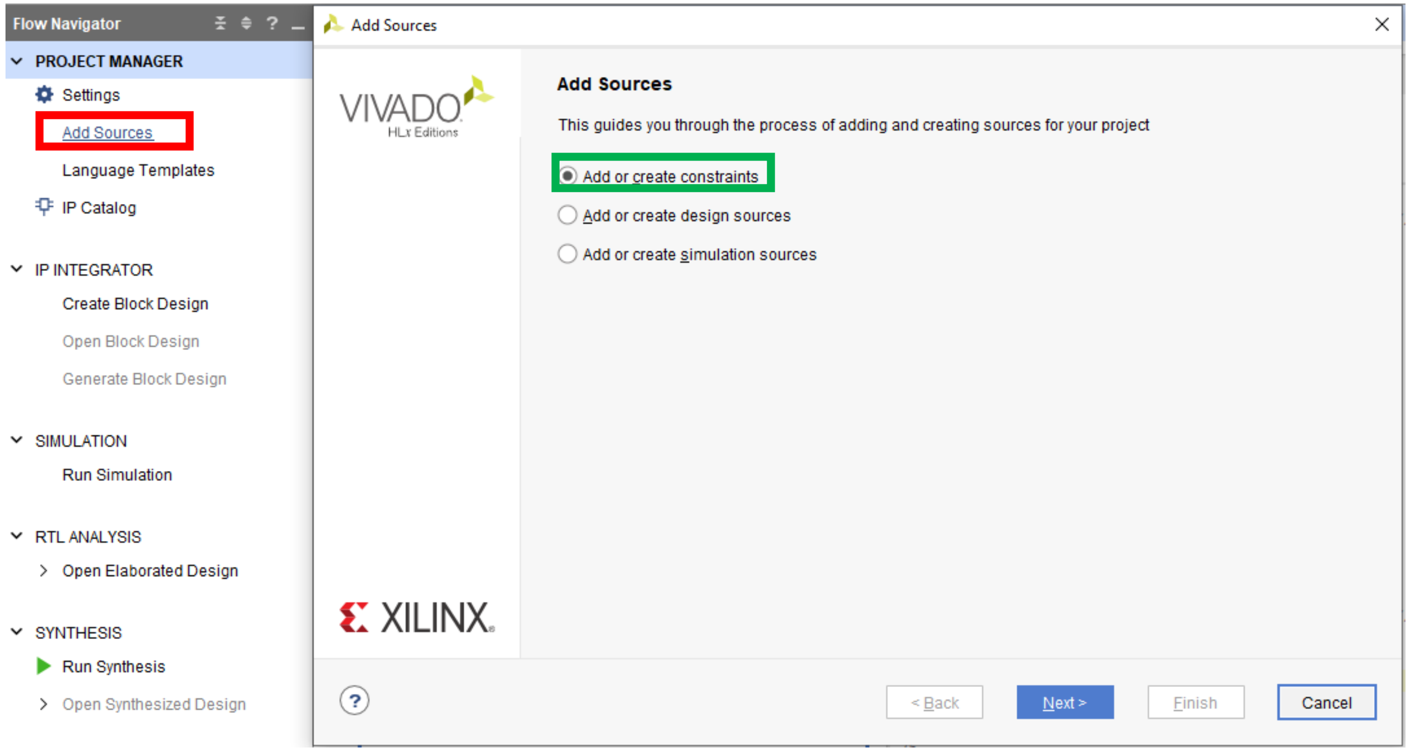
Synthesis

- When programming an FPGA through software such as Xilinx's Vivado, you need to inform the software what physical pins on the FPGA that you plan on using or connecting to in relation to the HDL code that you wrote to describe the behavior of the FPGA.



Synthesis

- You need to create a constraint file.
- Click on Add Source button
- Choose Add or create constraints
- Click on *Next* button



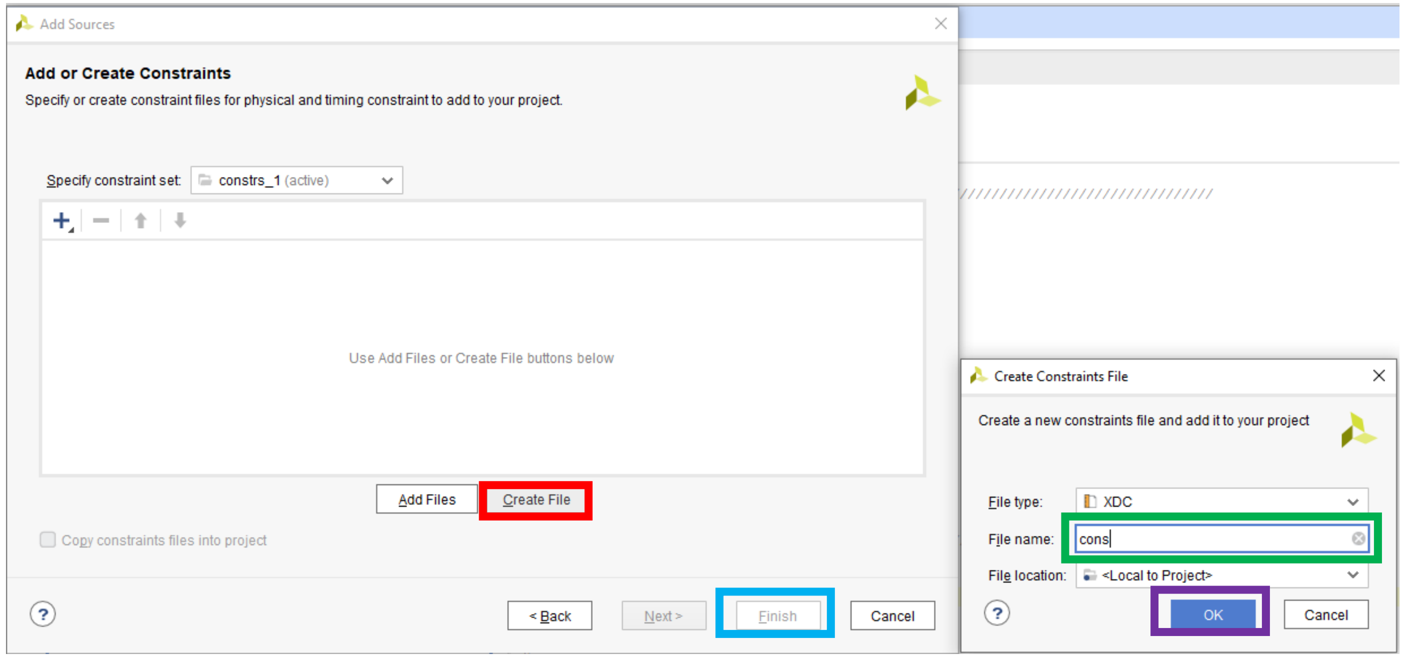
Add Sources

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

Synthesis

- Click on Create File
- Give a name to constraint file
- Click on OK
- Click on *Finish* button


