```
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
```

# AI-accelerator-course-preparation tutorial

# AI-accelerator-course-preparation tutorial

# Lab 1 Software platform Description

## 1.1 Goal

This part of the documentation provides an introduction for fresh users about how to install the Vivado® Design Suite on Windows or Linux OS(operating system).

## 1.2 Vivado® Design Suite

### Introduction

Vivado® Design Suite is a software suite produced by Xilinx for synthesis and analysis of HDL(Hardware description language) designs, superseding Xilinx ISE with additional features for SoC(system on a chip ) development and HLS(high-level synthesis).

It delivers a SoC-strength, IP-centric and system-centric, next generation development environment that has been built from the ground up to address the productivity bottlenecks in system-level integration and implementation. It comes in three editions:

- Vivado HL WebPack Edition
- Vivado HL Design Edition
- Vivado HL System Edition

### Component

#### Vivado High-Level Synthesis compiler

The **Vivado High-Level Synthesis** compiler enables C, C++ and SystemC programs to be directly targeted into Xilinx devices without the need to manually create RTL. Vivado HLS is widely reviewed to increase developer productivity, and is confirmed to support C++ classes, templates, functions and operator overloading.

#### Vivado Simulator

The **Vivado Simulator** is a component of the Vivado Design Suite. It is a compiled-language simulator that supports mixed-language, Tcl scripts, encrypted IP and enhanced verification.

#### Vivado IP Integrator

The **Vivado IP Integrator** allows engineers to quickly integrate and configure IP from the large Xilinx IP library. The Integrator is also tuned for MathWorks Simulink designs built with Xilinx's System Generator and Vivado High-Level Synthesis.

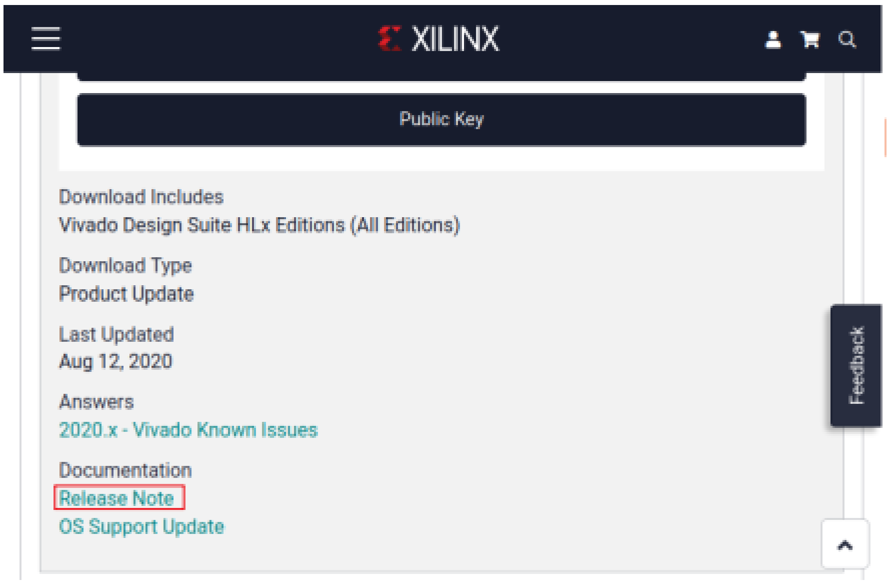
#### Vivado Tcl Store

The **Vivado Tcl Store** is a scripting system for developing add-ons to Vivado, and can be used to add and modify Vivado's capabilities. Tcl is the scripting language on which Vivado itself is based. All of Vivado's underlying functions can be invoked and controlled via Tcl scripts.

## 1.3 Resource downloading

Please check the link to download the required version of Vivado® Design Suite.
**Note**:

1. Download from the official website may require you register for an Xilinx account.
2. Suggested version of Vivado® Design Suite is **2020.1**, which i used to design and verify the lab and code. It can not be guaranteed that the lab code can work on all the versions of suite.
3. Strongly suggest reading the **release note** file under the documentation catalog, it will save your time for the platform preparation including the downloading, installation and the later licensing.



Download Includes
Vivado Design Suite HLx Editions (All Editions)

Download Type
Product Update

Last Updated
Aug 12, 2020

Answers
2020.x - Vivado Known Issues

Documentation
Release Note
OS Support Update

Figure 1.1

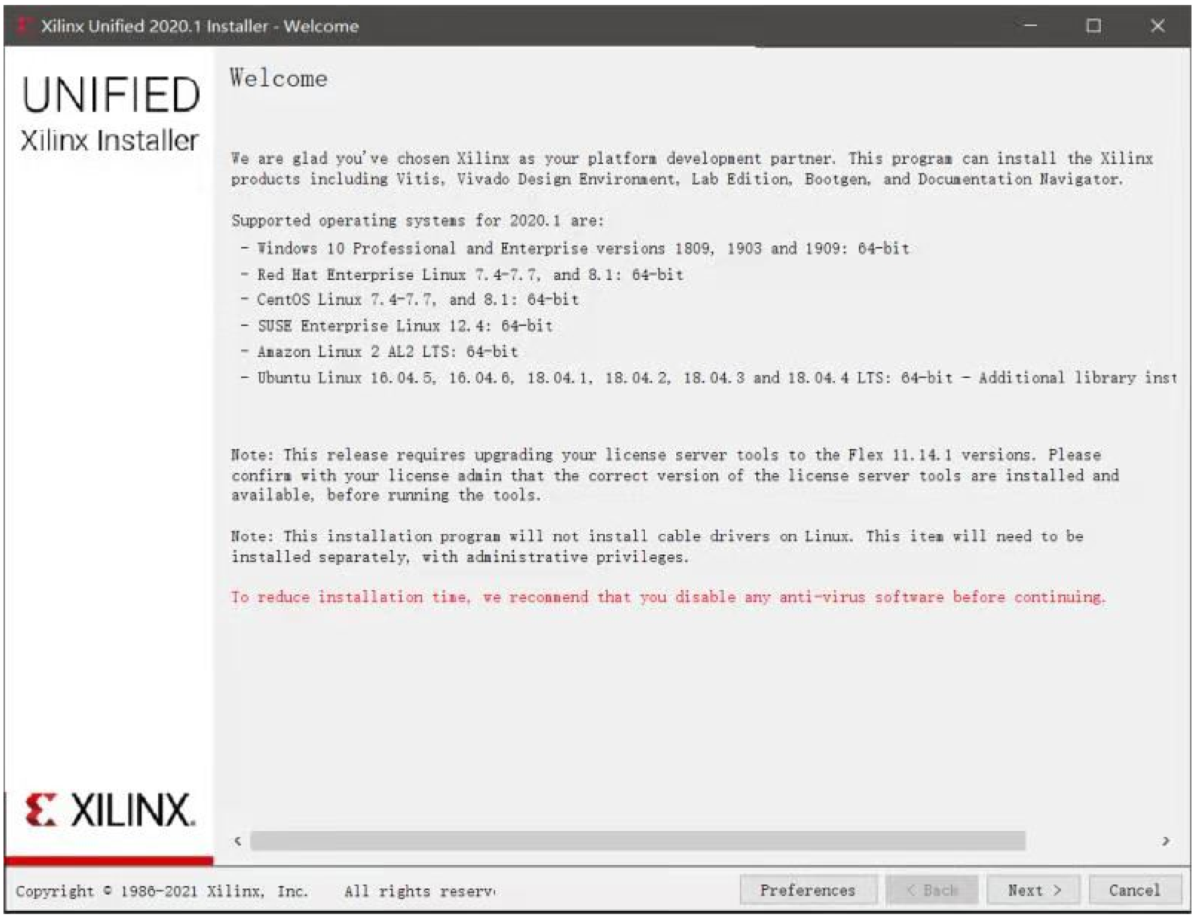
## 1.4 Final installation and licensing

After successful downloading of Vivado® Design Suite installer, it can be ready for installation. Please check the **release note** again to make sure your OS version can support the downloaded Vivado® Design Suite installer.

Here the documentation for Windows and Linux Ubuntu based installation instruction, you can choose from it, and skip another OS's description.

### Windows version

If you downloaded the lightweight installer, launch the downloaded file. You are prompted to log in and use your regular Xilinx login credentials to continue with the installation process.



Welcome

We are glad you've chosen Xilinx as your platform development partner. This program can install the Xilinx products including Vitis, Vivado Design Environment, Lab Edition, Bootgen, and Documentation Navigator.

