Lab 3. Block Design with IPs

In this lab, you will learn how to create a circuit in Vivado through block design by using intellectual properties (IPs), especially with the Xilinx University Program (XUP) Library.

Note:

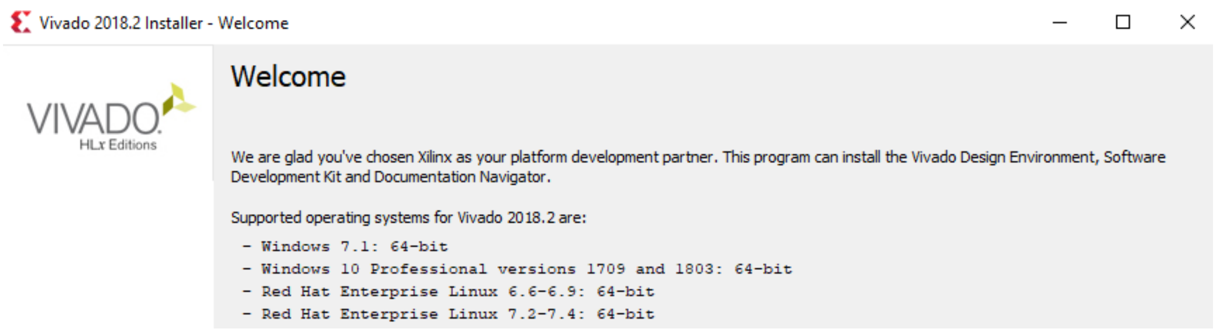
- If your machine already has Vivado and XUP library installed, you can jump to step 3.
- If you want to install Vivado on your own machine, you may follow steps 1 and 2.

Step 1. Download Vivado Design Suite WebPack Edition from https://www.xilinx.com/support/download.html.

Vivado HLx 2019.1 Update 2 (TAR/GZIP - 9.59 GB)

MD5 SUM Value : ebfeb5dabedf5292a2751acb2227dbda

When you click on the edition above, Xilinx will ask you to create an account. After creating your account, click next button, then you can start the self-installer file.



Welcome

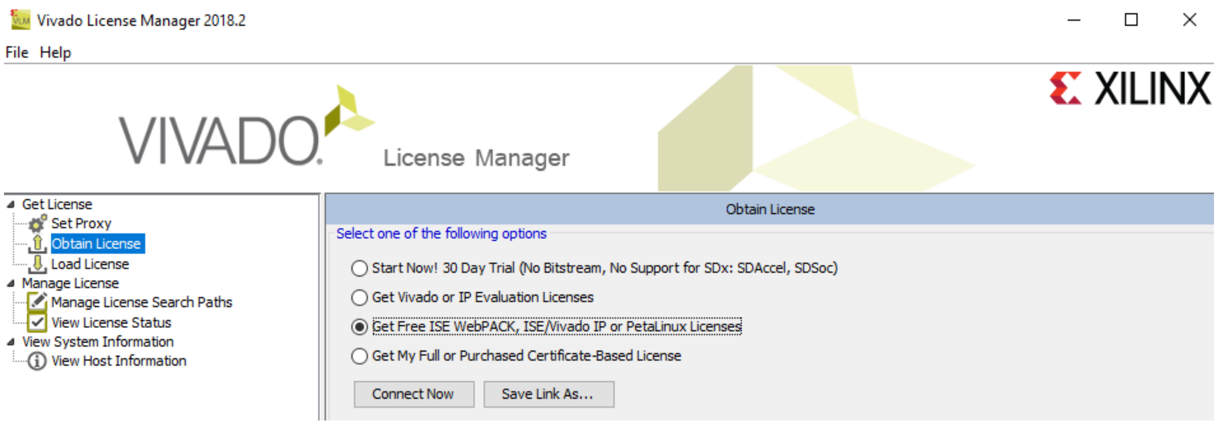
We are glad you've chosen Xilinx as your platform development partner. This program can install the Vivado Design Environment, Software Development Kit and Documentation Navigator.

Supported operating systems for Vivado 2018.2 are:

```
 - Windows 7.1: 64-bit
 - Windows 10 Professional versions 1709 and 1803: 64-bit
 - Red Hat Enterprise Linux 6.6-6.9: 64-bit
 - Red Hat Enterprise Linux 7.2-7.4: 64-bit
```

(The figure above is the screenshot from the 2018.2 version.)

Make sure that you have a required operating system. Next, it will ask you to log in with your account to download the program. To complete the download, you have to activate the license.



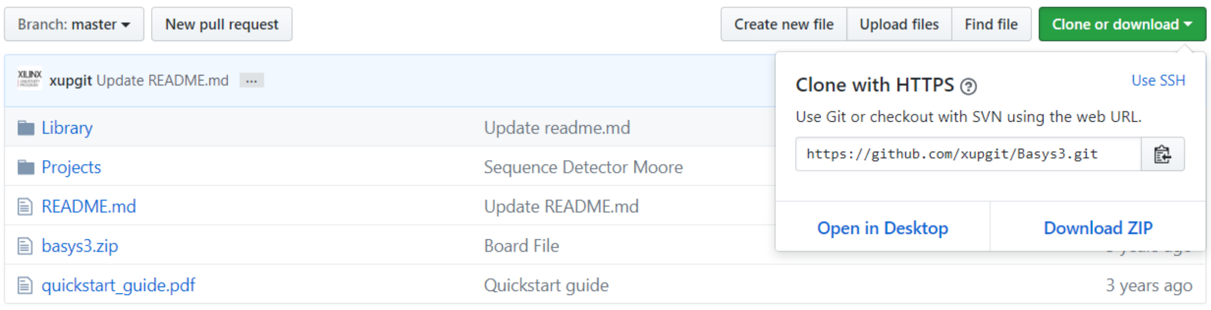
Select one of the following options

- ○ Start Now! 30 Day Trial (No Bitstream, No Support for SDx: SDAccel, SDSoc)
- ○ Get Vivado or IP Evaluation Licenses
- ● Get Free ISE WebPACK, ISE/Vivado IP or PetaLinux Licenses
- ○ Get My Full or Purchased Certificate-Based License

For this lab assignment, you can use the program installed in the lab computers. However, we strongly recommend you to download the program on your own computer to practice and for the future labs.

Step 2. Obtain the XUP library.

Visit https://github.com/xupgit/Basys3. Log in to GitHub with your account (you will have to create one if you do not have one). Click on clone or download.


Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

xupgit Update README.md

Library	Update readme.md	
Projects	Sequence Detector Moore	
README.md	Update README.md	
basys3.zip	Board File	
quickstart_guide.pdf	Quickstart guide	3 years ago

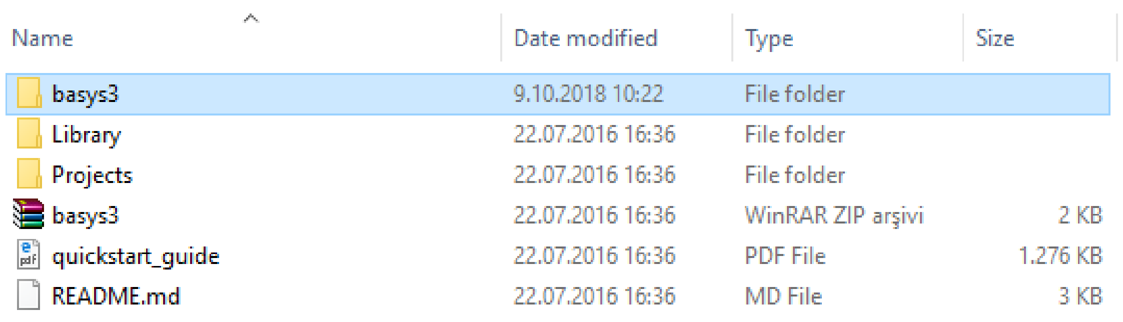
Clone with HTTPS | Use SSH

Use Git or checkout with SVN using the web URL.

```
https://github.com/xupgit/Basys3.git
```