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

Copy constraints files into project

Create Constraints File

Create a new constraints file and add it to your project

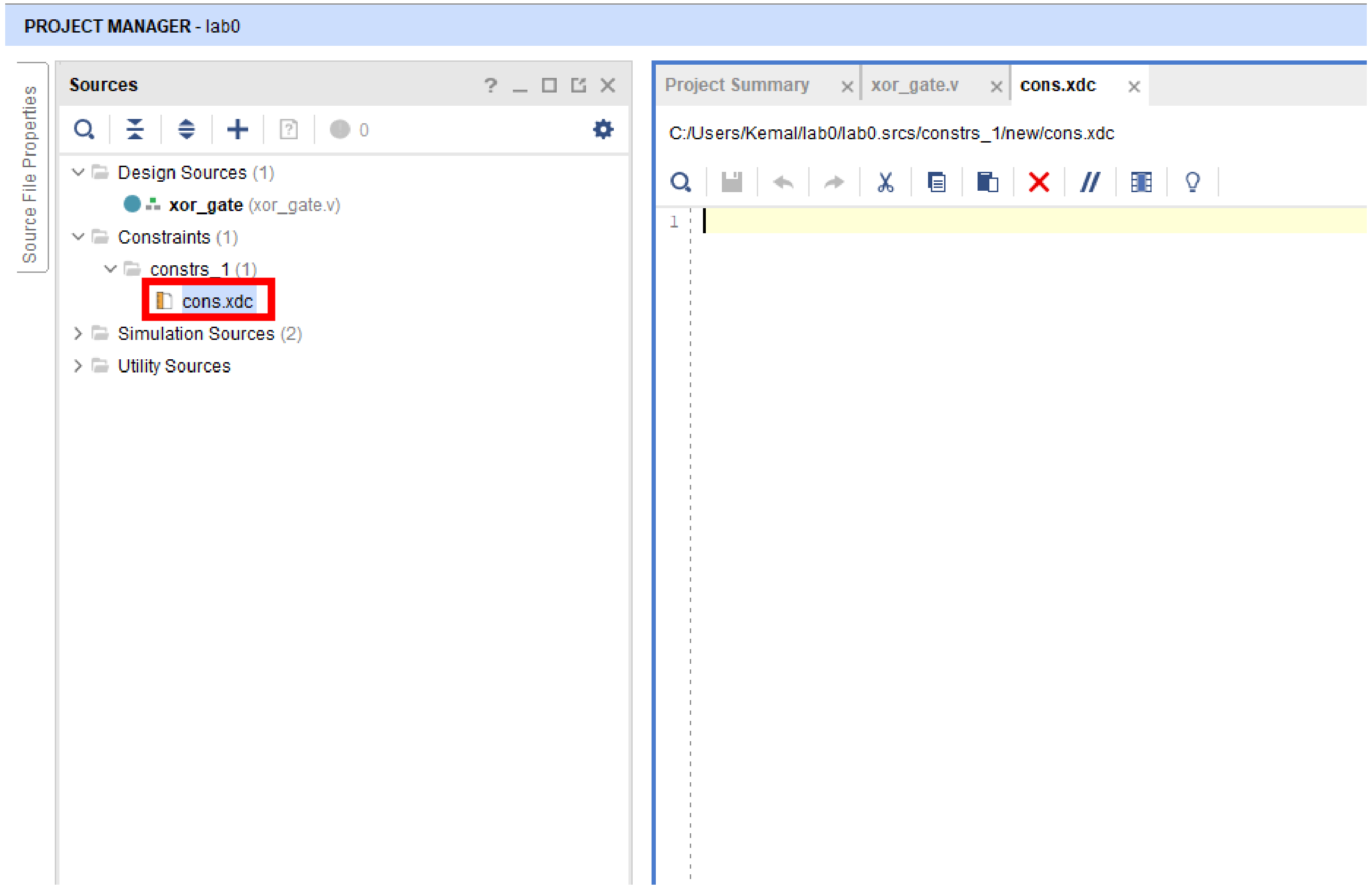
File type: XDC

File name: cons

File location: <Local to Project>

Synthesis

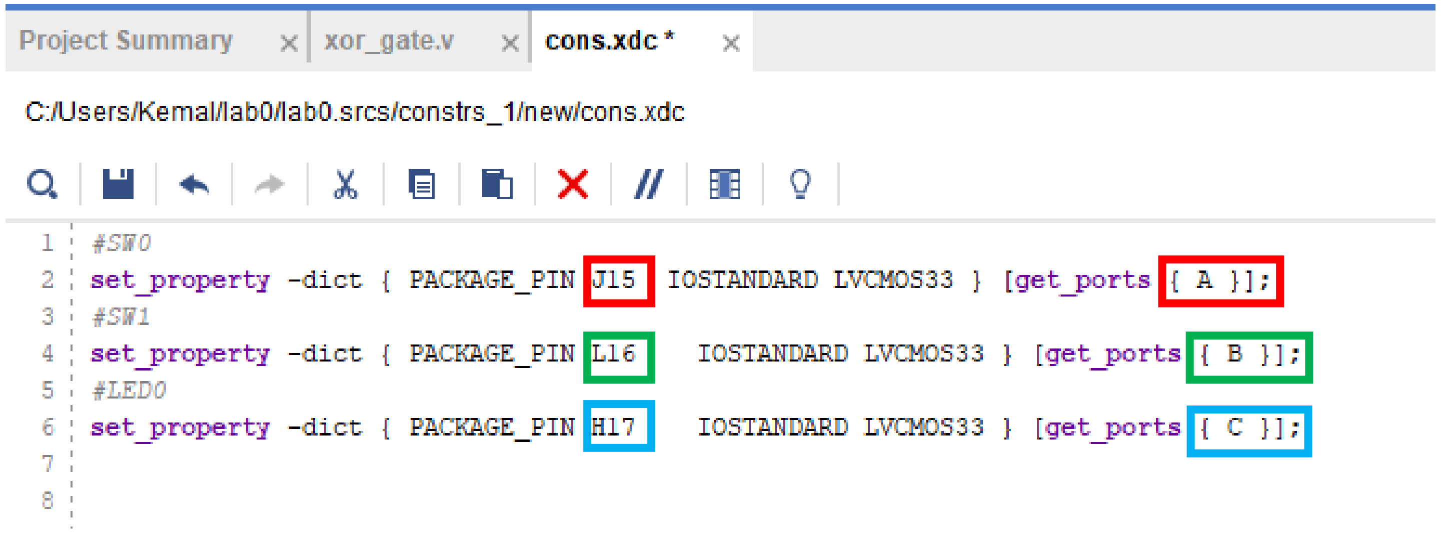
- Double click on your constraint file
- The editor will open it



C:/Users/Kemal/lab0/lab0.srcs/constrs_1/new/cons.xdc

Synthesis

- You need to specify the pins you want to use.
- J15 => Switch 0 => Input A
- L16 => Switch 1 => Input B
- H17 => LED 0 => Output C
- PACKAGE_PIN refers to the physical pin on the FPGA.
- IOSTANDART LVCMOS33 refers to the voltage standard for I/O ports.

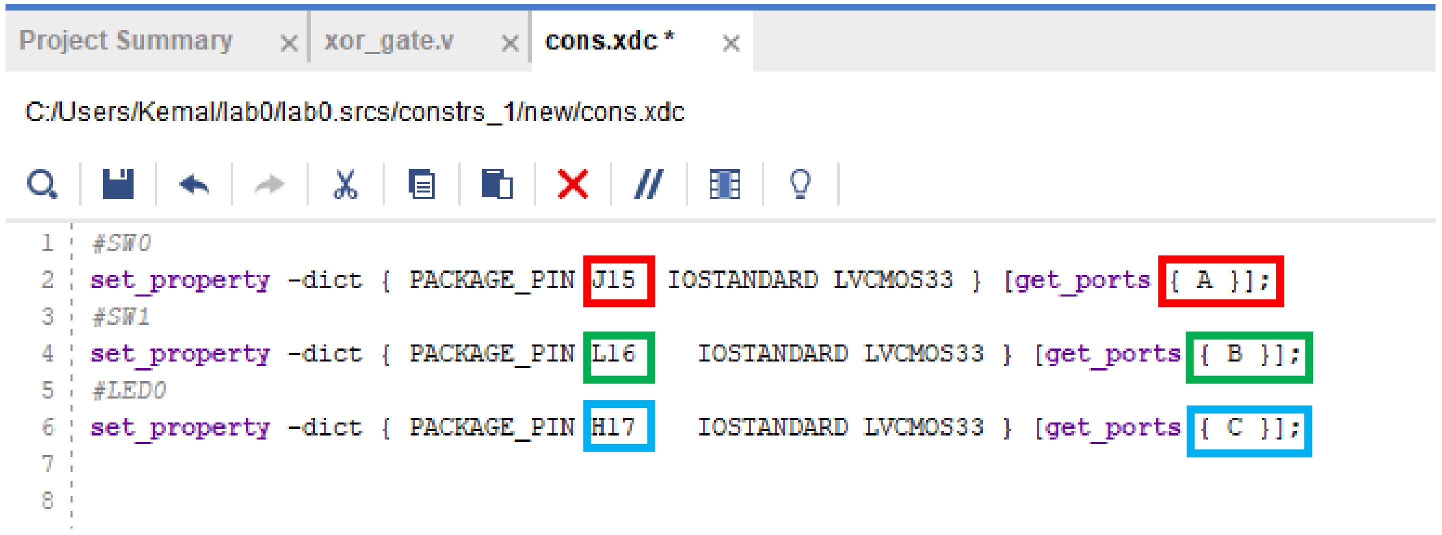


C:/Users/Kemal/lab0/lab0.srcs/constrs_1/new/cons.xdc

```
#SW0
set_property -dict { PACKAGE_PIN J15 IOSTANDARD LVCMOS33 } [get_ports { A }];
#SW1
set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { B }];
#LED0
set_property -dict { PACKAGE_PIN H17   IOSTANDARD LVCMOS33 } [get_ports { C }];
```

Synthesis

- You need to specify the input and output signals of your design.
- I/O names should match with your signal names.
- For example: If you use an input signal D, you need to specify it here and refer it to a physical pin.
- You need to change only the name of physical pins and the name of I/O signals.
- This example has 2 input and 1 output signals. You might have different number of them. In that case, you need to specify all of them as shown in this example.



C:/Users/Kemal/lab0/lab0.srcs/constrs_1/new/cons.xdc

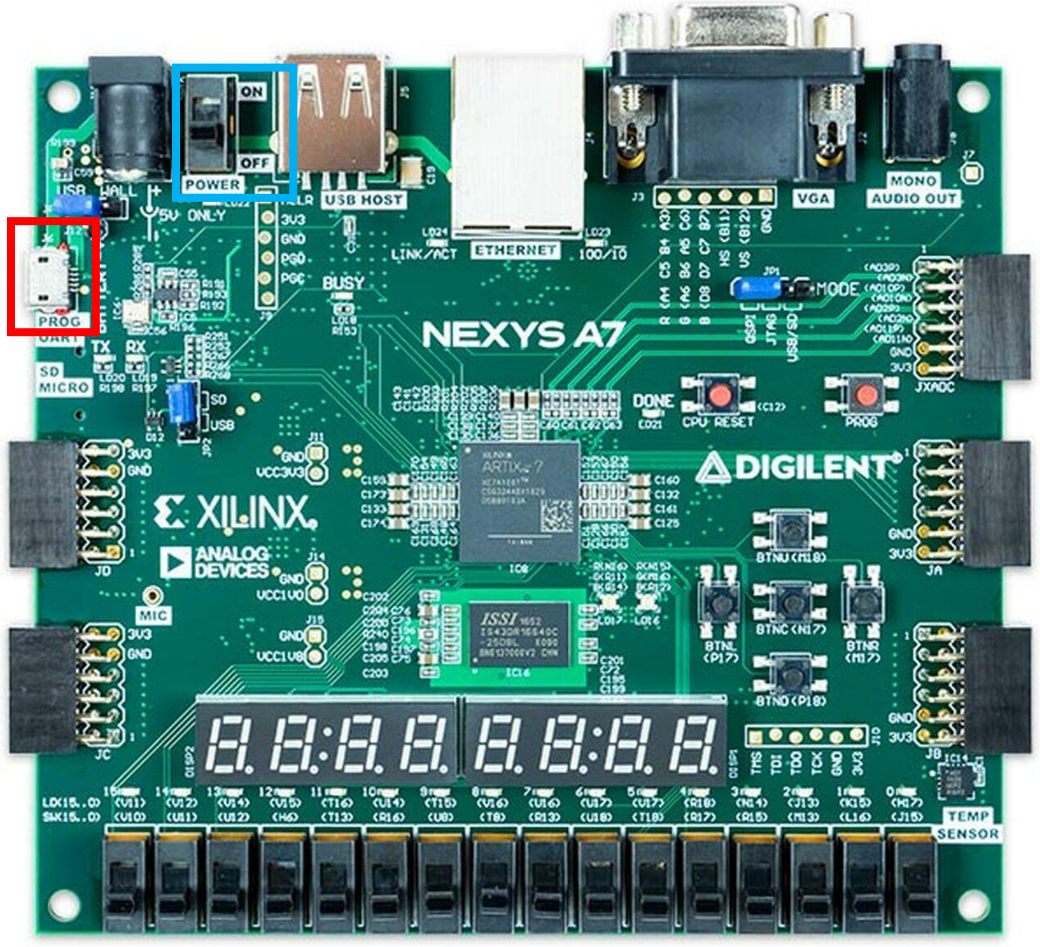
```
#SW0
set_property -dict { PACKAGE_PIN J15 IOSTANDARD LVCMOS33 } [get_ports { A }];
#SW1
set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { B }];
#LED0
set_property -dict { PACKAGE_PIN H17   IOSTANDARD LVCMOS33 } [get_ports { C }];
```