Supported operating systems for 2020.1 are:

- Windows 10 Professional and Enterprise versions 1809, 1903 and 1909: 64-bit
- Red Hat Enterprise Linux 7.4-7.7, and 8.1: 64-bit
- CentOS Linux 7.4-7.7, and 8.1: 64-bit
- SUSE Enterprise Linux 12.4: 64-bit
- Amazon Linux 2 AL2 LTS: 64-bit
- Ubuntu Linux 16.04.5, 16.04.6, 18.04.1, 18.04.2, 18.04.3 and 18.04.4 LTS: 64-bit - Additional library inst

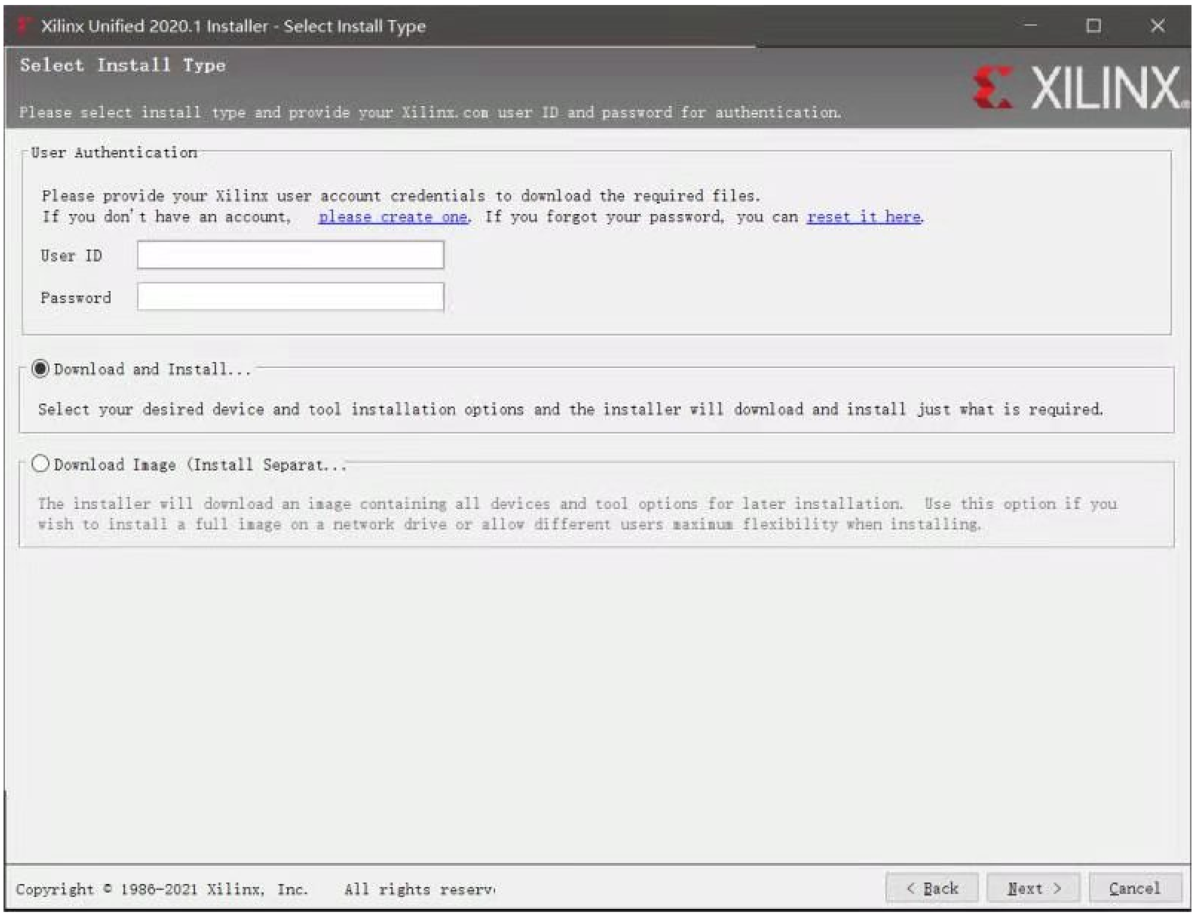
Note: This release requires upgrading your license server tools to the Flex 11.14.1 versions. Please confirm with your license admin that the correct version of the license server tools are installed and available, before running the tools.

Note: This installation program will not install cable drivers on Linux. This item will need to be installed separately, with administrative privileges.

To reduce installation time, we recommend that you disable any anti-virus software before continuing.

Copyright © 1986-2021 Xilinx, Inc.    All rights reserv

Figure 1.2



Select Install Type

Please select install type and provide your Xilinx.com user ID and password for authentication.

User Authentication

Please provide your Xilinx user account credentials to download the required files.
If you don't have an account, please create one. If you forgot your password, you can reset it here.

User ID

Password

(●) Download and Install...

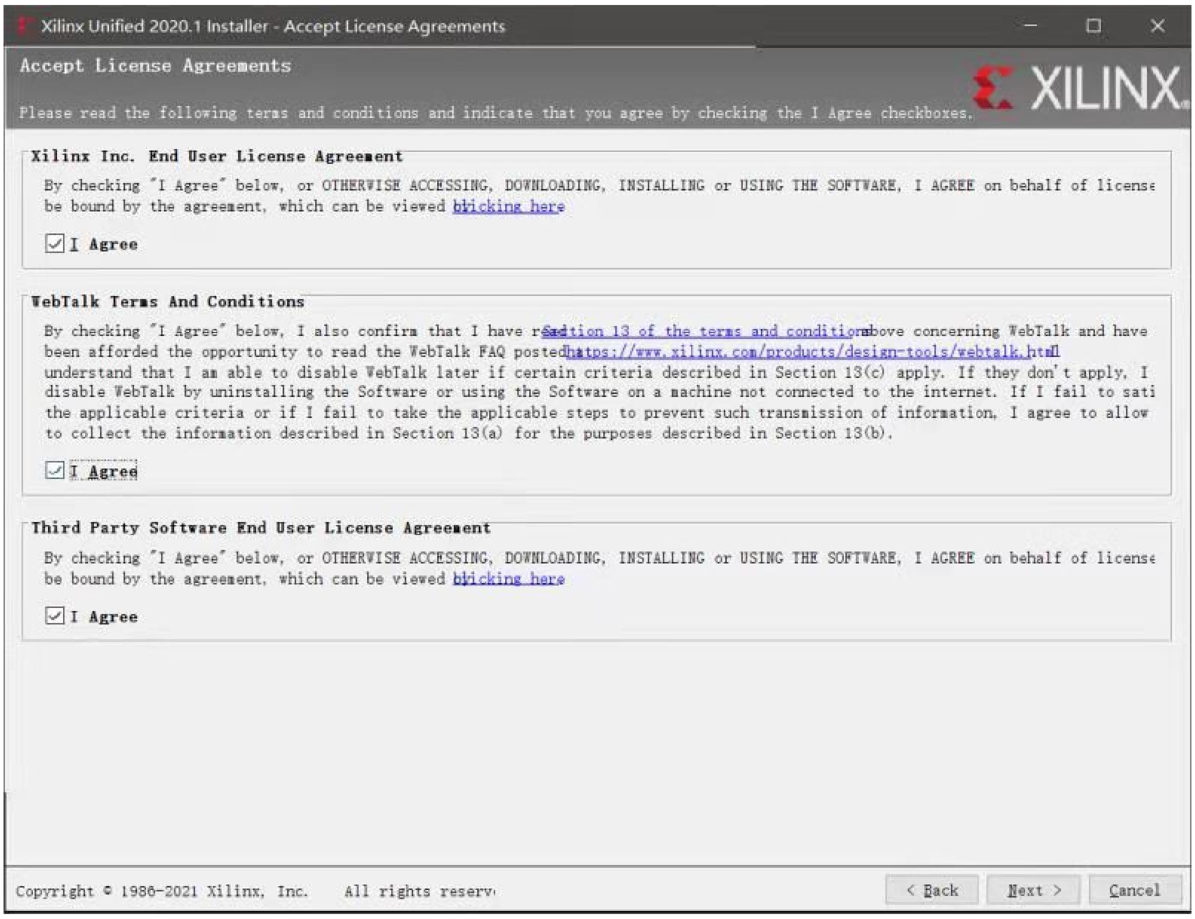
Select your desired device and tool installation options and the installer will download and install just what is required.

( ) Download Image (Install Separat...

The installer will download an image containing all devices and tool options for later installation. Use this option if you wish to install a full image on a network drive or allow different users maximum flexibility when installing.

Copyright © 1986-2021 Xilinx, Inc.    All rights reserv

Figure 1.3



Accept License Agreements

Please read the following terms and conditions and indicate that you agree by checking the I Agree checkboxes.