Open in Desktop | Download ZIP

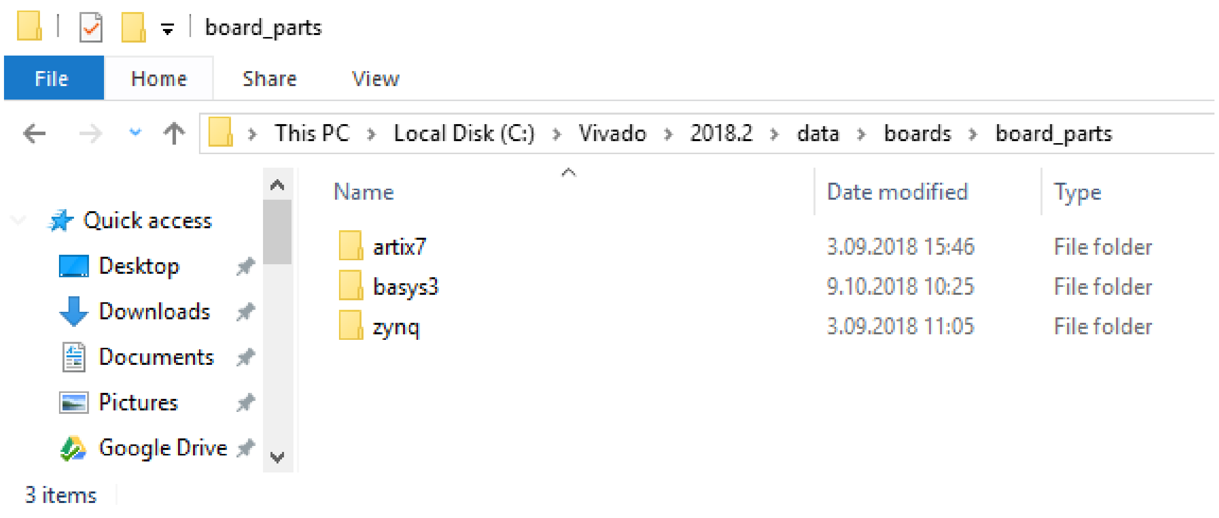
Extract the zip files.



Name	Date modified	Type	Size
basys3	9.10.2018 10:22	File folder	
Library	22.07.2016 16:36	File folder	
Projects	22.07.2016 16:36	File folder	
basys3	22.07.2016 16:36	WinRAR ZIP arşivi	2 KB
quickstart_guide	22.07.2016 16:36	PDF File	1.276 KB
README.md	22.07.2016 16:36	MD File	3 KB

Now copy the basys3 folder to C:\Vivado\2019.1\data\boards\board_parts

(Note: '2019.1' part can be different depending on the version of the program you installed.)


board_parts

File | Home | Share | View

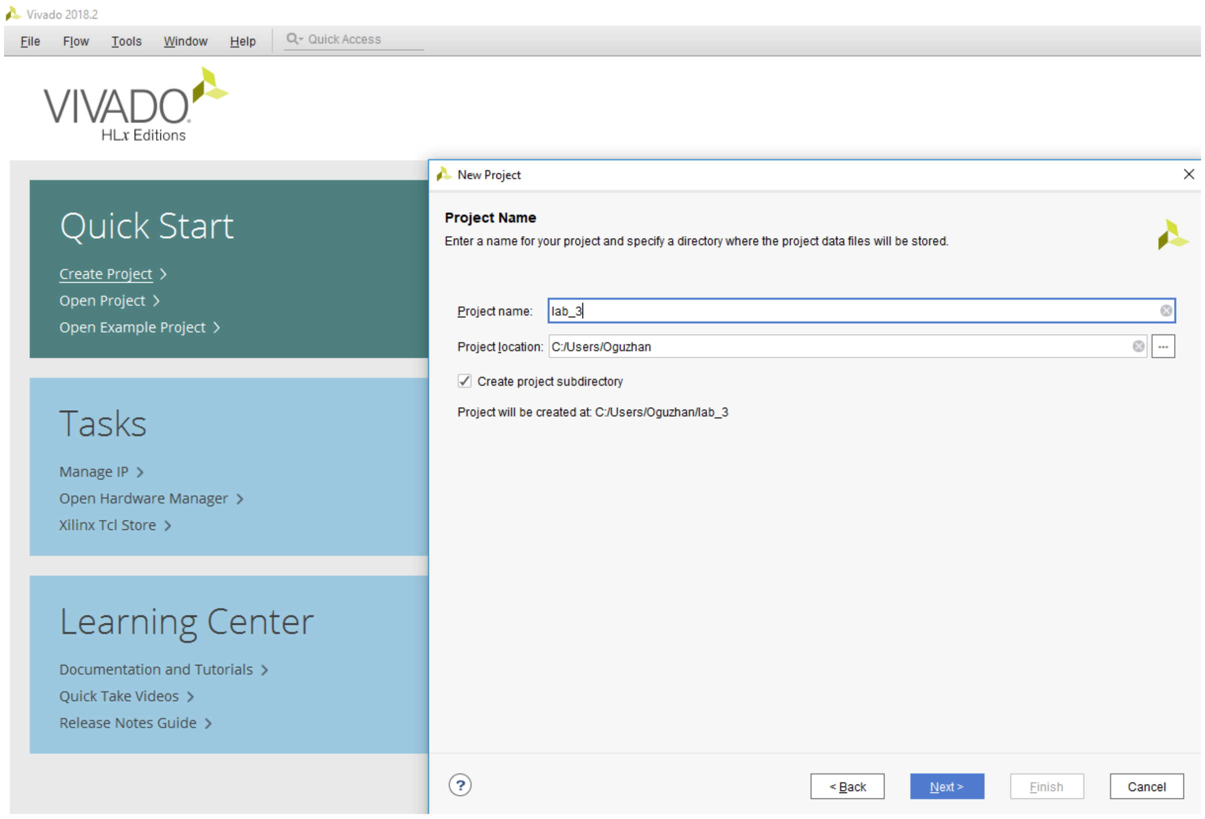
This PC › Local Disk (C:) › Vivado › 2018.2 › data › boards › board_parts

Quick access
Desktop
Downloads
Documents
Pictures
Google Drive

Name	Date modified	Type
artix7	3.09.2018 15:46	File folder
basys3	9.10.2018 10:25	File folder
zynq	3.09.2018 11:05	File folder

3 items

Step 3. Open the Vivado program and create a new project. Choose RTL project.



Project Name

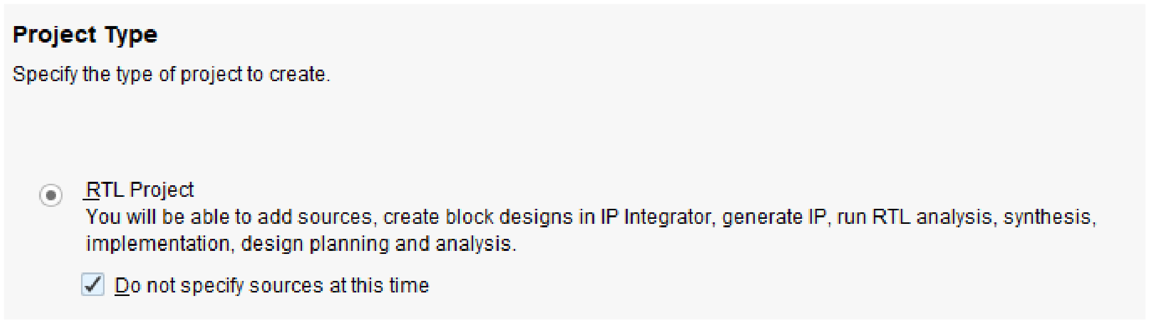
Enter a name for your project and specify a directory where the project data files will be stored.

Project name: lab_3

Project location: C:/Users/Oguzhan

✓ Create project subdirectory

Project will be created at: C:/Users/Oguzhan/lab_3



Project Type

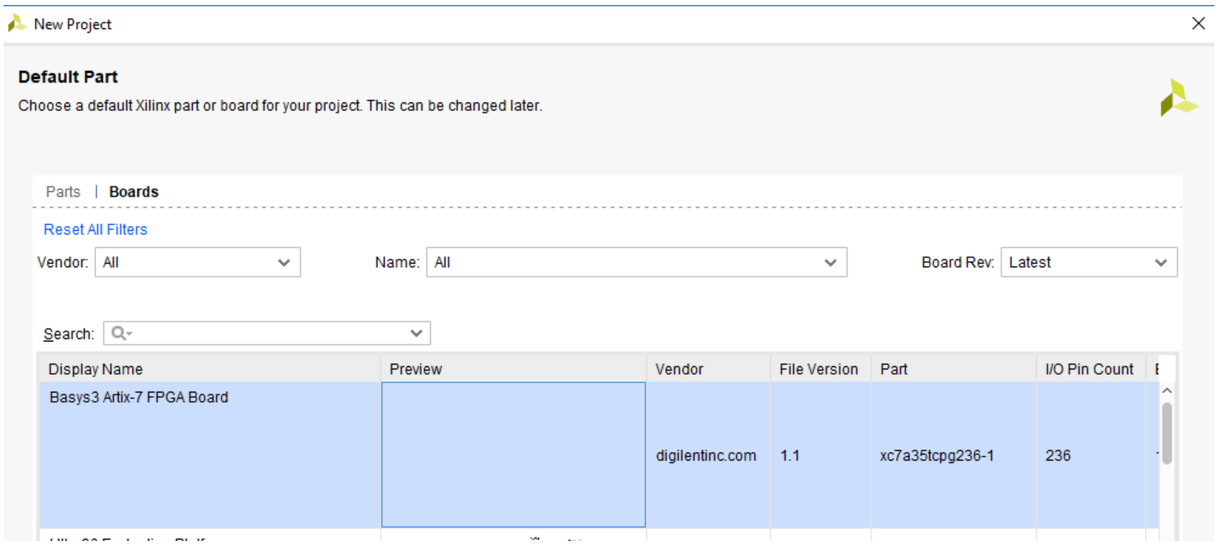
Specify the type of project to create.