Constraint File

```
#SW0
set_property -dict { PACKAGE_PIN J15  IOSTANDARD LVCMOS33 }
[get_ports { A }];
#SW1
set_property -dict { PACKAGE_PIN L16    IOSTANDARD LVCMOS33 }
[get_ports { B }];
#LED0
set_property -dict { PACKAGE_PIN H17    IOSTANDARD LVCMOS33 }
[get_ports { C }];
```

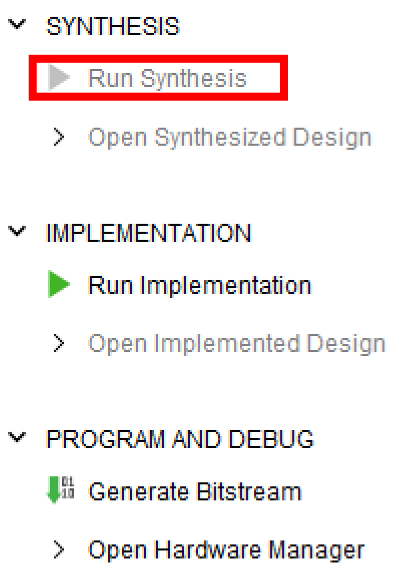
Nexys A7 – 100T FPGA Board

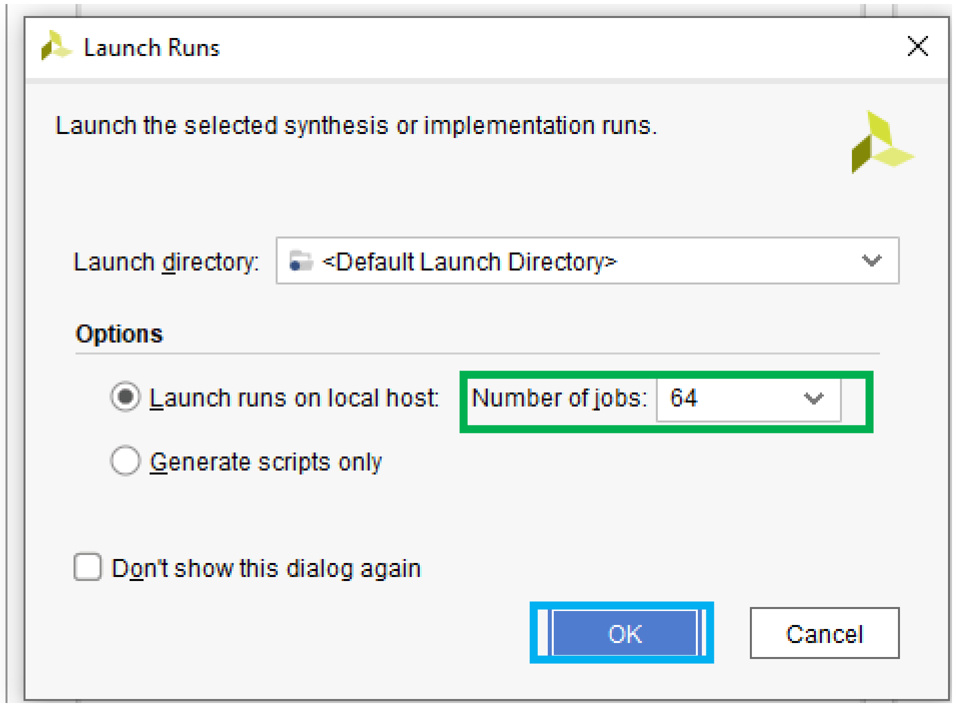
- The board comes with a USB connector. Connect the board to your computer by using USB cable. Connection port is shown in the red box.
- After connection, you need power on the board as shown in the box.



Synthesis

- Save your constraint file.
- Click on Run Synthesis
- You can choose the number of jobs. The higher number results in faster synthesis time. It is limited to the number of threads that your CPU has.
- Click on *OK* button




Launch Runs

Launch the selected synthesis or implementation runs.

Launch directory: <Default Launch Directory>

Options

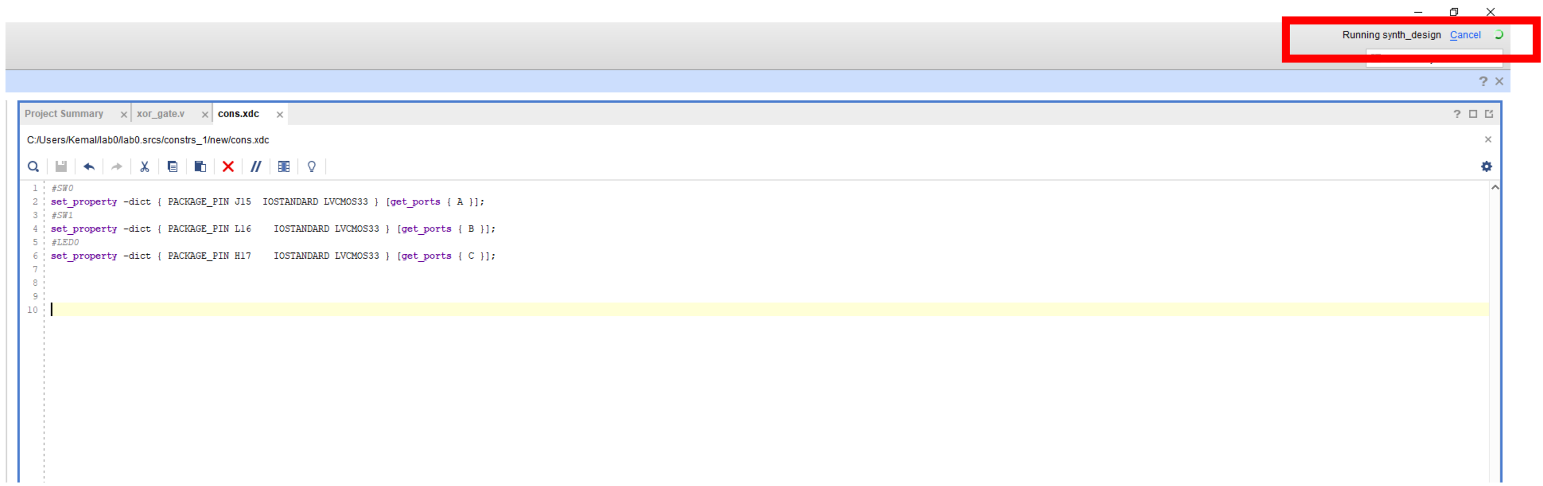
[✓ Launch runs on local host:] Number of jobs: 64

Generate scripts only

Don't show this dialog again

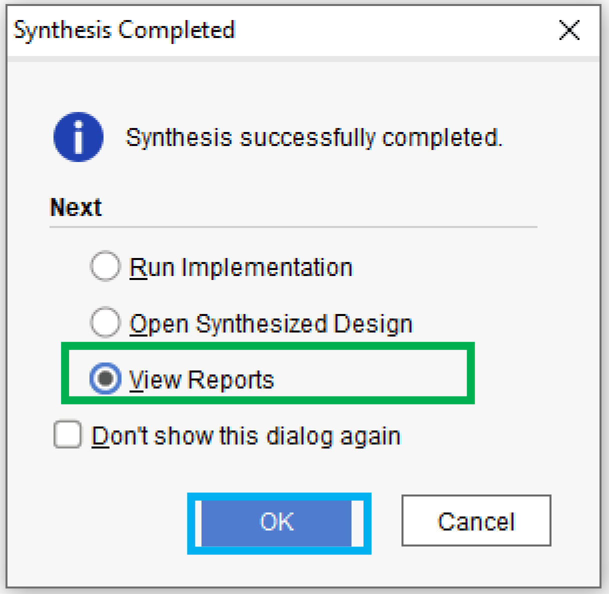
Synthesis

- Synthesis starts running.
- After it finishes, a window opens.
- Click on View Reports.
- Click on *OK* button



C:/Users/Kemal/lab0/lab0.srcs/constrs_1/new/cons.xdc

```
#SW0
set_property -dict { PACKAGE_PIN J15  IOSTANDARD LVCMOS33 } [get_ports { A }];
#SW1
set_property -dict { PACKAGE_PIN L16    IOSTANDARD LVCMOS33 } [get_ports { B }];
#LED0
set_property -dict { PACKAGE_PIN H17    IOSTANDARD LVCMOS33 } [get_ports { C }];
```



Synthesis Completed

Synthesis successfully completed.