**Xilinx Inc. End User License Agreement**

By checking "I Agree" below, or OTHERWISE ACCESSING, DOWNLOADING, INSTALLING or USING THE SOFTWARE, I AGREE on behalf of license
be bound by the agreement, which can be viewed blicking here

[✓] I Agree

**WebTalk Terms And Conditions**

By checking "I Agree" below, I also confirm that I have reSection 13 of the terms and conditionsabove concerning WebTalk and have
been afforded the opportunity to read the WebTalk FAQ postedhttps://www.xilinx.com/products/design-tools/webtalk.html
understand that I am able to disable WebTalk later if certain criteria described in Section 13(c) apply. If they don't apply, I
disable WebTalk by uninstalling the Software or using the Software on a machine not connected to the internet. If I fail to sati
the applicable criteria or if I fail to take the applicable steps to prevent such transmission of information, I agree to allow
to collect the information described in Section 13(a) for the purposes described in Section 13(b).

[✓] I Agree

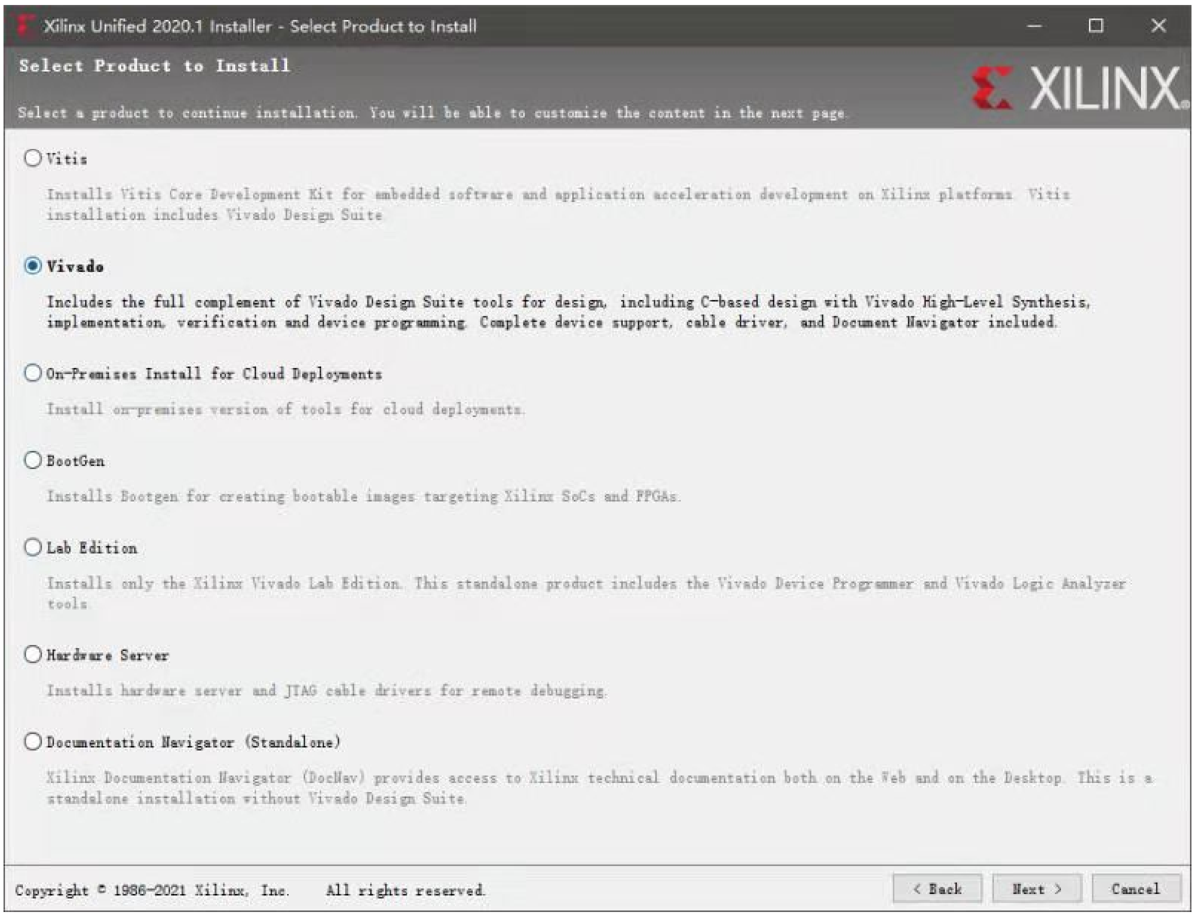
**Third Party Software End User License Agreement**

By checking "I Agree" below, or OTHERWISE ACCESSING, DOWNLOADING, INSTALLING or USING THE SOFTWARE, I AGREE on behalf of license
be bound by the agreement, which can be viewed blicking here

[✓] I Agree

Copyright © 1986-2021 Xilinx, Inc. All rights reserv

Figure 1.4



Select Product to Install

Select a product to continue installation. You will be able to customize the content in the next page.

○ Vitis

Installs Vitis Core Development Kit for embedded software and application acceleration development on Xilinx platforms. Vitis installation includes Vivado Design Suite.

[✓ Vivado]

Includes the full complement of Vivado Design Suite tools for design, including C-based design with Vivado High-Level Synthesis, implementation, verification and device programming. Complete device support, cable driver, and Document Navigator included.

○ On-Premises Install for Cloud Deployments

Install on-premises version of tools for cloud deployments.

○ BootGen

Installs Bootgen for creating bootable images targeting Xilinx SoCs and FPGAs.

○ Lab Edition

Installs only the Xilinx Vivado Lab Edition. This standalone product includes the Vivado Device Programmer and Vivado Logic Analyzer tools.

○ Hardware Server

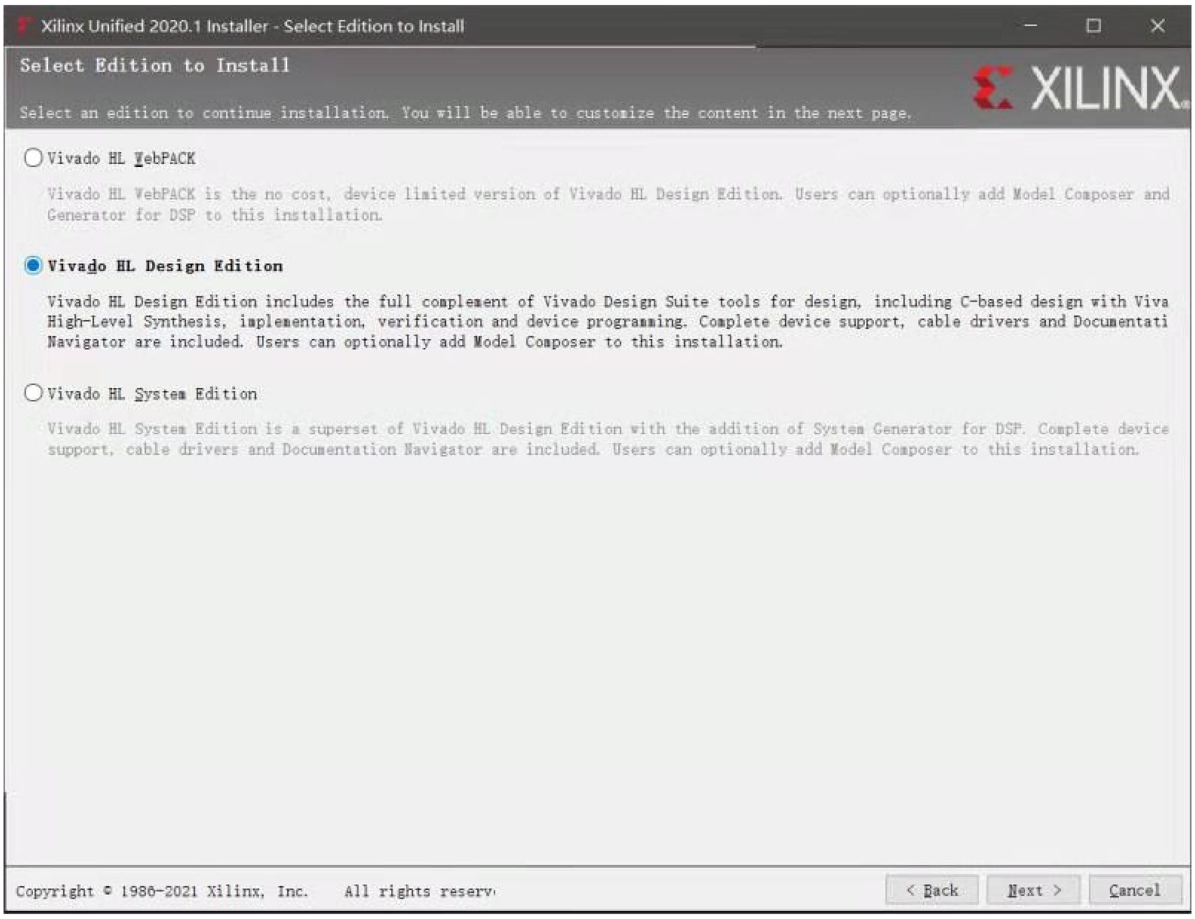
Installs hardware server and JTAG cable drivers for remote debugging.