● 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

Next, click on boards and choose Basys3.



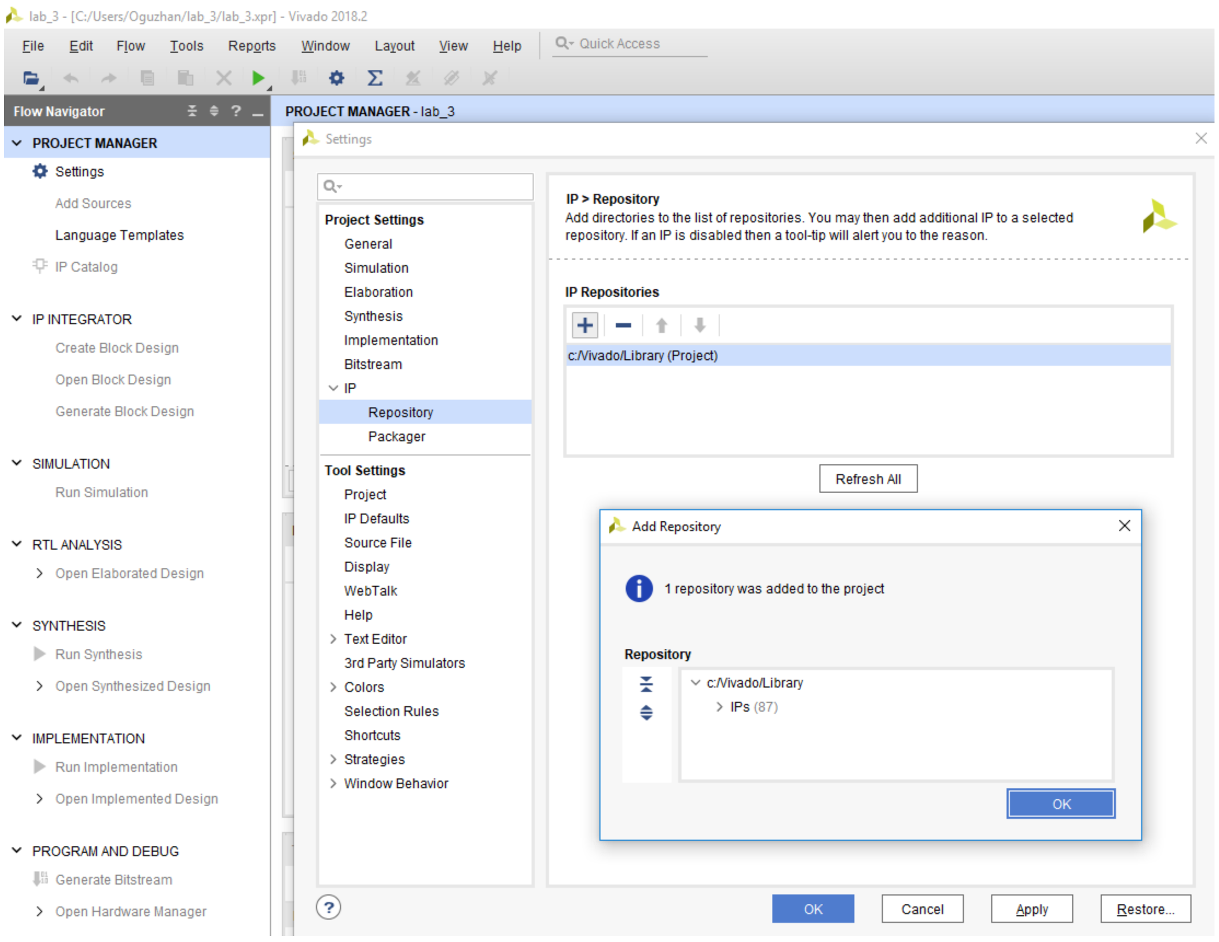
Default Part

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

Vendor: All Name: All Board Rev: Latest

Display Name	Preview	Vendor	File Version	Part	I/O Pin Count
Basys3 Artix-7 FPGA Board		digilentinc.com	1.1	xc7a35tcpg236-1	236

Step 4. Under the Project Manager, open the Project settings. Find IP and open Repository. Click on add (+) button and choose Library obtained from GitHub.

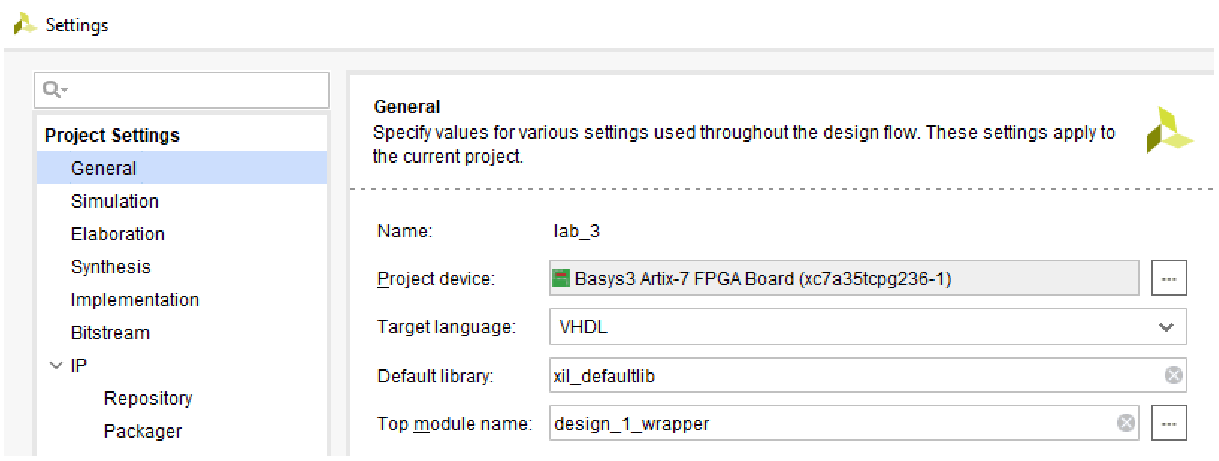

lab_3 - [C:/Users/Oguzhan/lab_3/lab_3.xpr] - Vivado 2018.2

IP > Repository
Add directories to the list of repositories. You may then add additional IP to a selected repository. If an IP is disabled then a tool-tip will alert you to the reason.

IP Repositories
c:/Vivado/Library (Project)

Add Repository
1 repository was added to the project
Repository
c:/Vivado/Library
IPs (87)

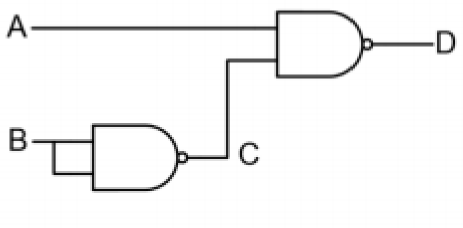
Make sure that in the General setting, language is defined as VHDL, not Verilog.


Settings

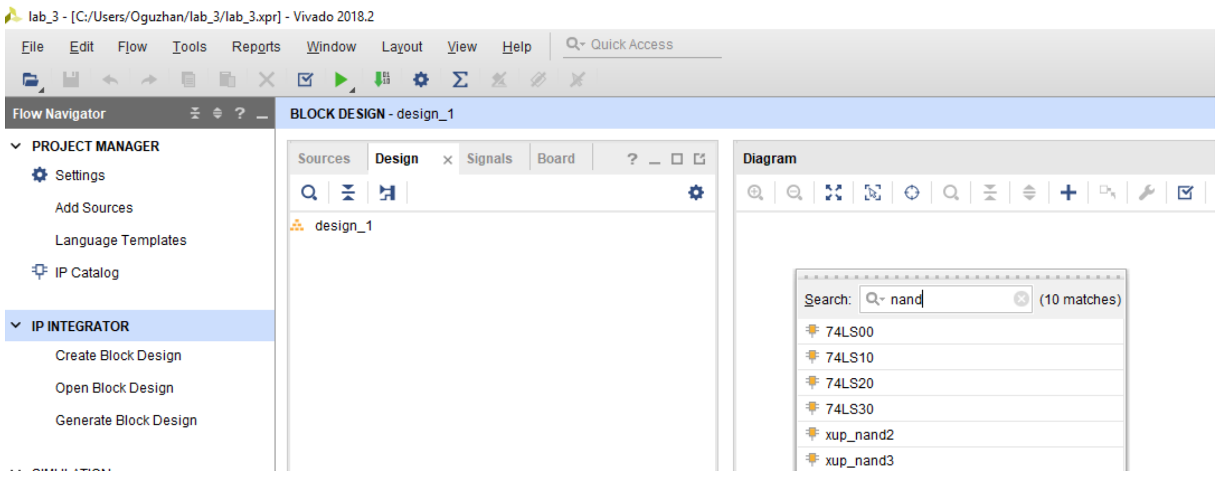
General
Specify values for various settings used throughout the design flow. These settings apply to the current project.