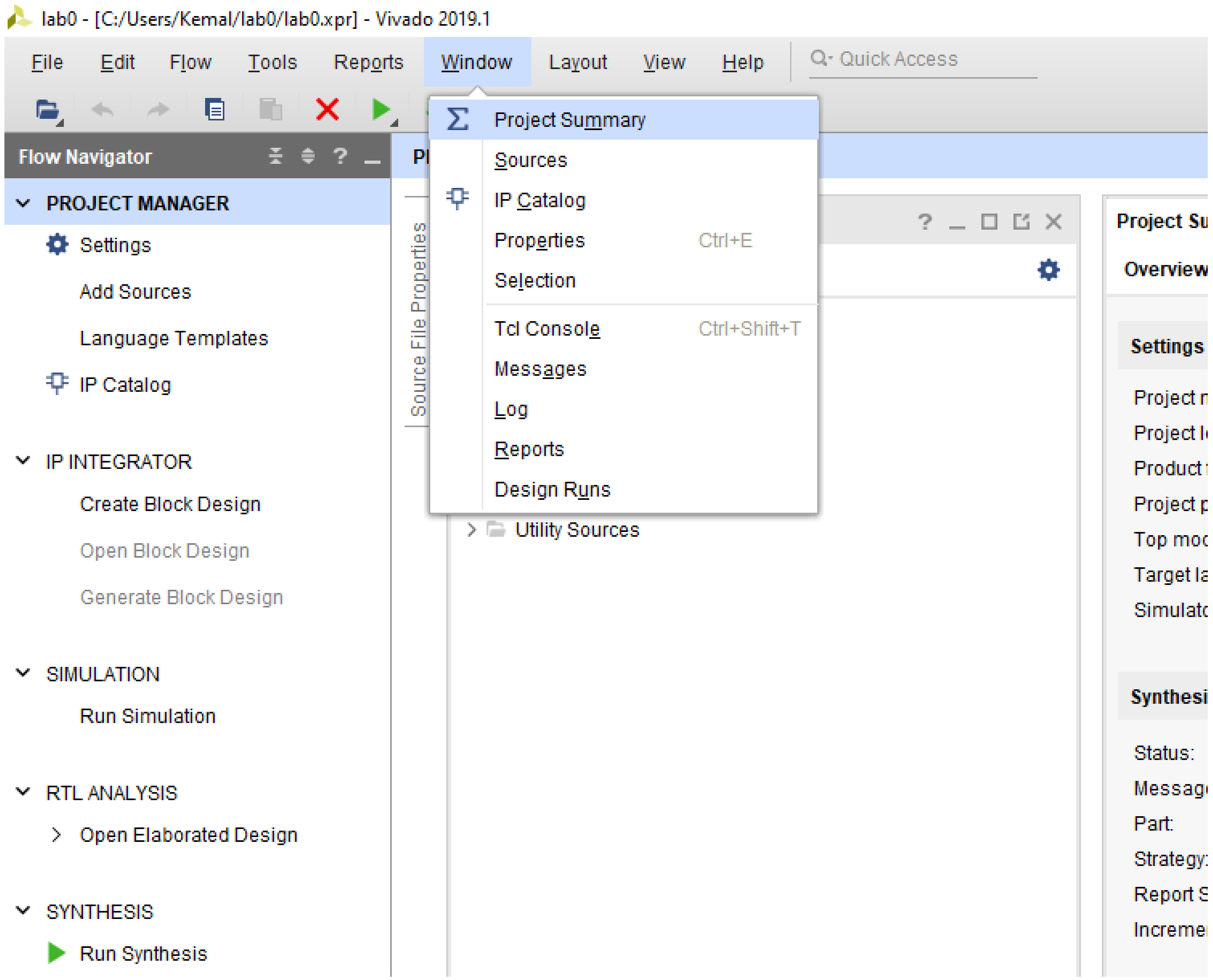
Next

- ○ Run Implementation
- ○ Open Synthesized Design
- ● View Reports

☐ Don't show this dialog again

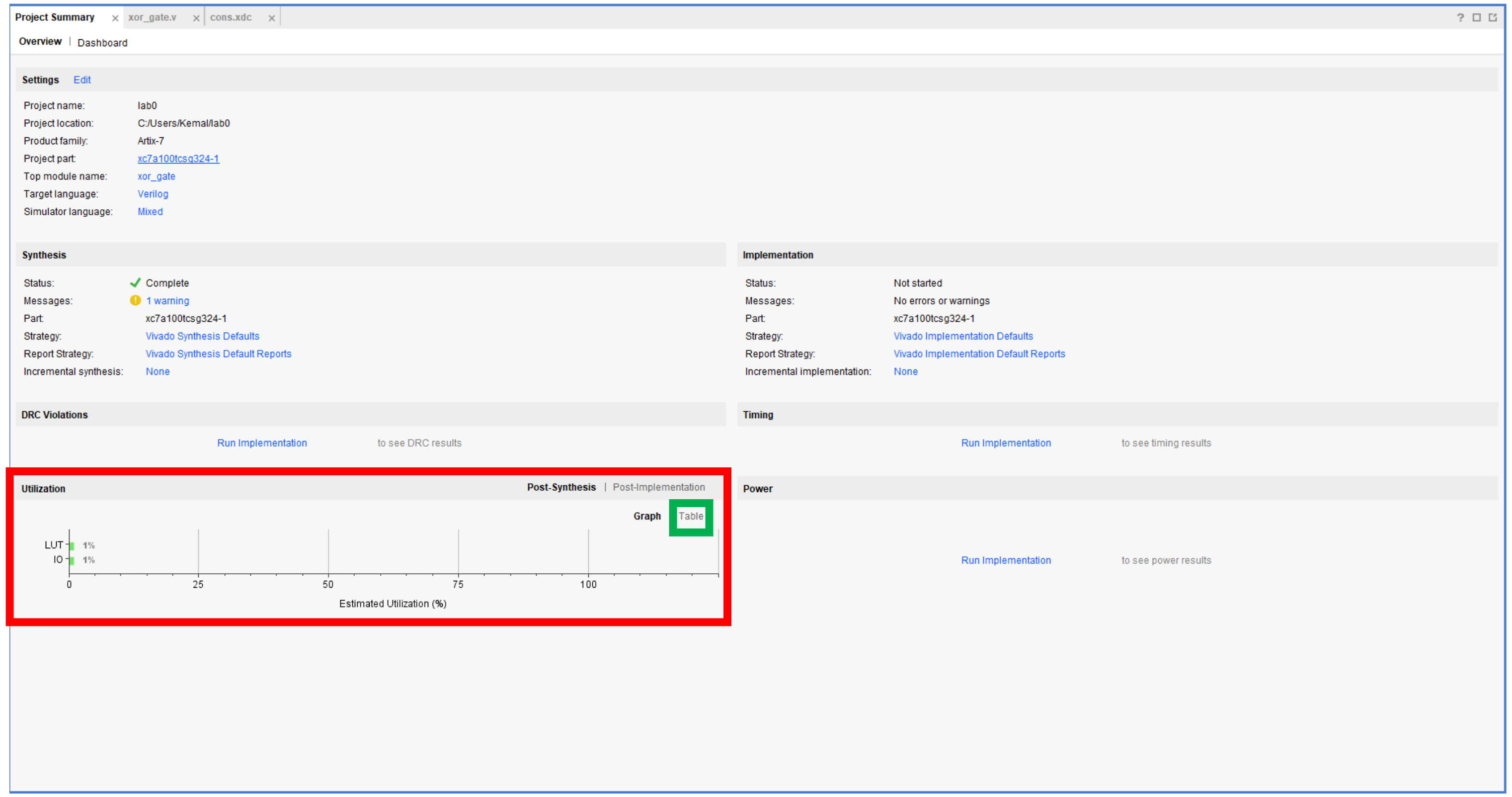
Synthesis

- Go to Project Summary



Synthesis

- Project Summary shows the utilization rates for the FPGA
- Click on Table to see how much resources your design uses after synthesis.


Project Summary | xor_gate.v | cons.xdc

Overview | Dashboard

Settings Edit

Project name:	lab0
Project location:	C:/Users/Kemal/lab0
Product family:	Artix-7
Project part:	xc7a100tcsg324-1
Top module name:	xor_gate
Target language:	Verilog
Simulator language:	Mixed

Synthesis

Status:	✓ Complete
Messages:	1 warning
Part:	xc7a100tcsg324-1
Strategy:	Vivado Synthesis Defaults
Report Strategy:	Vivado Synthesis Default Reports
Incremental synthesis:	None

Implementation

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

DRC Violations

Run Implementation to see DRC results

Timing

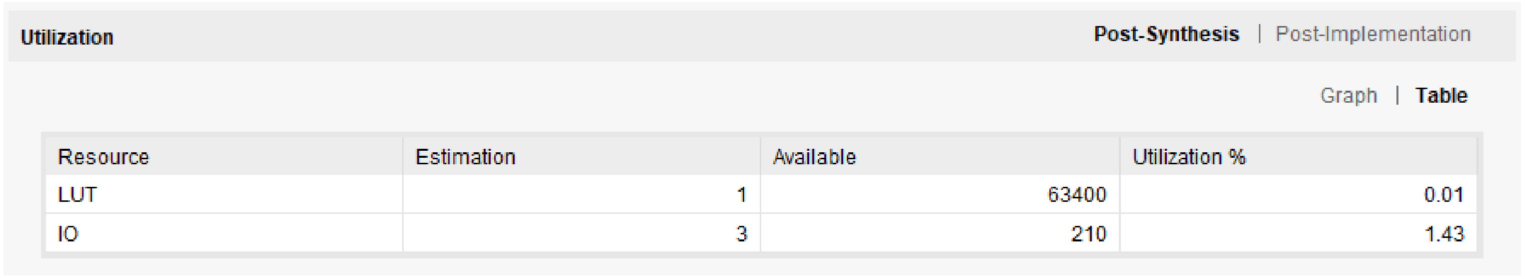
Run Implementation to see timing results