○ Documentation Navigator (Standalone)

Xilinx Documentation Navigator (DocNav) provides access to Xilinx technical documentation both on the Web and on the Desktop. This is a standalone installation without Vivado Design Suite.

Copyright © 1986-2021 Xilinx, Inc. All rights reserved.

Figure 1.5



Xilinx Unified 2020.1 Installer - Select Edition to Install

Select Edition to Install

Select an edition to continue installation. You will be able to customize the content in the next page.

○ Vivado HL WebPACK

Vivado HL WebPACK is the no cost, device limited version of Vivado HL Design Edition. Users can optionally add Model Composer and Generator for DSP to this installation.

[✓ Vivado HL Design Edition]

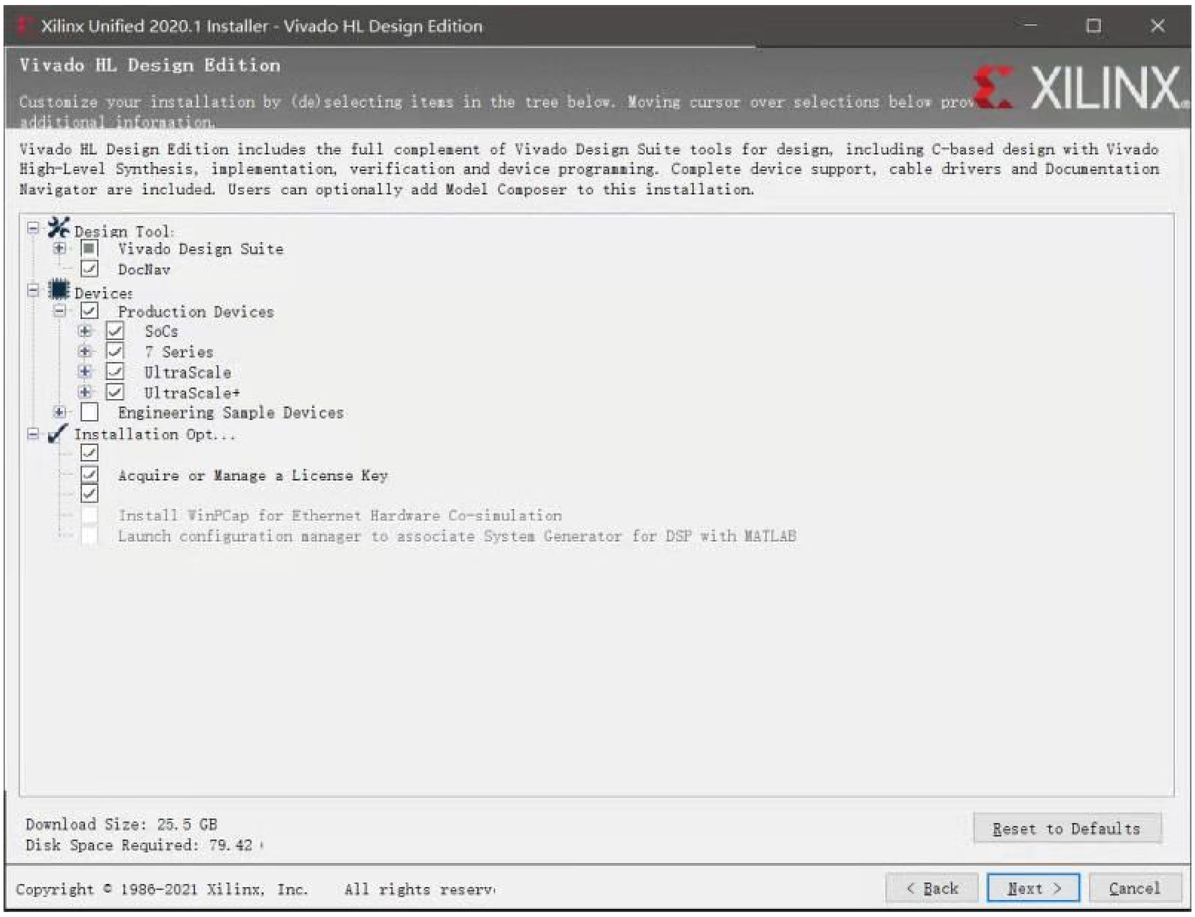
Vivado HL Design Edition includes the full complement of Vivado Design Suite tools for design, including C-based design with Viva High-Level Synthesis, implementation, verification and device programming. Complete device support, cable drivers and Documentati Navigator are included. Users can optionally add Model Composer to this installation.

○ Vivado HL System Edition

Vivado HL System Edition is a superset of Vivado HL Design Edition with the addition of System Generator for DSP. Complete device support, cable drivers and Documentation Navigator are included. Users can optionally add Model Composer to this installation.

Copyright © 1986-2021 Xilinx, Inc. All rights reserv

Figure 1.6



Xilinx Unified 2020.1 Installer - Vivado HL Design Edition

Vivado HL Design Edition

Customize your installation by (de)selecting items in the tree below. Moving cursor over selections below prov additional information.

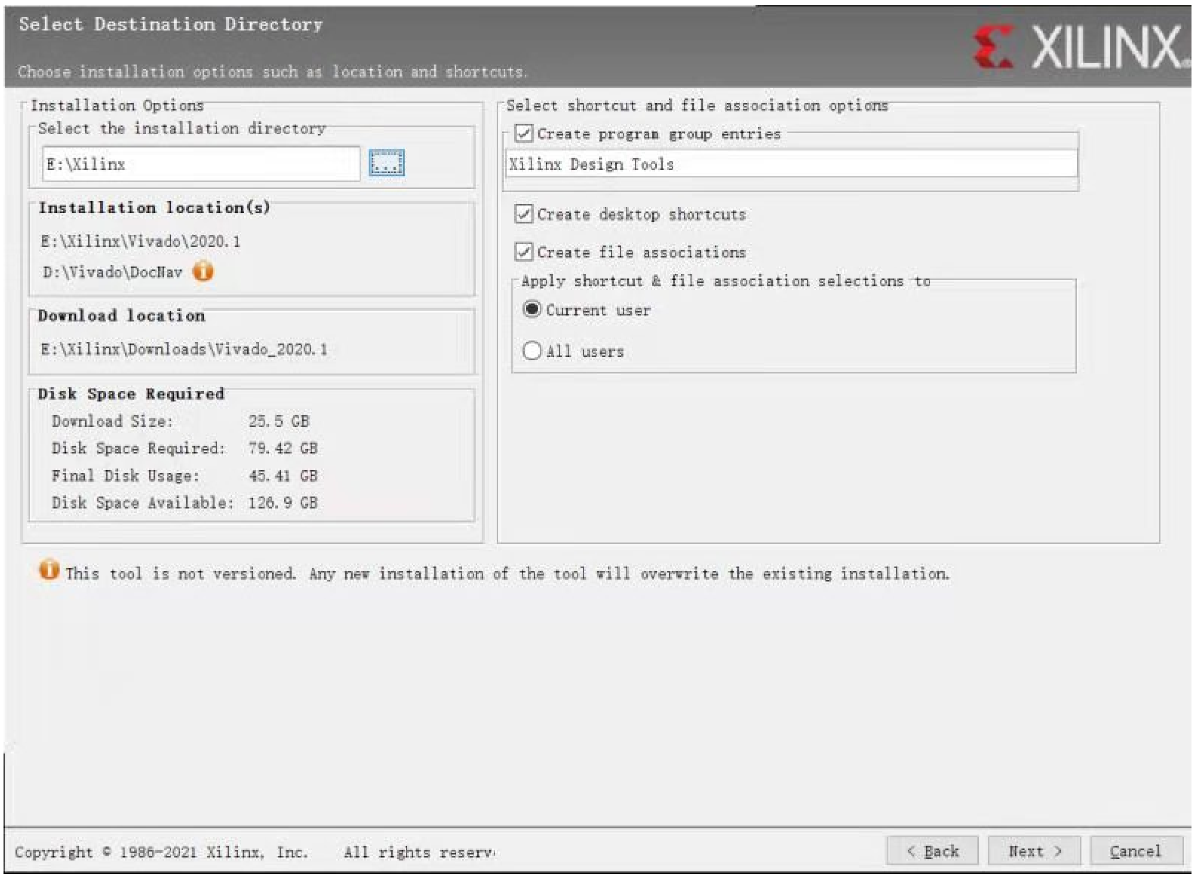
Vivado HL Design Edition includes the full complement of Vivado Design Suite tools for design, including C-based design with Vivado High-Level Synthesis, implementation, verification and device programming. Complete device support, cable drivers and Documentation Navigator are included. Users can optionally add Model Composer to this installation.