Name:	lab_3
Project device:	Basys3 Artix-7 FPGA Board (xc7a35tcpg236-1)
Target language:	VHDL
Default library:	xil_defaultlib
Top module name:	design_1_wrapper

Step 5. Now we are ready to create a circuit. Click on Create Block Design. Continue with default options. Your assignment is to create a circuit below using Vivado and implement it via a Basys3 board.

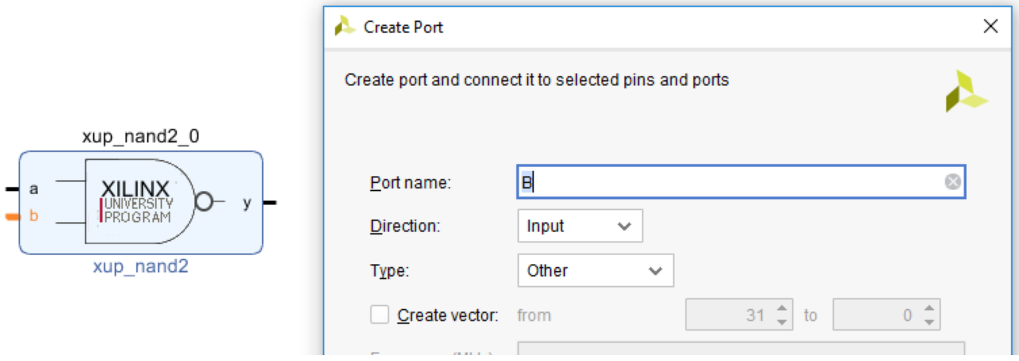


From the Diagram window, press the add (+) button to add an IP. Search for NAND2. Choose xup_nand2. Right click over the gate and choose Create Port. Change the port name to 'B'.


lab_3 - [C:/Users/Oguzhan/lab_3/lab_3.xpr] - Vivado 2018.2

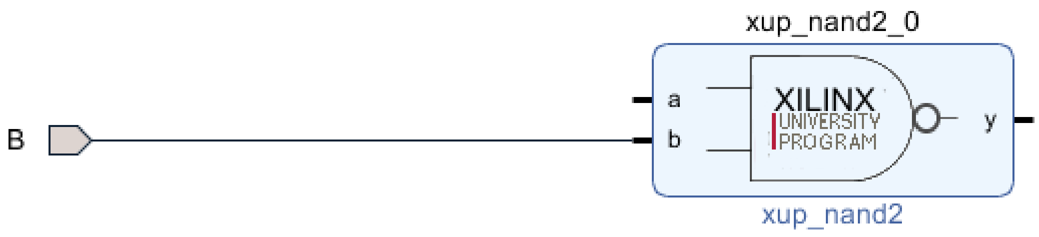
Search: nand (10 matches)

74LS00
74LS10
74LS20
74LS30
xup_nand2
xup_nand3

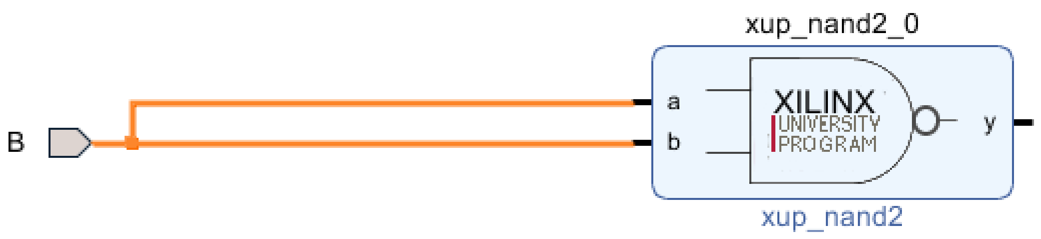


Create port and connect it to selected pins and ports

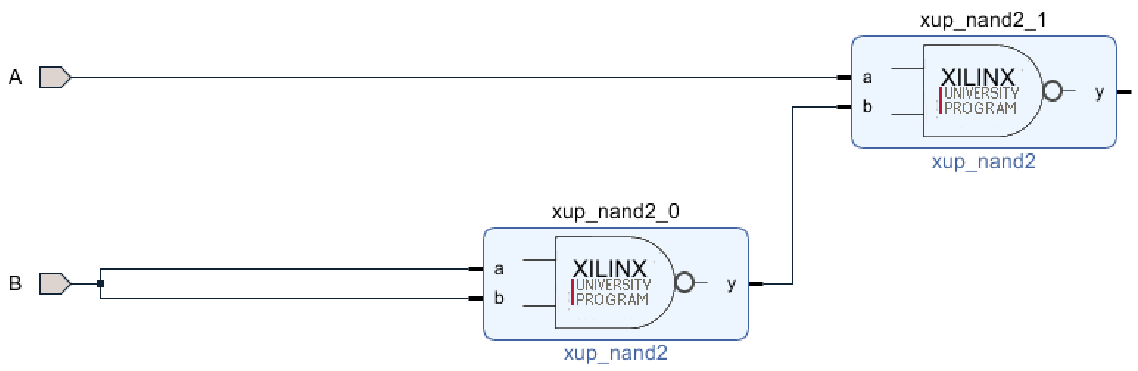
Port name: B
Direction: Input
Type: Other
Create vector: from 31 to 0



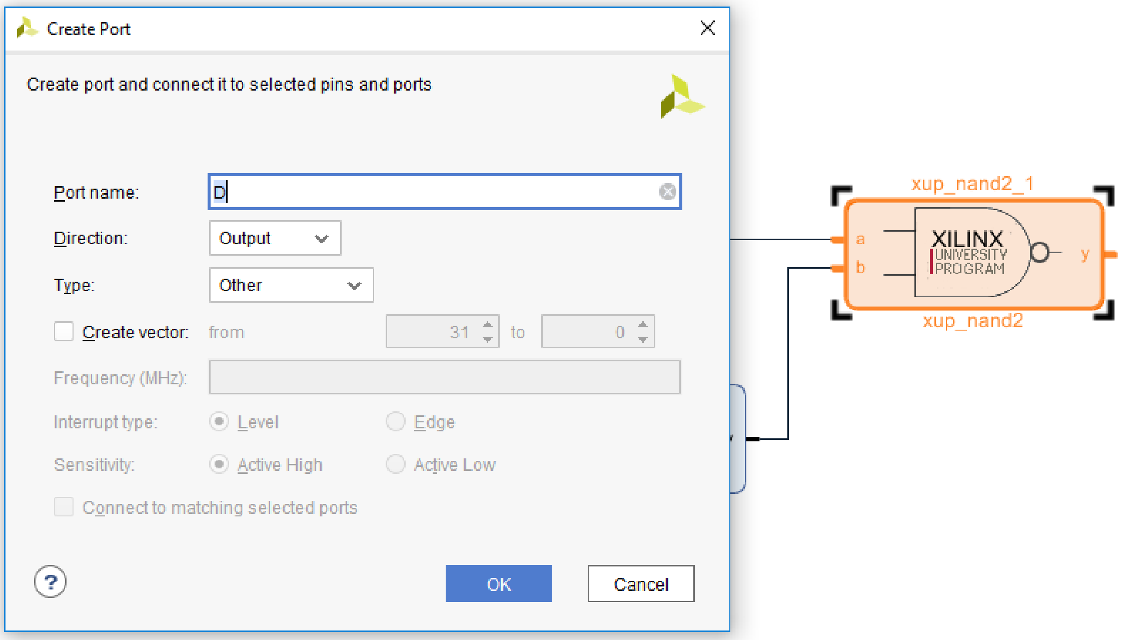
When a mouse pointer comes close to input, a pen will appear. Use the pen to draw connections as below.



Add one more 2-input NAND gate and create a port A. Connect the output of the first NAND2 to one of the inputs of the second NAND2.