Utilization Post-Synthesis | Post-Implementation

Graph Table

Power

Run Implementation to see power results

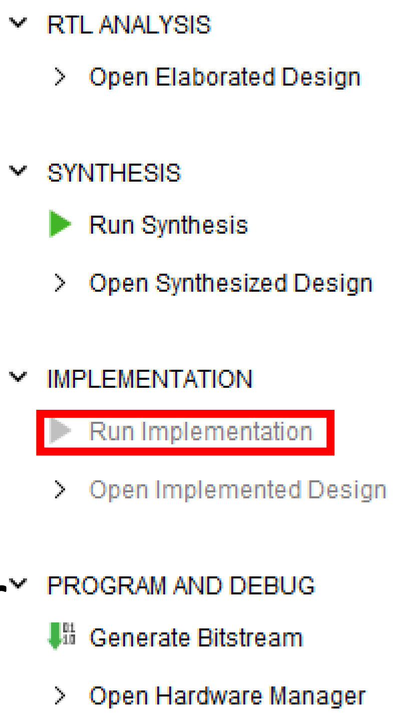

Utilization **Post-Synthesis** | Post-Implementation

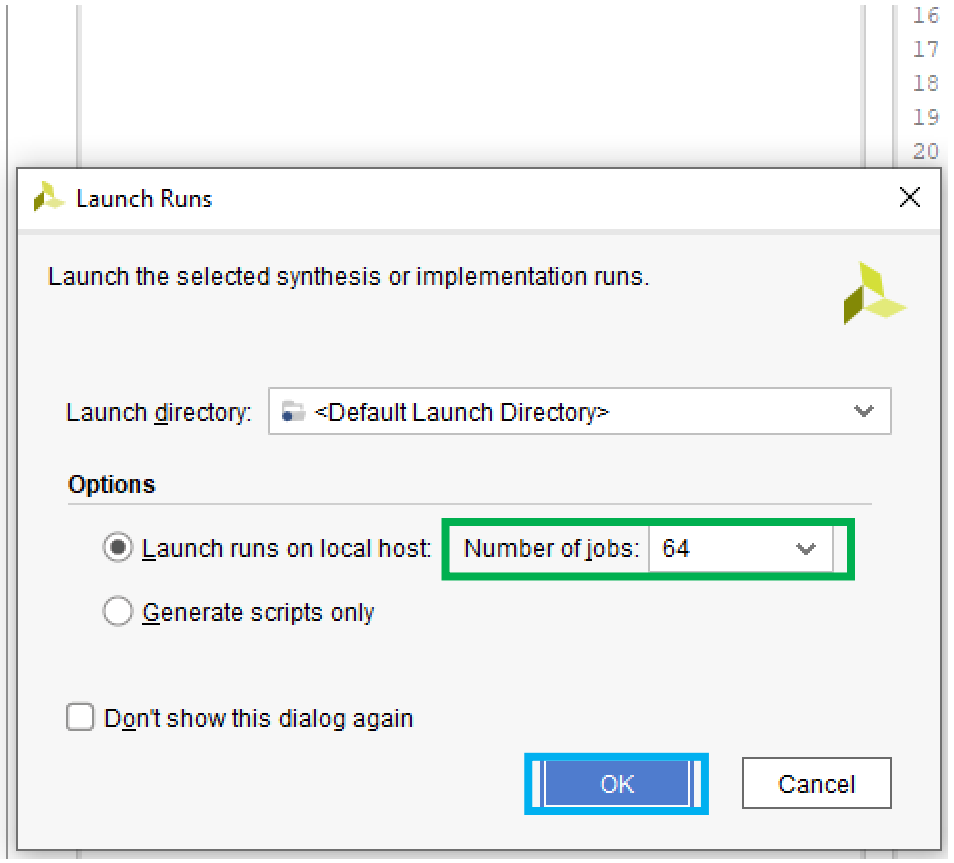
Graph | **Table**

Resource	Estimation	Available	Utilization %
LUT	1	63400	0.01
IO	3	210	1.43

Implementation

- Click on Run Implementation
- You can choose the number of jobs. The higher number results in faster implementation time. It is limited to the number of threads that your CPU has.
- Click on *OK* button




Launch Runs

Launch the selected synthesis or implementation runs.

Launch directory: <Default Launch Directory>

Options

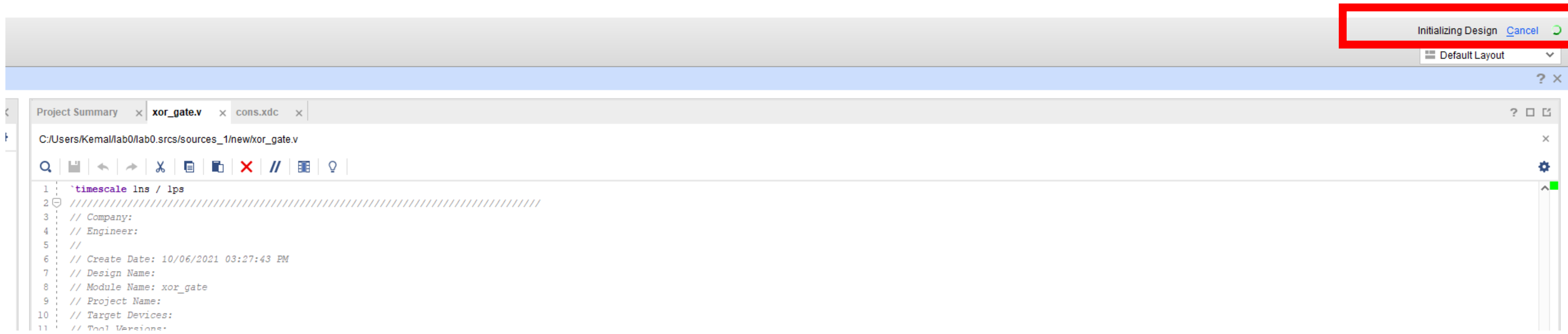
(●) Launch runs on local host: Number of jobs: 64

( ) Generate scripts only

☐ Don't show this dialog again

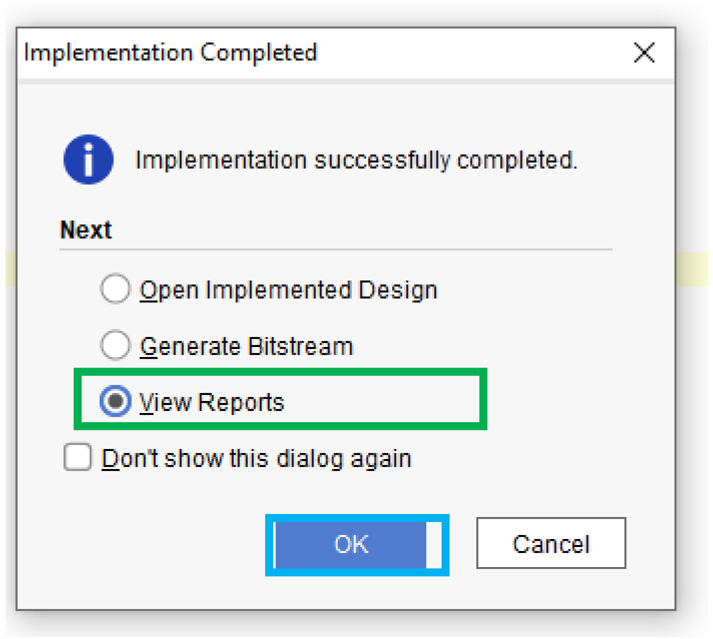
Implementation

- Implementation starts running.
- After it finishes, a window opens.
- Click on View Reports.
- Click on *OK* button