- Design Tool:
  - ■ Vivado Design Suite
  - ✓ DocNav
- Devices
  - ✓ Production Devices
    - ✓ SoCs
    - ✓ 7 Series
    - ✓ UltraScale
    - ✓ UltraScale+
  - ☐ Engineering Sample Devices
- Installation Opt...
  - ✓
  - ✓ Acquire or Manage a License Key
  - ✓
  - ☐ Install WinPCap for Ethernet Hardware Co-simulation
  - ☐ Launch configuration manager to associate System Generator for DSP with MATLAB

Download Size: 25.5 GB
Disk Space Required: 79.42

Copyright © 1986-2021 Xilinx, Inc. All rights reserv

Figure 1.7


Select Destination Directory

Choose installation options such as location and shortcuts.

Installation Options

Select the installation directory: E:\Xilinx

Installation location(s)
E:\Xilinx\Vivado\2020.1
D:\Vivado\DocNav

Download location
E:\Xilinx\Downloads\Vivado_2020.1

Disk Space Required

|  |  |
|---|---|
| Download Size: | 25.5 GB |
| Disk Space Required: | 79.42 GB |
| Final Disk Usage: | 45.41 GB |
| Disk Space Available: | 126.9 GB |

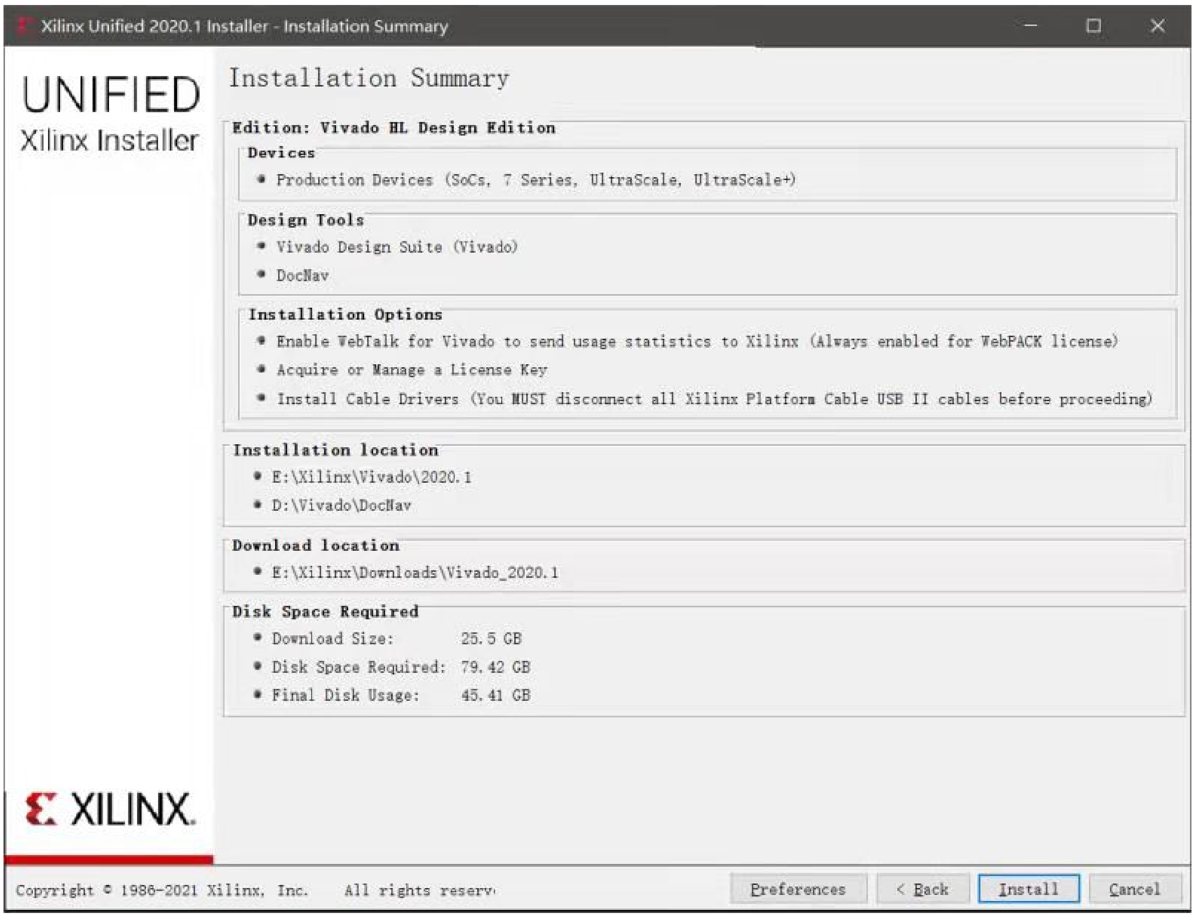
Select shortcut and file association options
[✓] Create program group entries: Xilinx Design Tools
[✓] Create desktop shortcuts
[✓] Create file associations

Apply shortcut & file association selections to
[✓ Current user]
All users

This tool is not versioned. Any new installation of the tool will overwrite the existing installation.

Copyright © 1986-2021 Xilinx, Inc. All rights reserv

Figure 1.8


Xilinx Unified 2020.1 Installer - Installation Summary

UNIFIED Xilinx Installer

Installation Summary

Edition: Vivado HL Design Edition

Devices
- Production Devices (SoCs, 7 Series, UltraScale, UltraScale+)

Design Tools
- Vivado Design Suite (Vivado)
- DocNav

Installation Options
- Enable WebTalk for Vivado to send usage statistics to Xilinx (Always enabled for WebPACK license)
- Acquire or Manage a License Key
- Install Cable Drivers (You MUST disconnect all Xilinx Platform Cable USB II cables before proceeding)

Installation location
- E:\Xilinx\Vivado\2020.1
- D:\Vivado\DocNav

Download location
- E:\Xilinx\Downloads\Vivado_2020.1

Disk Space Required
- Download Size: 25.5 GB
- Disk Space Required: 79.42 GB
- Final Disk Usage: 45.41 GB