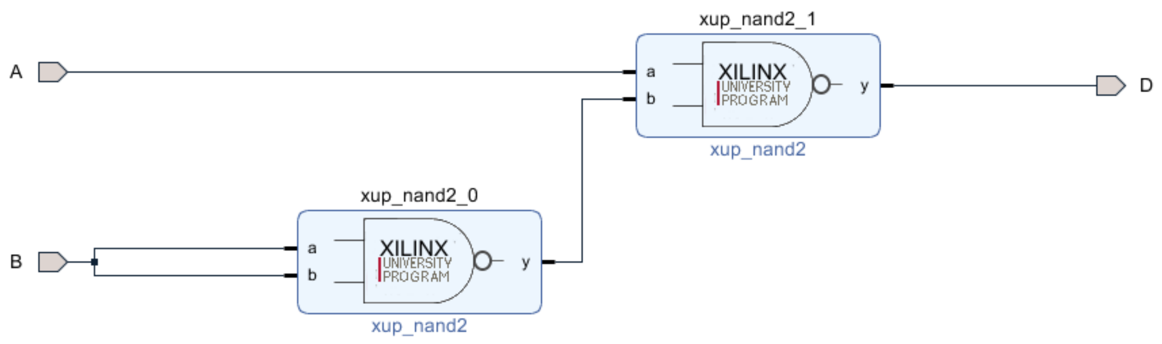
Create a circuit output port. Right click on the second NAND2 and choose Create Port. Give the port name as ‘D’. Choose the direction as Output.


Create Port

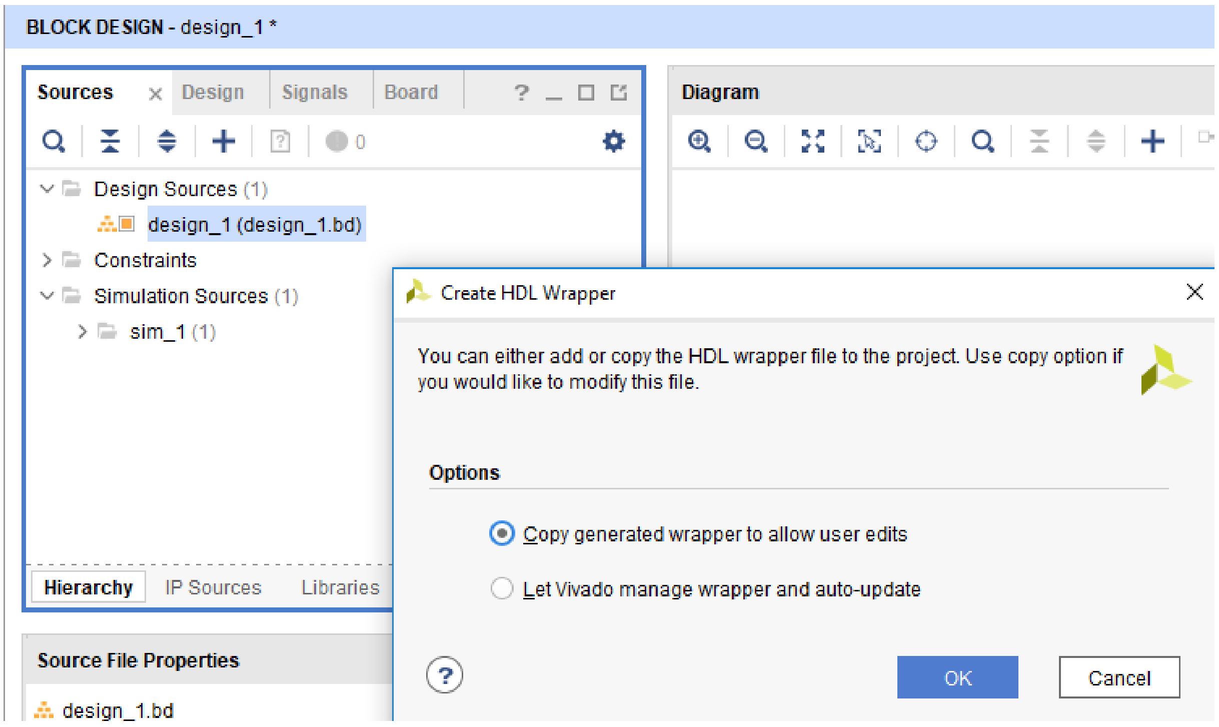
Create port and connect it to selected pins and ports

Port name:	D
Direction:	Output
Type:	Other
Create vector:	from 31 to 0
Frequency (MHz):	
Interrupt type:	[● Level] Edge
Sensitivity:	[● Active High] Active Low
Connect to matching selected ports	

Now you should have completed the circuit as below.



Step 6. Now we are ready to observe the VHDL code of this design. In Sources tab, right click on your design and choose Create HDL Wrapper. Now, you can see the VHDL code of the design.



Create HDL Wrapper

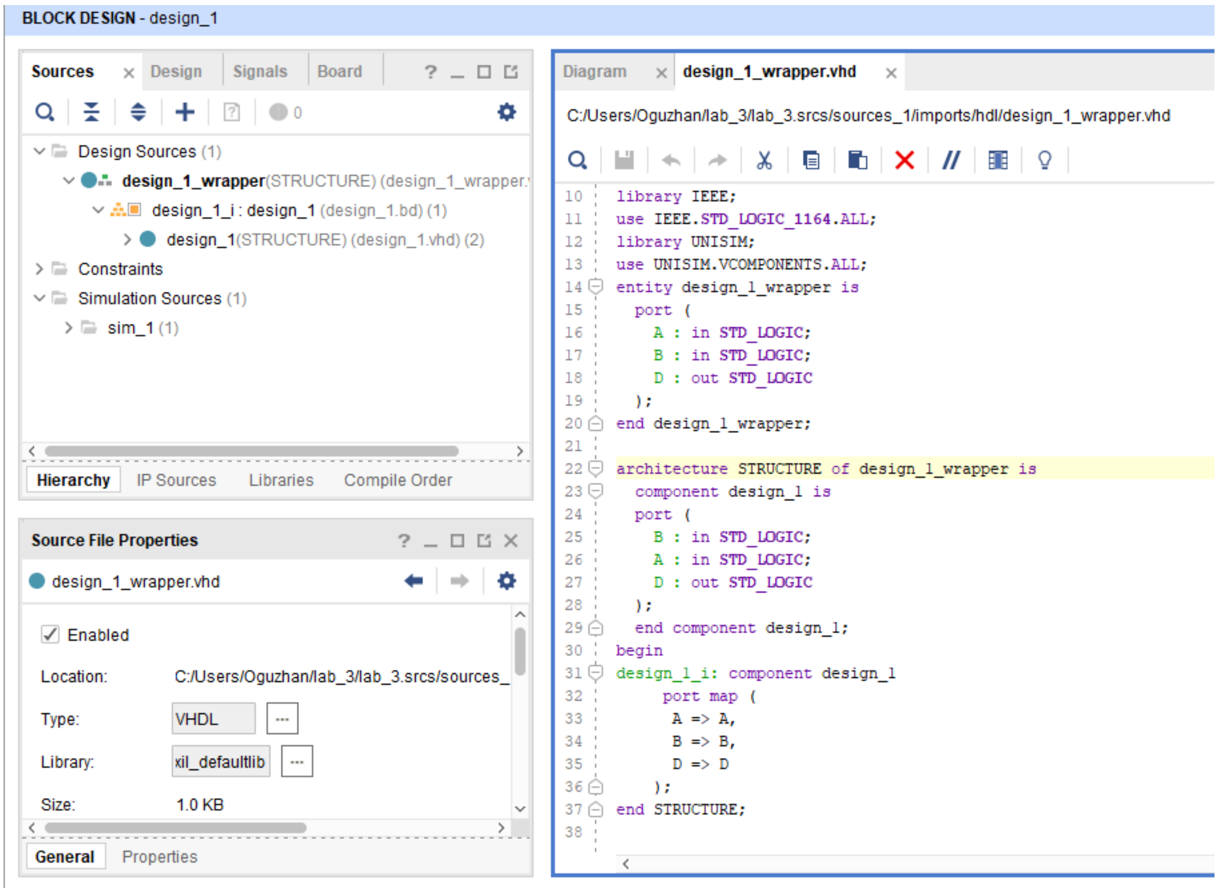
You can either add or copy the HDL wrapper file to the project. Use copy option if you would like to modify this file.