C:/Users/Kemal/lab0/lab0.srcs/sources_1/new/xor_gate.v

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 10/06/2021 03:27:43 PM
// Design Name:
// Module Name: xor_gate
// Project Name:
// Target Devices:
// Tool Versions:
```



Implementation Completed

Implementation successfully completed.

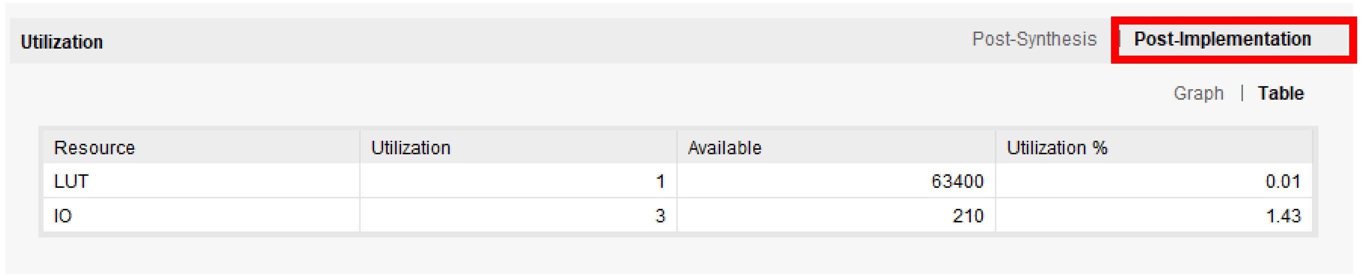
Next

- ( ) Open Implemented Design
- ( ) Generate Bitstream
- [✓ View Reports]

☐ Don't show this dialog again

Implementation

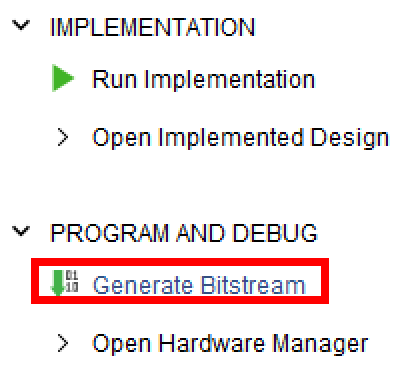
- Go to Project Summary
- Utilization table now shows the post-implementation values.

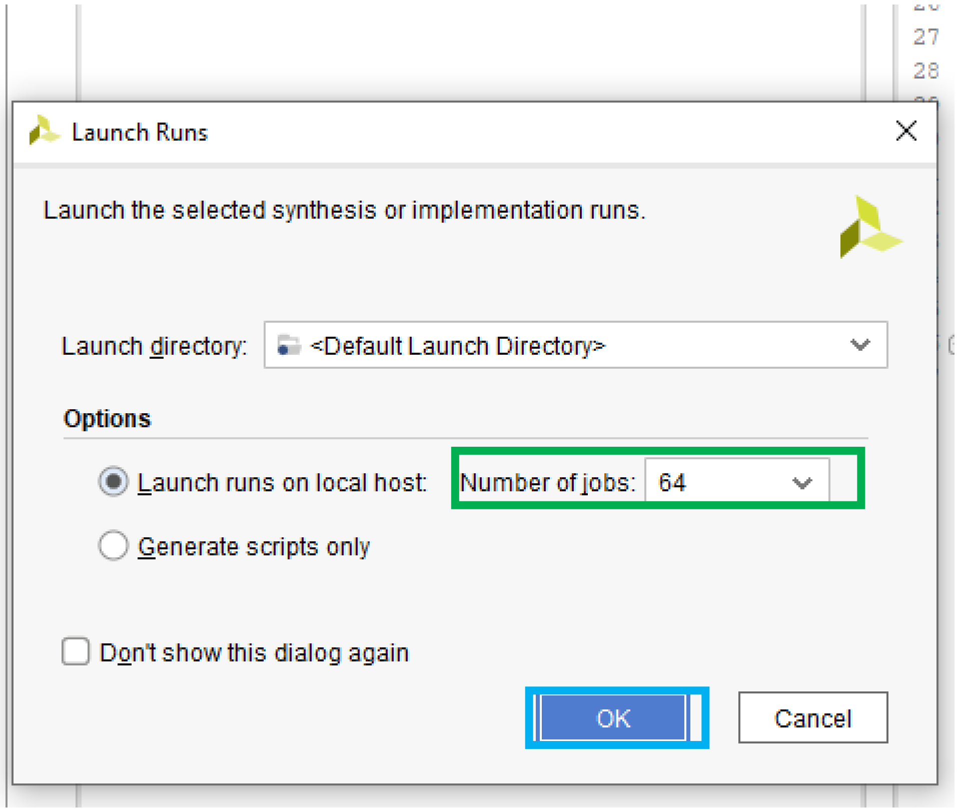


Resource	Utilization	Available	Utilization %
LUT	1	63400	0.01
IO	3	210	1.43

Generate a bit file

- Click on Generate Bitstream
- You can choose the number of jobs. The higher number results in faster generation time. It is limited to the number of threads that your CPU has.
- Click on *OK* button




Launch Runs

Launch the selected synthesis or implementation runs.

Launch directory: <Default Launch Directory>

Options

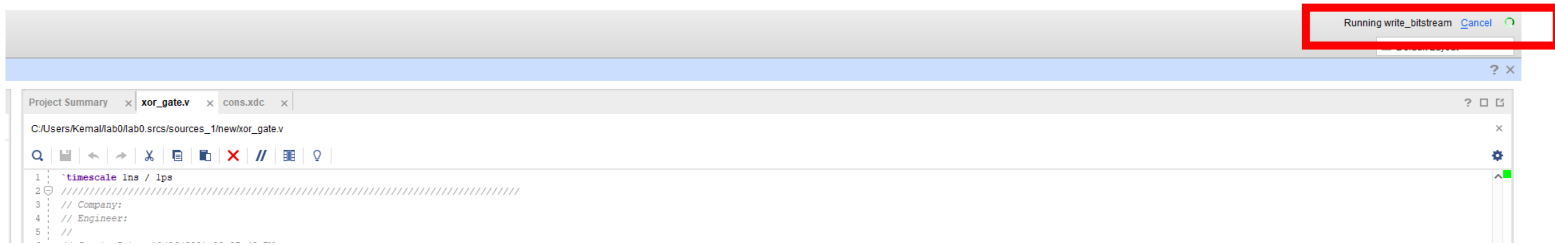
Launch runs on local host: [✓] Number of jobs: 64

Generate scripts only

Don't show this dialog again

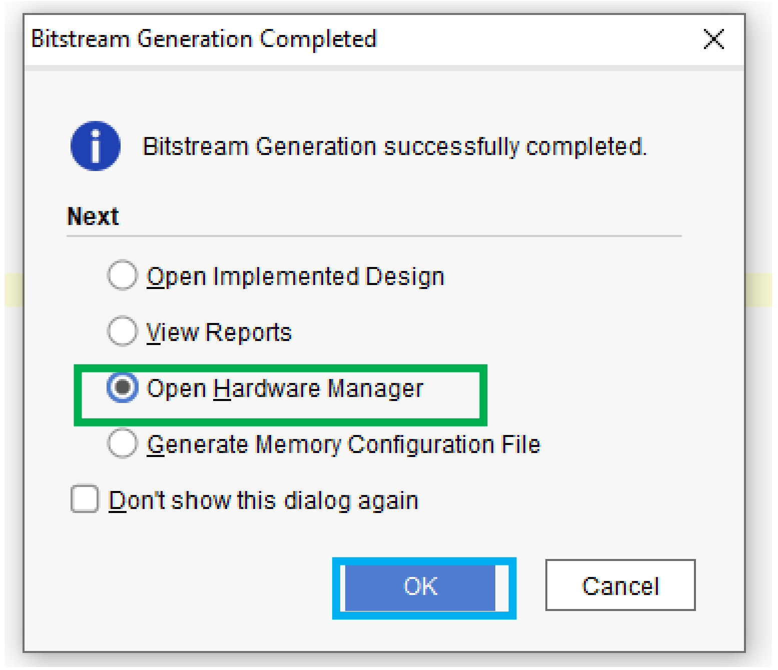
Generate a bit file

- Bitstream is generating.
- After it finishes, a window opens.
- Click on Open Hardware Manager.
- Click on *OK* button



C:/Users/Kemal/lab0/lab0.srcs/sources_1/new/xor_gate.v

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
```



Bitstream Generation Completed

Bitstream Generation successfully completed.

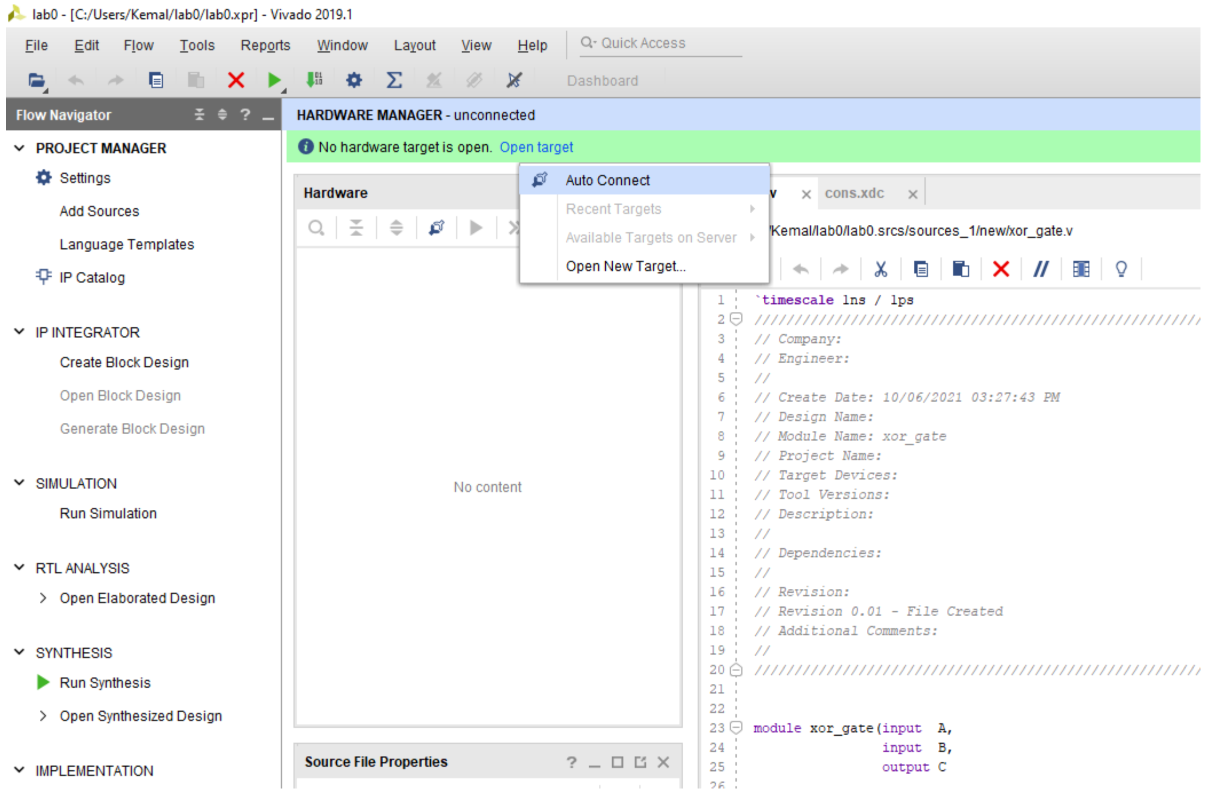
Next

- ○ Open Implemented Design
- ○ View Reports
- ● Open Hardware Manager
- ○ Generate Memory Configuration File