Copyright © 1986-2021 Xilinx, Inc. All rights reserv

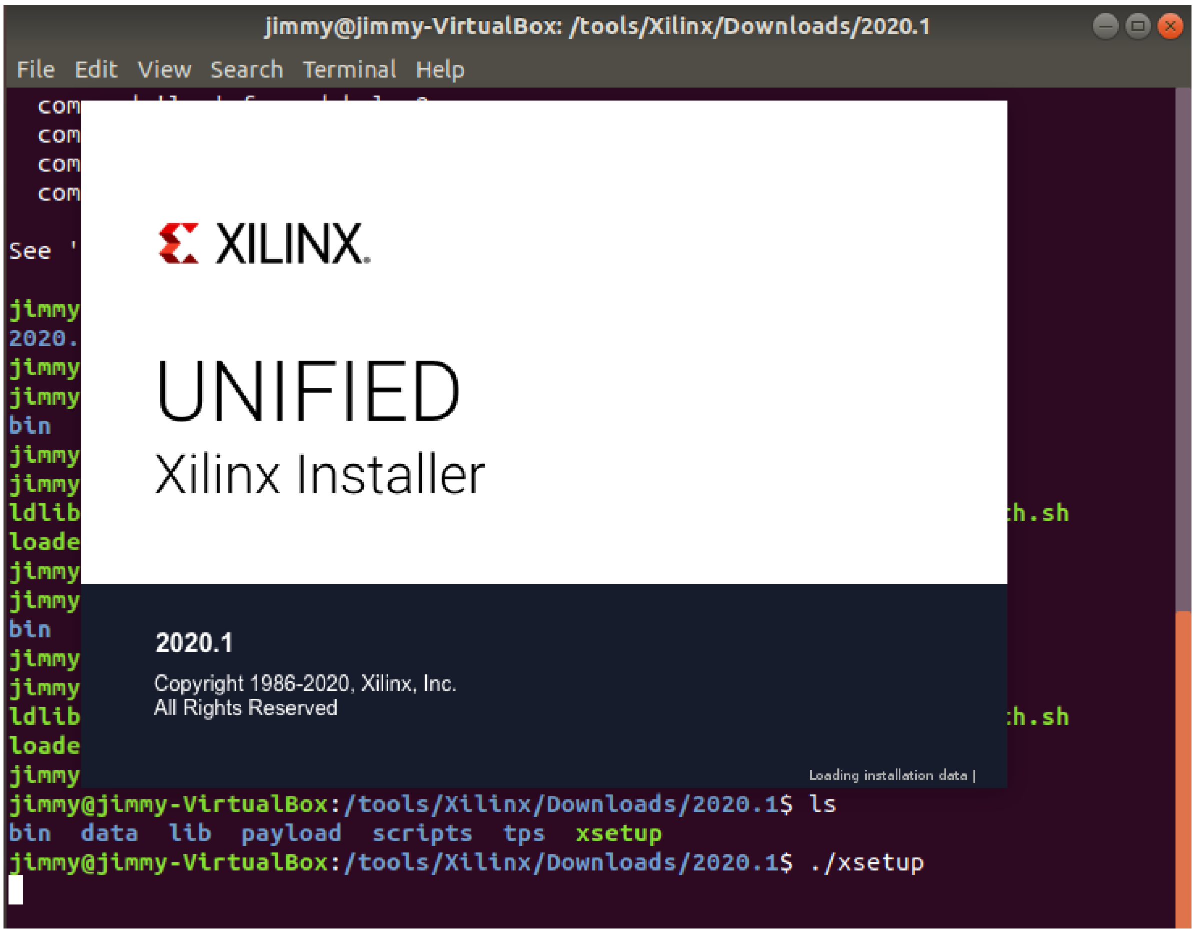
Figure 1.9

## Linux version

Find the mirror image in the resoure link: Ubuntu Linux's installation is not difficult. Two options are available:

1. Direct installation on your PC or Laptop, but it will erase all the files from your previous OS.
2. Virtual Machine(Oracle or VM workstation).
   For option 1, please refer to this guide for USB booting or DVD formatting.

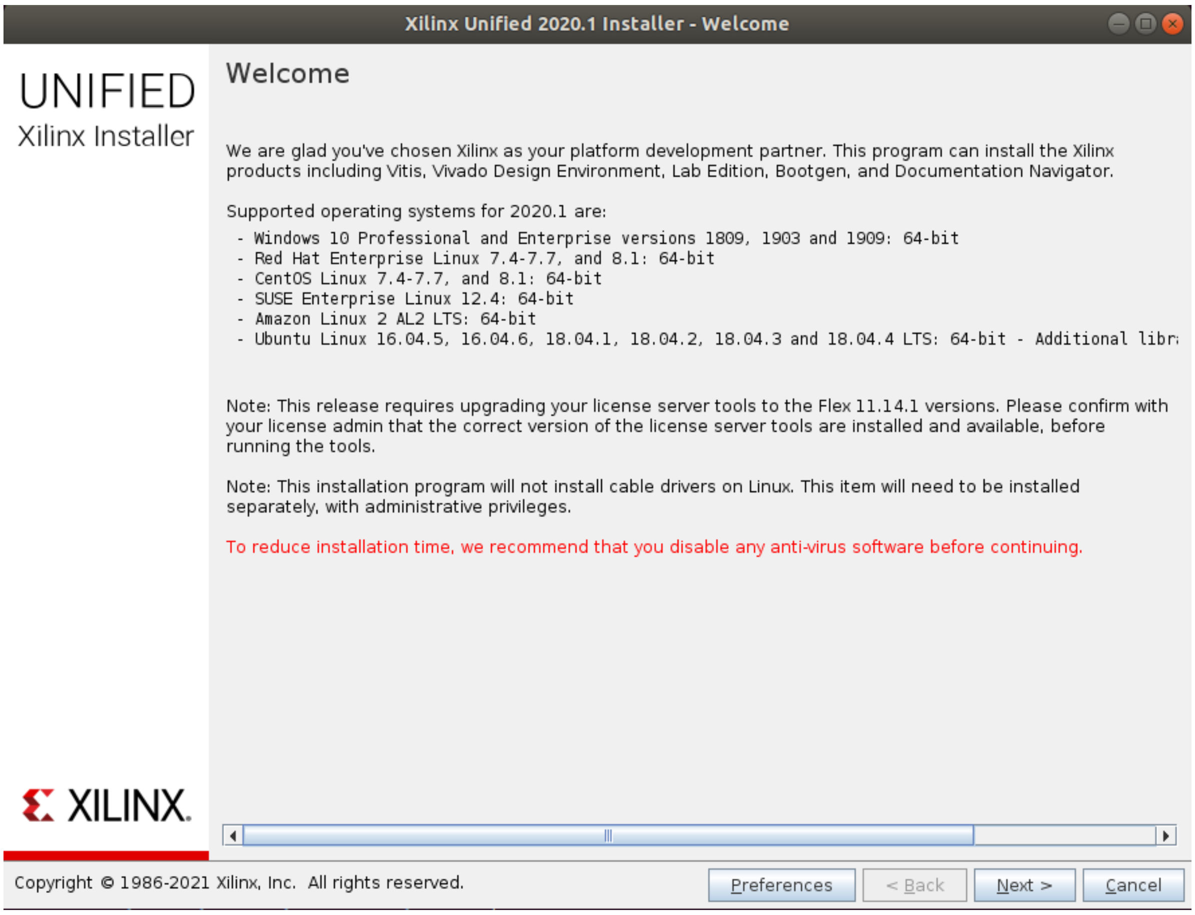
If you downloaded the lightweight installer, launch the downloaded file. You are prompted to log in and use your regular Xilinx login credentials to continue with the installation process.
**Note:** On Linux the file is a .bin file and can be launched by running ./.bin in the shell terminal, as can be seen in [Figure 1.10][#110] .
Please ensure that you have changed the file permissions to execute.


jimmy@jimmy-VirtualBox: /tools/Xilinx/Downloads/2020.1
File Edit View Search Terminal Help
XILINX
UNIFIED
Xilinx Installer
2020.1
Copyright 1986-2020, Xilinx, Inc.
All Rights Reserved
Loading installation data |

```
jimmy@jimmy-VirtualBox:/tools/Xilinx/Downloads/2020.1$ ls
bin  data  lib  payload  scripts  tps  xsetup
jimmy@jimmy-VirtualBox:/tools/Xilinx/Downloads/2020.1$ ./xsetup
```

Figure 1.10

Following the initializing of the installer, a welcome window will appear with the necessary information about the supported operating systems, shown in [Figure 1.11][111].



Welcome

We are glad you've chosen Xilinx as your platform development partner. This program can install the Xilinx products including Vitis, Vivado Design Environment, Lab Edition, Bootgen, and Documentation Navigator.

Supported operating systems for 2020.1 are:

```
 - Windows 10 Professional and Enterprise versions 1809, 1903 and 1909: 64-bit
 - Red Hat Enterprise Linux 7.4-7.7, and 8.1: 64-bit
 - CentOS Linux 7.4-7.7, and 8.1: 64-bit
 - SUSE Enterprise Linux 12.4: 64-bit
 - Amazon Linux 2 AL2 LTS: 64-bit
 - Ubuntu Linux 16.04.5, 16.04.6, 18.04.1, 18.04.2, 18.04.3 and 18.04.4 LTS: 64-bit - Additional libr
```

Note: This release requires upgrading your license server tools to the Flex 11.14.1 versions. Please confirm with your license admin that the correct version of the license server tools are installed and available, before running the tools.