Options

[✓ Copy generated wrapper to allow user edits]

Let Vivado manage wrapper and auto-update

OK Cancel



BLOCK DESIGN - design_1

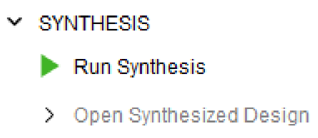
- [✓] Enabled
- Location: C:/Users/Oguzhan/lab_3/lab_3.srcs/sources_
- Type: VHDL
- Library: xil_defaultlib
- Size: 1.0 KB

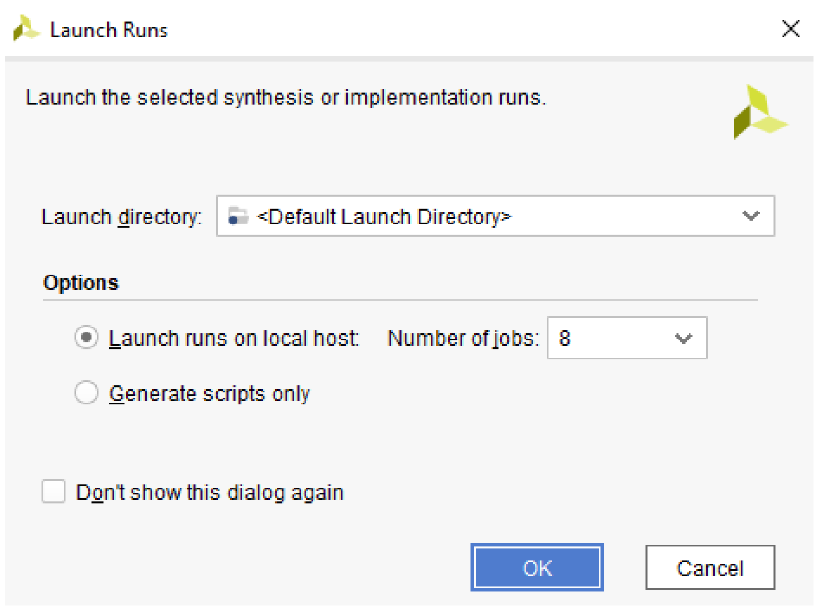
C:/Users/Oguzhan/lab_3/lab_3.srcs/sources_1/imports/hdl/design_1_wrapper.vhd

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
library UNISIM;
use UNISIM.VCOMPONENTS.ALL;
entity design_1_wrapper is
  port (
    A : in STD_LOGIC;
    B : in STD_LOGIC;
    D : out STD_LOGIC
  );
end design_1_wrapper;