☐ Don't show this dialog again

Program the FPGA

- Click on Open Target
- Click on Auto Connect



HARDWARE MANAGER - unconnected

No hardware target is open. Open target

No content

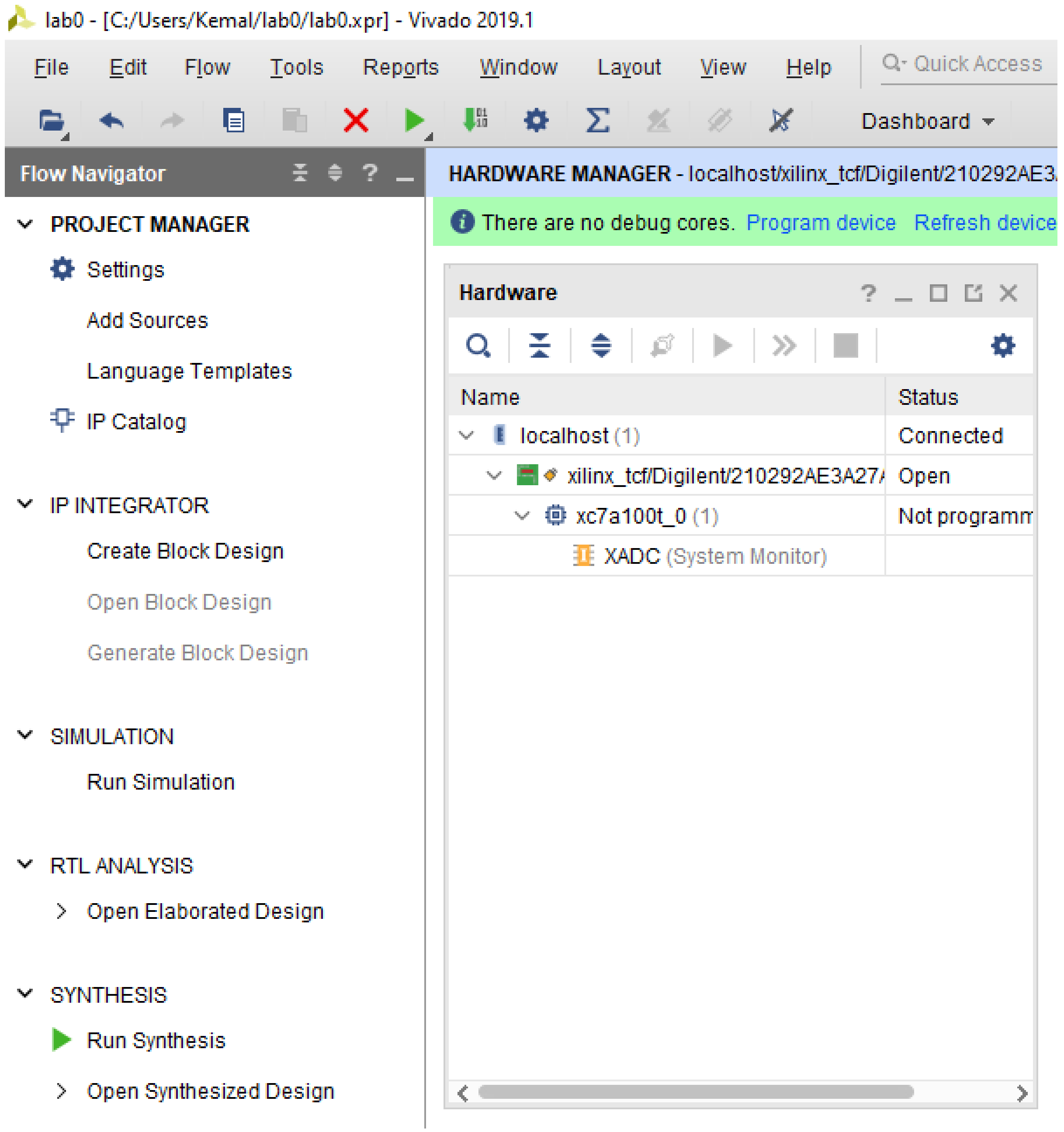
Kemal/lab0/lab0.srcs/sources_1/new/xor_gate.v

```
`timescale 1ns / 1ps
////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 10/06/2021 03:27:43 PM
// Design Name:
// Module Name: xor_gate
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////////////////////////////


module xor_gate(input  A,
                input  B,
                output C
```

Program the FPGA

- Vivado should find the connected FPGA to your computer.



lab0 - [C:/Users/Kemal/lab0/lab0.xpr] - Vivado 2019.1

File Edit Flow Tools Reports Window Layout View Help Quick Access

Dashboard

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

HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210292AE3

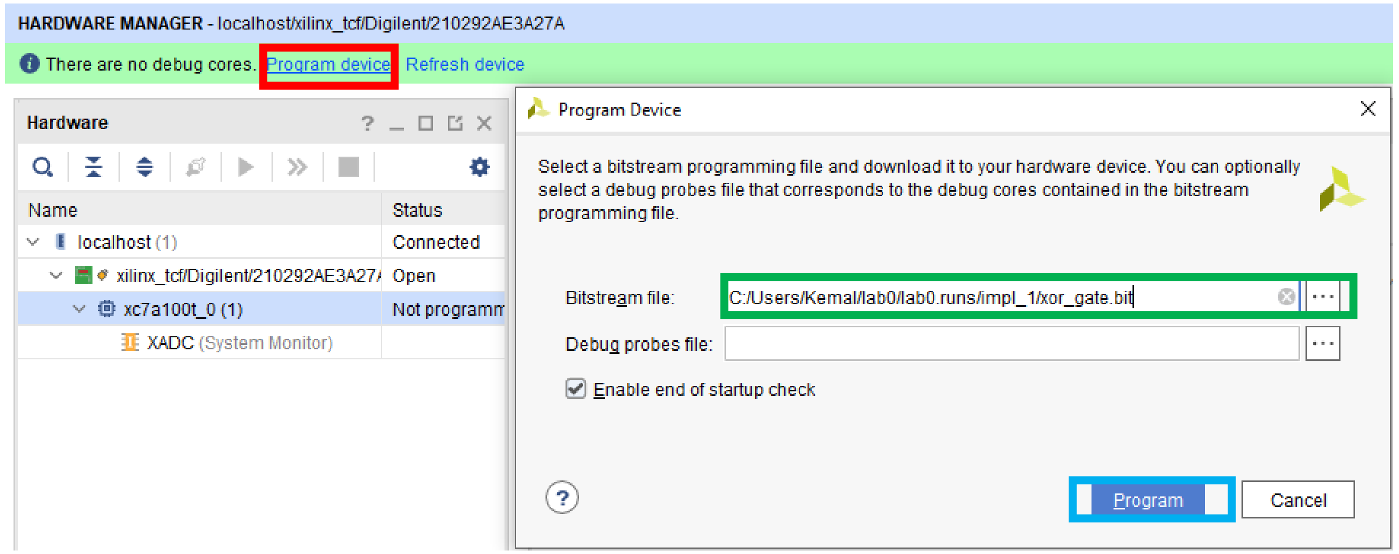
There are no debug cores. Program device Refresh device

Hardware

Name	Status
localhost (1)	Connected
xilinx_tcf/Digilent/210292AE3A27	Open
xc7a100t_0 (1)	Not programm
XADC (System Monitor)	

Program the FPGA

- Click on Program device.
- Generated bitstream file is chosen as default.
- Click on Program.



HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210292AE3A27A

There are no debug cores. Program device Refresh device

Hardware

Name	Status
localhost (1)	Connected
xilinx_tcf/Digilent/210292AE3A27/	Open
xc7a100t_0 (1)	Not programm
XADC (System Monitor)	

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.

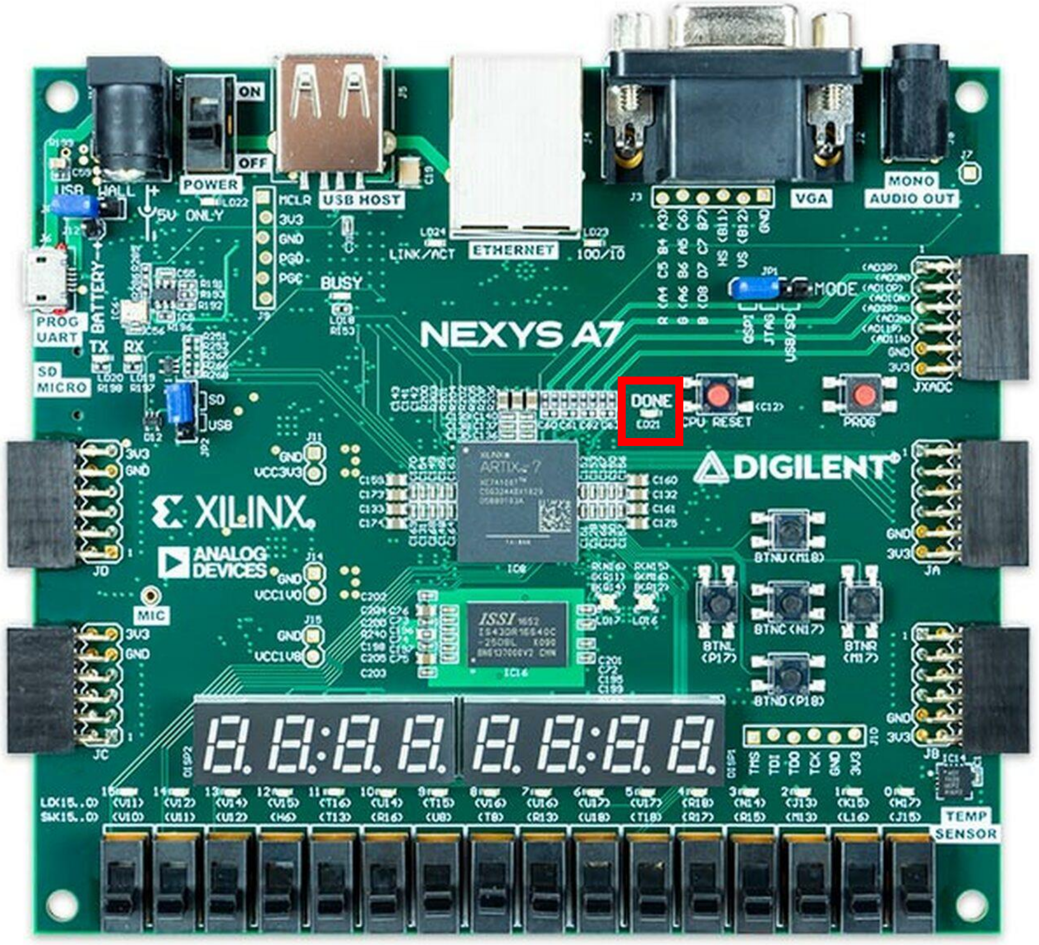
Bitstream file: C:/Users/Kemal/lab0/lab0.runs/impl_1/xor_gate.bit

Debug probes file:

☑ Enable end of startup check

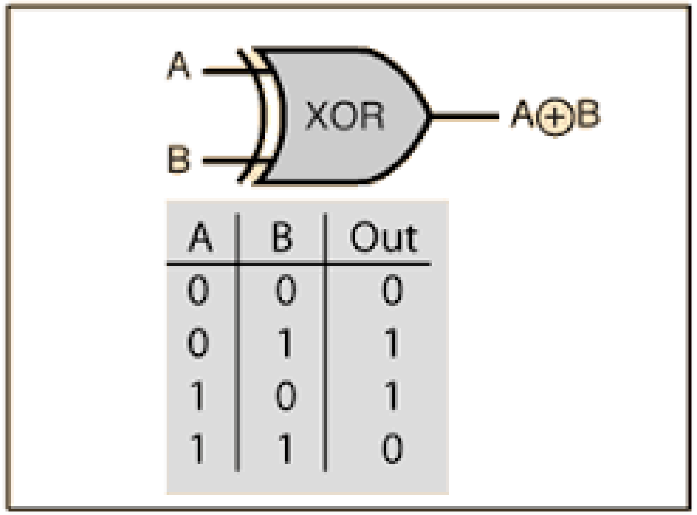
Program the FPGA

- Your design is realized on the FPGA.
- Check DONE LED on the board.