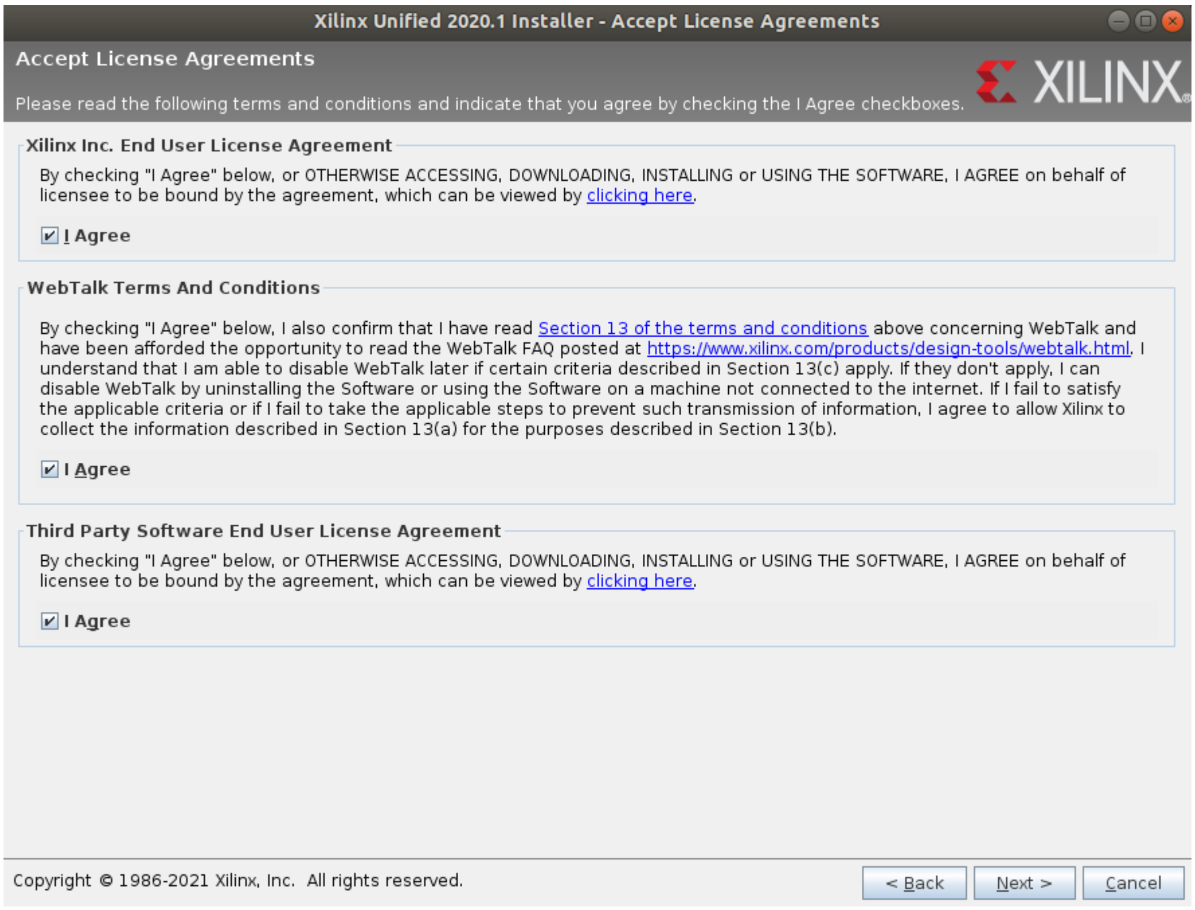
Note: This installation program will not install cable drivers on Linux. This item will need to be installed separately, with administrative privileges.

To reduce installation time, we recommend that you disable any anti-virus software before continuing.

Copyright © 1986-2021 Xilinx, Inc. All rights reserved.

Figure 1.11

Accept all the license agreement, otherwise the installation will not proceed, displayed in Figure 1.12.



Accept License Agreements

Please read the following terms and conditions and indicate that you agree by checking the I Agree checkboxes.

Xilinx Inc. End User License Agreement

By checking "I Agree" below, or OTHERWISE ACCESSING, DOWNLOADING, INSTALLING or USING THE SOFTWARE, I AGREE on behalf of licensee to be bound by the agreement, which can be viewed by clicking here.

[✓] I Agree

WebTalk Terms And Conditions

By checking "I Agree" below, I also confirm that I have read Section 13 of the terms and conditions above concerning WebTalk and have been afforded the opportunity to read the WebTalk FAQ posted at https://www.xilinx.com/products/design-tools/webtalk.html. I understand that I am able to disable WebTalk later if certain criteria described in Section 13(c) apply. If they don't apply, I can disable WebTalk by uninstalling the Software or using the Software on a machine not connected to the internet. If I fail to satisfy the applicable criteria or if I fail to take the applicable steps to prevent such transmission of information, I agree to allow Xilinx to collect the information described in Section 13(a) for the purposes described in Section 13(b).

[✓] I Agree

Third Party Software End User License Agreement

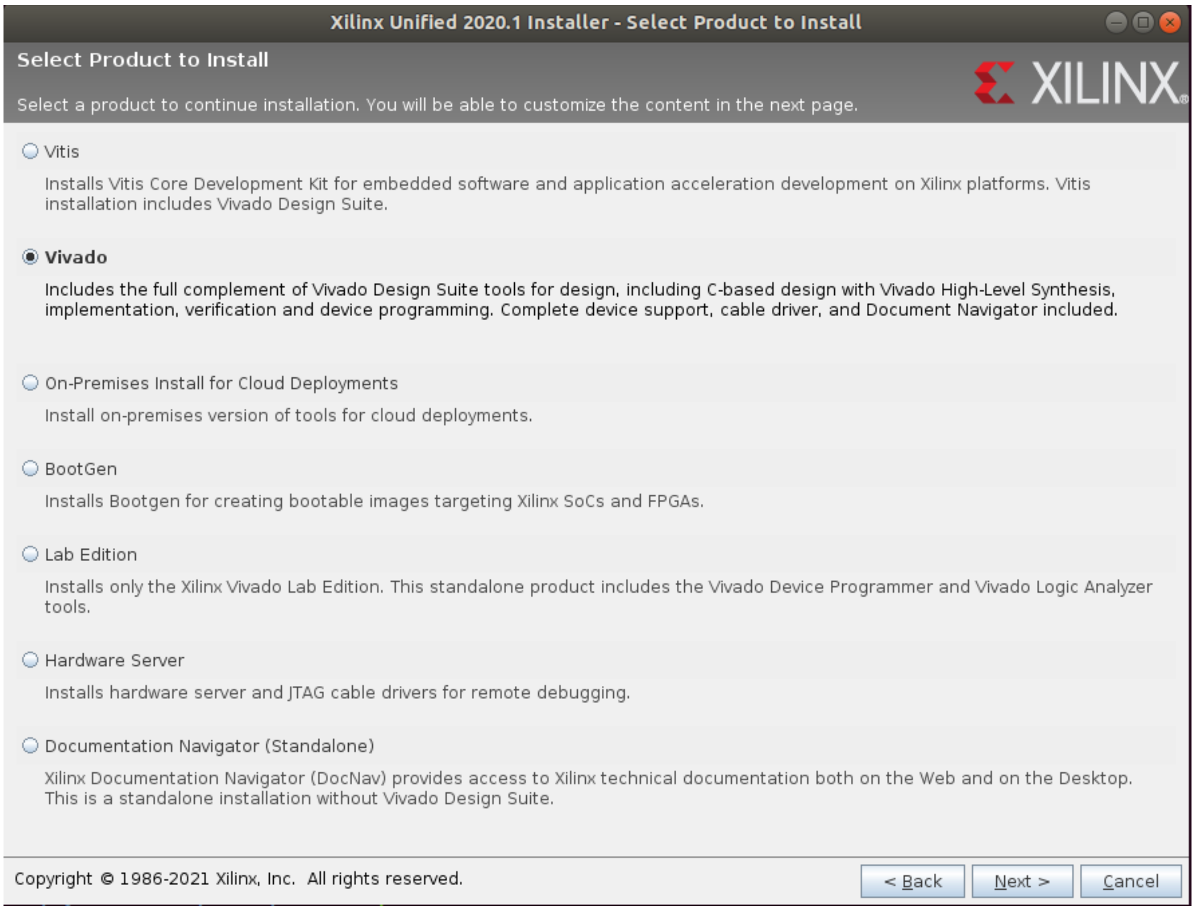
By checking "I Agree" below, or OTHERWISE ACCESSING, DOWNLOADING, INSTALLING or USING THE SOFTWARE, I AGREE on behalf of licensee to be bound by the agreement, which can be viewed by clicking here.

[✓] I Agree

Copyright © 1986-2021 Xilinx, Inc. All rights reserved.

Figure 1.12

Shown in Figure 4, take the Vivado product, including all the required toolkits and library. To experience the Machine learning and AIoT function, you can take the Vitis option, which also includes the Vivado Design Suite.



**Select Product to Install**

Select a product to continue installation. You will be able to customize the content in the next page.

○ Vitis
Installs Vitis Core Development Kit for embedded software and application acceleration development on Xilinx platforms. Vitis installation includes Vivado Design Suite.

● **Vivado**
Includes the full complement of Vivado Design Suite tools for design, including C-based design with Vivado High-Level Synthesis, implementation, verification and device programming. Complete device support, cable driver, and Document Navigator included.

○ On-Premises Install for Cloud Deployments
Install on-premises version of tools for cloud deployments.

○ BootGen
Installs Bootgen for creating bootable images targeting Xilinx SoCs and FPGAs.

○ Lab Edition
Installs only the Xilinx Vivado Lab Edition. This standalone product includes the Vivado Device Programmer and Vivado Logic Analyzer tools.