architecture STRUCTURE of design_1_wrapper is
  component design_1 is
  port (
    B : in STD_LOGIC;
    A : in STD_LOGIC;
    D : out STD_LOGIC
  );
  end component design_1;
begin
design_1_i: component design_1
     port map (
      A => A,
      B => B,
      D => D
    );
end STRUCTURE;
```

Step 7. After creating a VHDL code, there are four steps to program a Basys3 board. First step is to run synthesis. Synthesis is the process of transforming an RTL-specified design into a gate-level representation.





Launch Runs

Launch the selected synthesis or implementation runs.

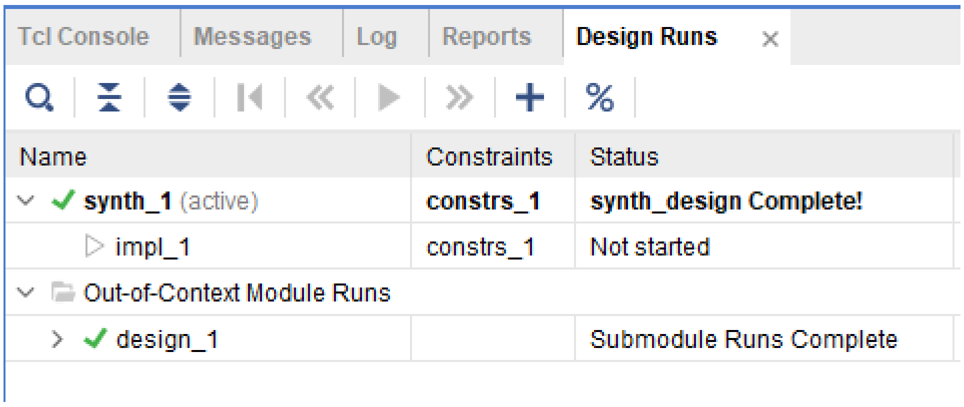
Launch directory: <Default Launch Directory>

Options

- [✓] Launch runs on local host: Number of jobs: 8
- Generate scripts only

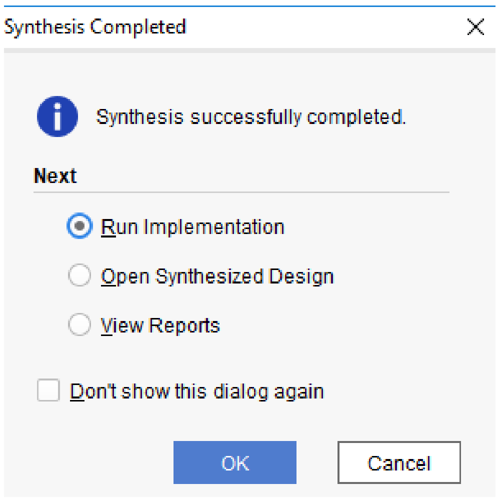
Don't show this dialog again

You can follow the progress from the Design Runs tab.



Name	Constraints	Status
✓ **synth_1** (active)	**constrs_1**	**synth_design Complete!**
impl_1	constrs_1	Not started
Out-of-Context Module Runs		
✓ design_1		Submodule Runs Complete

After synthesis is completed, a new window will pop up.



Synthesis Completed

Synthesis successfully completed.

Next

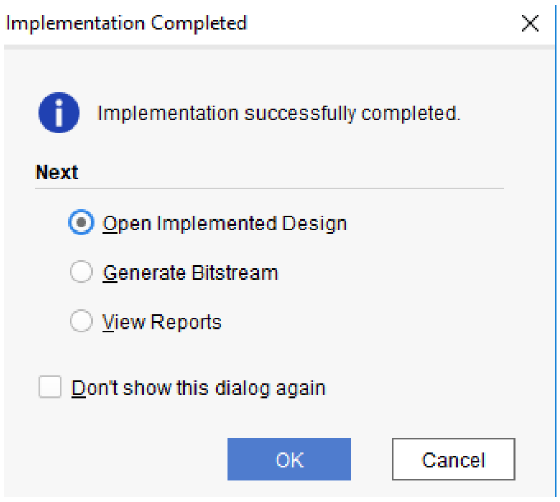
[✓ Run Implementation]

Open Synthesized Design

View Reports

Don't show this dialog again

Step 8. Run the implementation. Implementation process transforms a logical netlist and constraints into a placed and routed design, ready for bitstream generation. Once the implementation is completed, you can open the implemented design to observe how the design is implemented inside the FPGA.



Implementation Completed

Implementation successfully completed.

Next

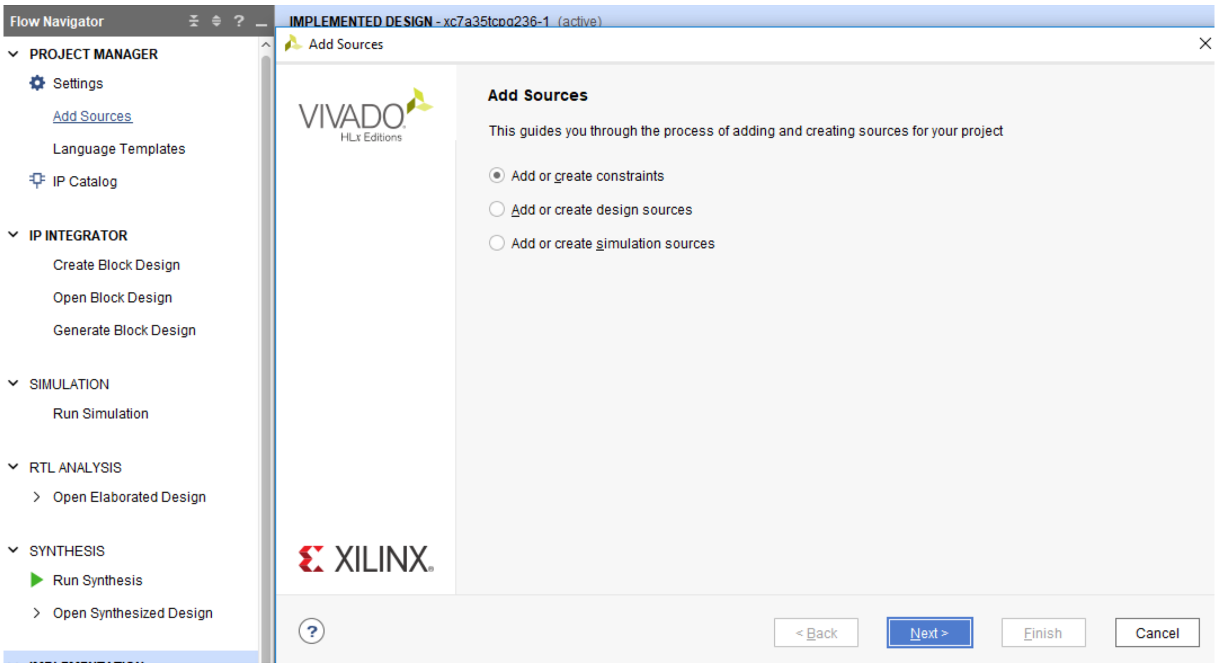
[✓ Open Implemented Design]

Generate Bitstream

View Reports

Don't show this dialog again

Step 9. Click on add sources and then choose add or create constraints.

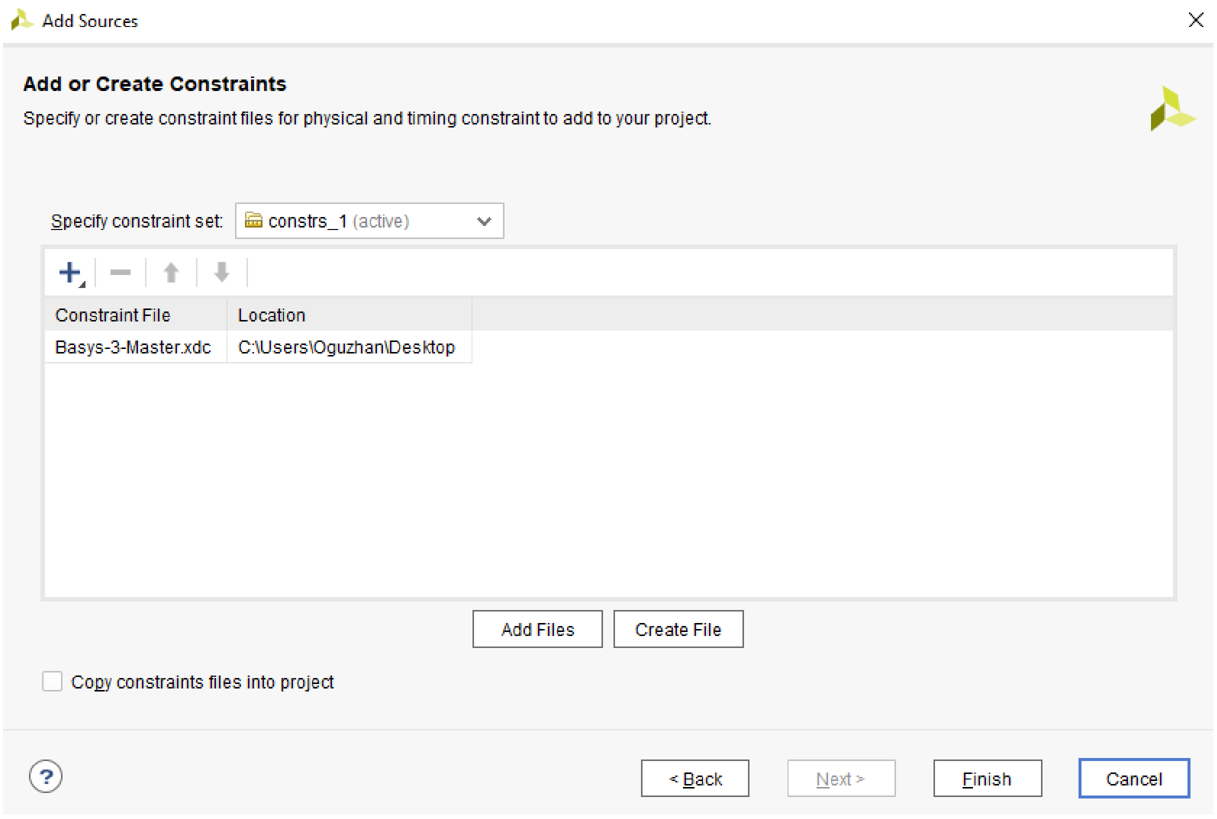


Add Sources

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

Download Basys3.xdc file from Canvas and add the file as a constraint file.



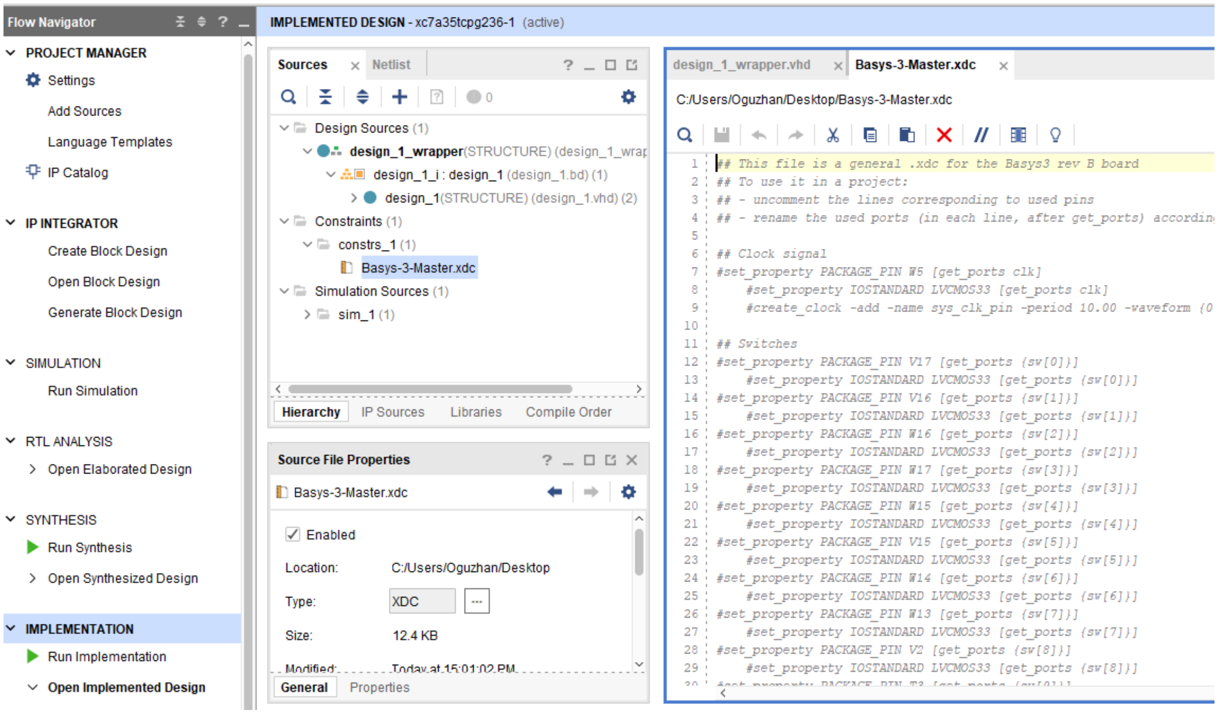
Add or Create Constraints

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

Specify constraint set: constrs_1 (active)

Constraint File	Location
Basys-3-Master.xdc	C:\Users\Oguzhan\Desktop

Copy constraints files into project



IMPLEMENTED DESIGN - xc7a35tcpg236-1 (active)

Basys-3-Master.xdc

✓ Enabled

Location: C:/Users/Oguzhan/Desktop

Type: XDC

Size: 12.4 KB

Modified: Today at 15:01:02 PM

C:/Users/Oguzhan/Desktop/Basys-3-Master.xdc

```
## This file is a general .xdc for the Basys3 rev B board
## To use it in a project:
## - uncomment the lines corresponding to used pins
## - rename the used ports (in each line, after get_ports) accordin

## Clock signal
#set_property PACKAGE_PIN W5 [get_ports clk]
    #set_property IOSTANDARD LVCMOS33 [get_ports clk]
    #create_clock -add -name sys_clk_pin -period 10.00 -waveform {0

## Switches
#set_property PACKAGE_PIN V17 [get_ports {sw[0]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {sw[0]}]
#set_property PACKAGE_PIN V16 [get_ports {sw[1]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {sw[1]}]
#set_property PACKAGE_PIN W16 [get_ports {sw[2]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {sw[2]}]
#set_property PACKAGE_PIN W17 [get_ports {sw[3]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {sw[3]}]
#set_property PACKAGE_PIN W15 [get_ports {sw[4]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {sw[4]}]
#set_property PACKAGE_PIN V15 [get_ports {sw[5]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {sw[5]}]
#set_property PACKAGE_PIN W14 [get_ports {sw[6]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {sw[6]}]
#set_property PACKAGE_PIN W13 [get_ports {sw[7]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {sw[7]}]
#set_property PACKAGE_PIN V2 [get_ports {sw[8]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {sw[8]}]
```

Now you need to define your inputs and outputs, and to match them with Basys3 peripherals. For instance, we have two binary inputs A and B. We can use switches on the board for the inputs. In the Basys3 board, there are 16 switches (you can see from the constraint file that there are sixteen switches, from sw[0] to sw[15]).

To use switches for the inputs, you need to uncomment the corresponding lines. Then, change the variable names based on your design. You need to leave all other unused switches as commented.

For example, in order to use the first two switches for the inputs A and B, change the following lines from:

```
set_property PACKAGE_PIN V17 [get_ports {sw[0]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {sw[0]}]
set_property PACKAGE_PIN V16 [get_ports {sw[1]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {sw[1]}]
```

to:

```
set_property PACKAGE_PIN V17 [get_ports {A}]
        set_property IOSTANDARD LVCMOS33 [get_ports {A}]
set_property PACKAGE_PIN V16 [get_ports {B}]
    set_property IOSTANDARD LVCMOS33 [get_ports {B}]
```

You also need to assign the output D to a board peripheral. An LED is a useful option to check the function.

Change the following lines from:

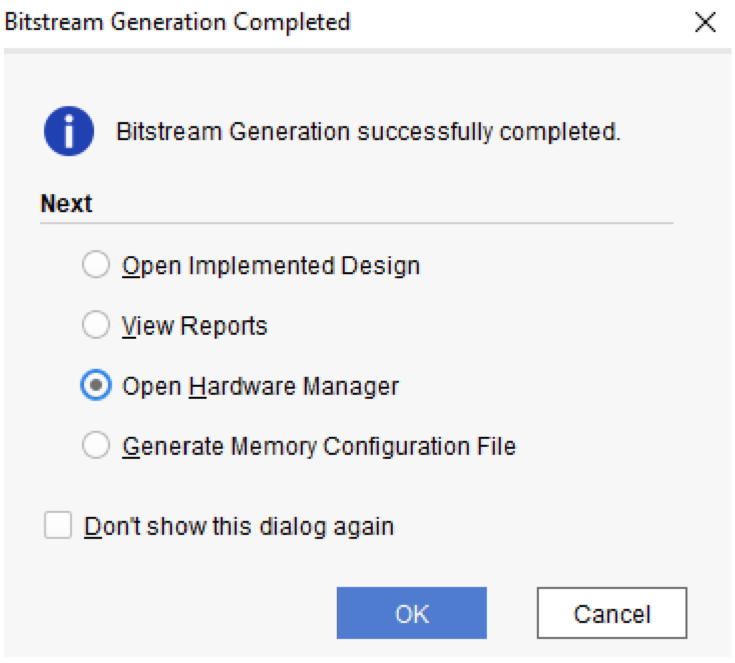
```
set_property PACKAGE_PIN U16 [get_ports {led[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {led[0]}]
```

to:

```
set_property PACKAGE_PIN U16 [get_ports {D}]
    set_property IOSTANDARD LVCMOS33 [get_ports {D}]
```

Do not forget to save the changes.

Step 10. Now click on Generate Bitstream file to make your design to 0s and 1s. After it is completed, open hardware manager.



Bitstream Generation Completed

Bitstream Generation successfully completed.

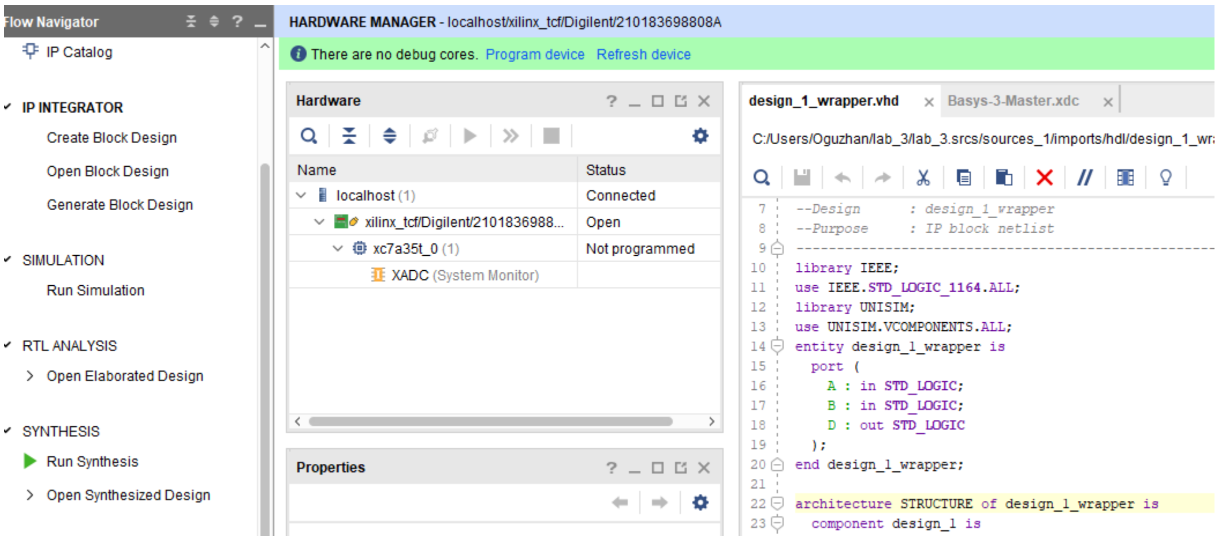
Next

- ( ) Open Implemented Design
- ( ) View Reports
- [✓ Open Hardware Manager]
- ( ) Generate Memory Configuration File

[ ] Don't show this dialog again

Connect the board to the pc through a USB cable.

Then, program the device.



HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210183698808A

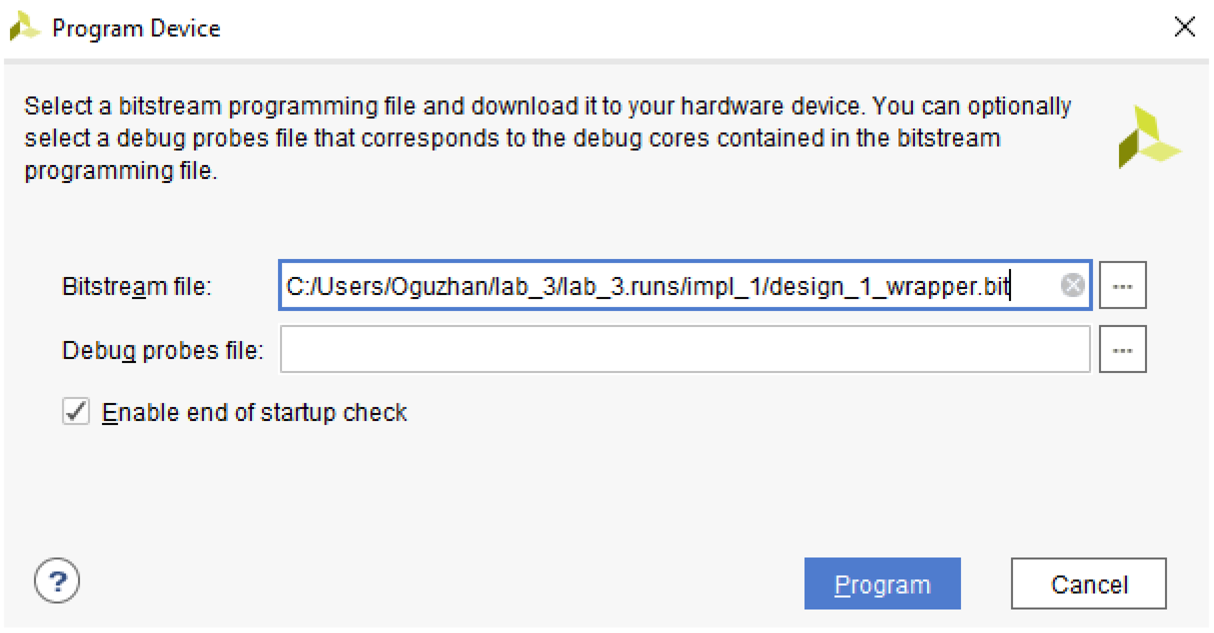
There are no debug cores. Program device Refresh device

Name	Status
localhost (1)	Connected
xilinx_tcf/Digilent/2101836988...	Open
xc7a35t_0 (1)	Not programmed
XADC (System Monitor)	

C:/Users/Oguzhan/lab_3/lab_3.srcs/sources_1/imports/hdl/design_1_wr

```
--Design      : design_1_wrapper
--Purpose     : IP block netlist
----------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
library UNISIM;
use UNISIM.VCOMPONENTS.ALL;
entity design_1_wrapper is
  port (
    A : in STD_LOGIC;
    B : in STD_LOGIC;
    D : out STD_LOGIC
  );
end design_1_wrapper;

architecture STRUCTURE of design_1_wrapper is
  component design_1 is
```



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/Oguzhan/lab_3/lab_3.runs/impl_1/design_1_wrapper.bit

Debug probes file:

[✓] Enable end of startup check

Step 11. Test the function of the design on the Basys3 board.