Using the FPGA

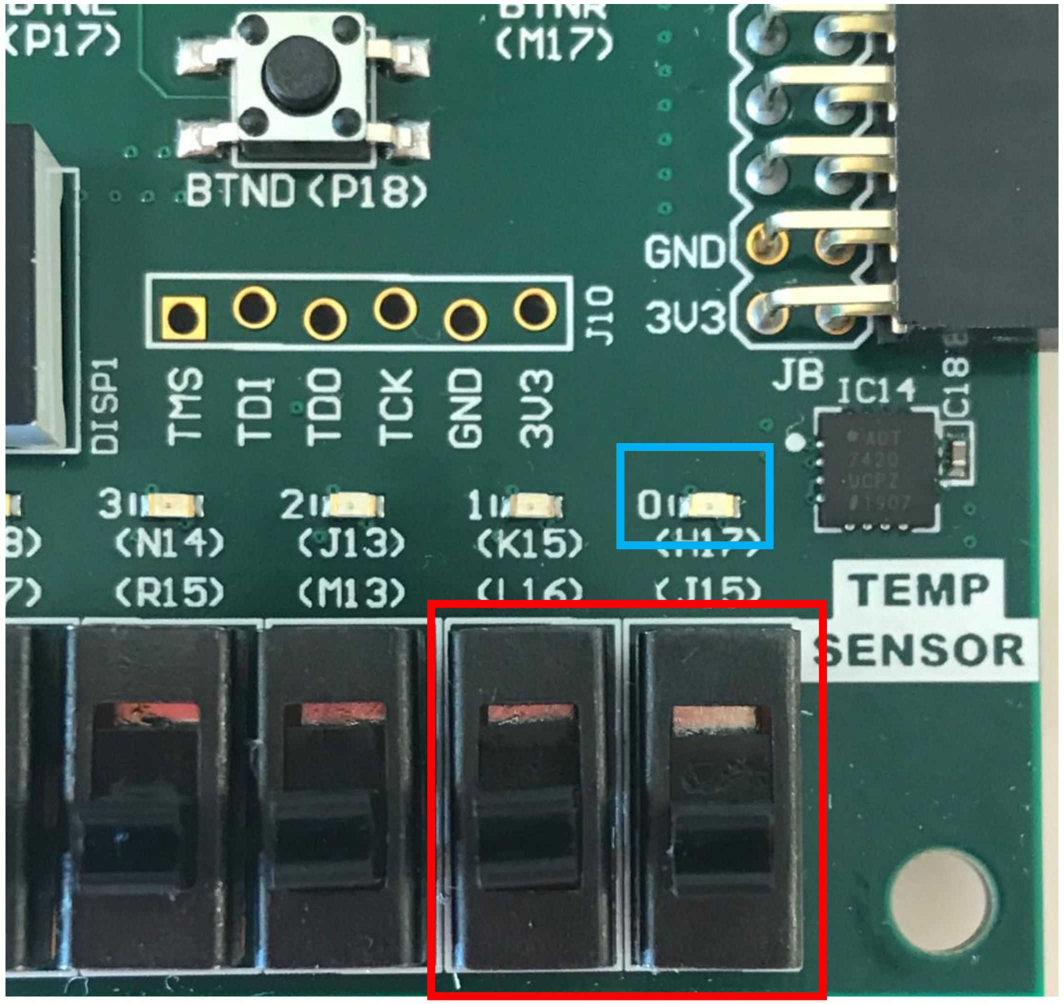
- You need to check your design on the FPGA by using switches.
- Observe the output LEDs.
- This sample design realizes XOR gate.
- The truth table of XOR gate is shown.



A	B	Out
0	0	0
0	1	1
1	0	1
1	1	0

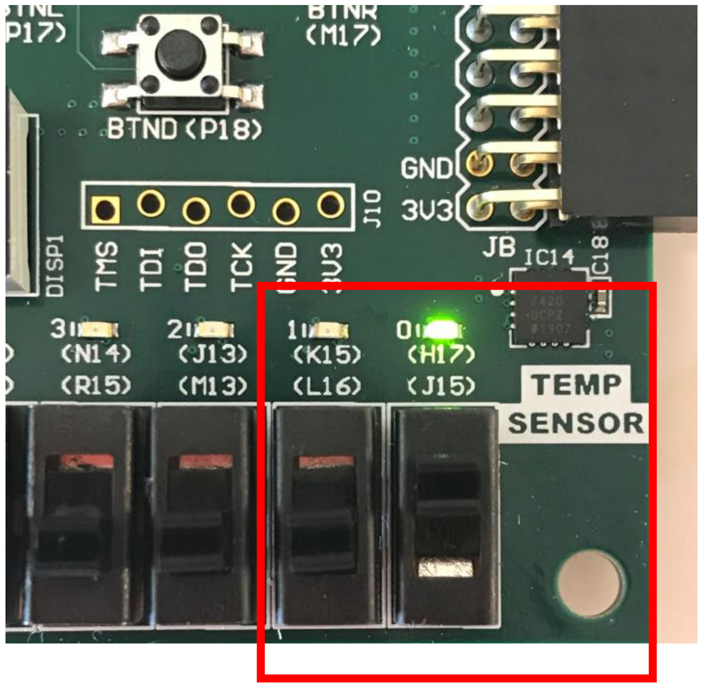
Using the FPGA

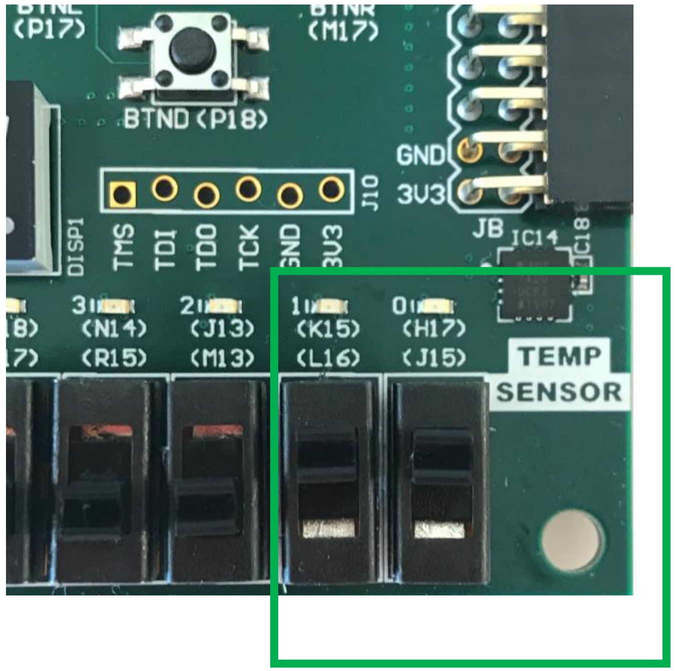
- Observe that when both switches(inputs A&B) are in Logic-0 position, LED 0 does not turn on.
- A, B => 0, Output => 0 (refer to the truth table of XOR gate)

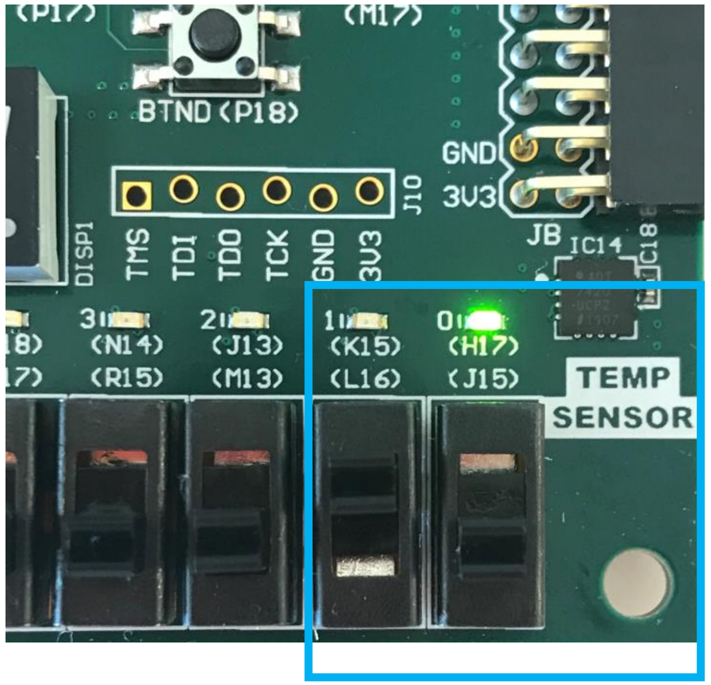


Using the FPGA

- A=>1, B=>0, Output => 1
- A=>0, B=>1, Output => 1
- A=>1, B=>1, Output => 0







- You need to observe the expected behavior of your design on the FPGA.
- It should be convenient with the simulation result of your design.
- The truth table of your design should match with the behavior of your design on the FPGA.