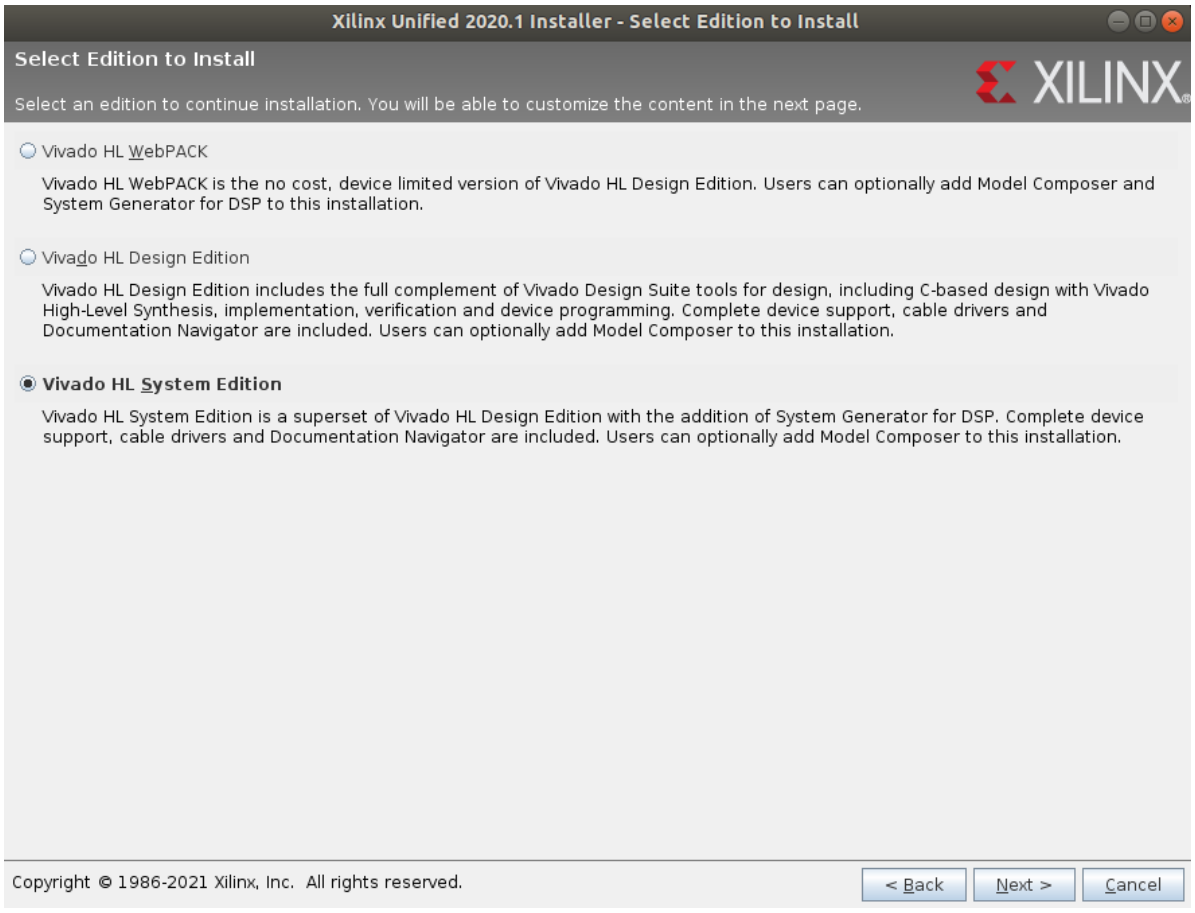
○ Hardware Server
Installs hardware server and JTAG cable drivers for remote debugging.

○ Documentation Navigator (Standalone)
Xilinx Documentation Navigator (DocNav) provides access to Xilinx technical documentation both on the Web and on the Desktop. This is a standalone installation without Vivado Design Suite.

Copyright © 1986-2021 Xilinx, Inc. All rights reserved.

Figure 1.13

Choose the Vivado HL System Edition.



Xilinx Unified 2020.1 Installer - Select Edition to Install

Select Edition to Install

Select an edition to continue installation. You will be able to customize the content in the next page.

( ) Vivado HL WebPACK

Vivado HL WebPACK is the no cost, device limited version of Vivado HL Design Edition. Users can optionally add Model Composer and System Generator for DSP to this installation.

( ) Vivado HL Design Edition

Vivado HL Design Edition includes the full complement of Vivado Design Suite tools for design, including C-based design with Vivado High-Level Synthesis, implementation, verification and device programming. Complete device support, cable drivers and Documentation Navigator are included. Users can optionally add Model Composer to this installation.

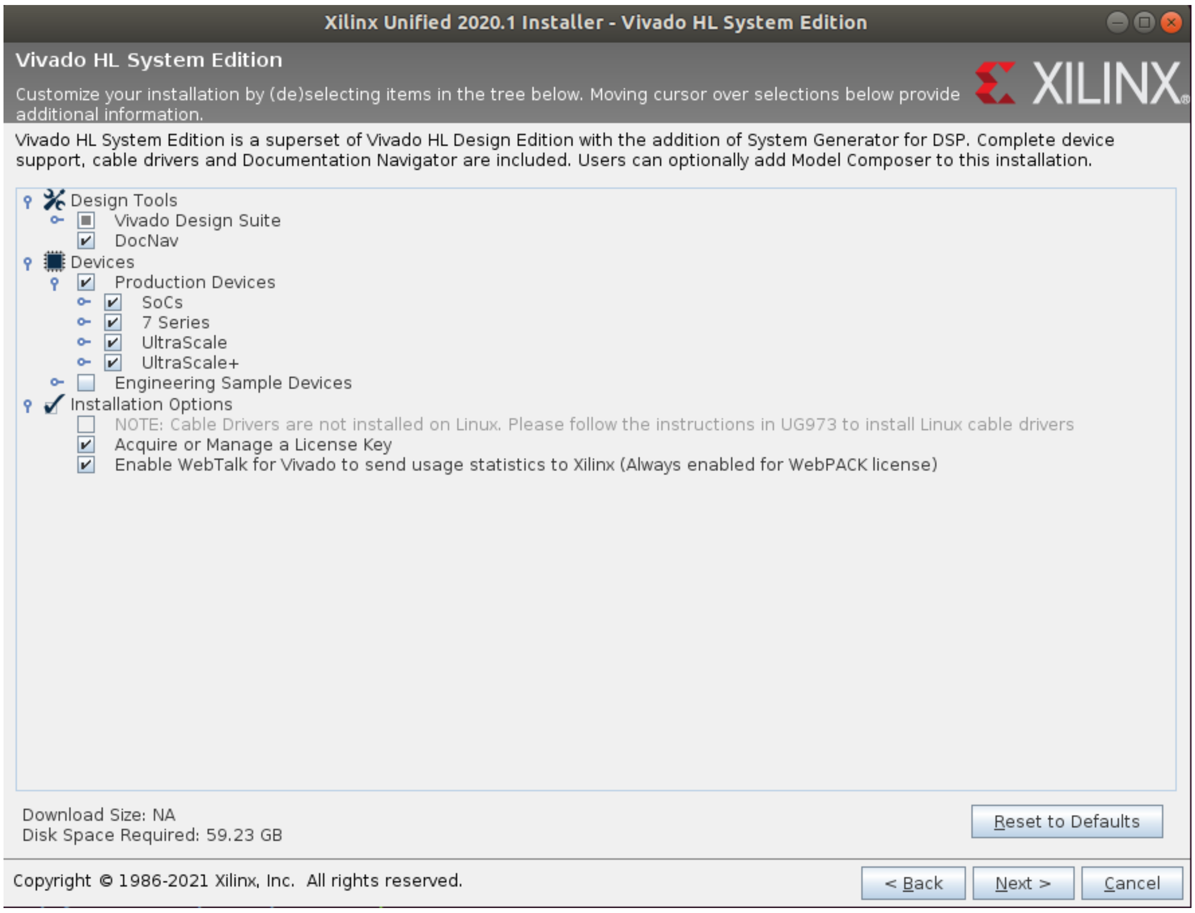
[✓ **Vivado HL System Edition**]

Vivado HL System Edition is a superset of Vivado HL Design Edition with the addition of System Generator for DSP. Complete device support, cable drivers and Documentation Navigator are included. Users can optionally add Model Composer to this installation.

Copyright © 1986-2021 Xilinx, Inc. All rights reserved.

Figure 1.14

After the Edition version is set, the installer will display the available design tools for your reference. Here i suggest selecting all the tools and devices, shown in Figure 6.



Xilinx Unified 2020.1 Installer - Vivado HL System Edition

Vivado HL System Edition

Customize your installation by (de)selecting items in the tree below. Moving cursor over selections below provide additional information.

Vivado HL System Edition is a superset of Vivado HL Design Edition with the addition of System Generator for DSP. Complete device support, cable drivers and Documentation Navigator are included. Users can optionally add Model Composer to this installation.

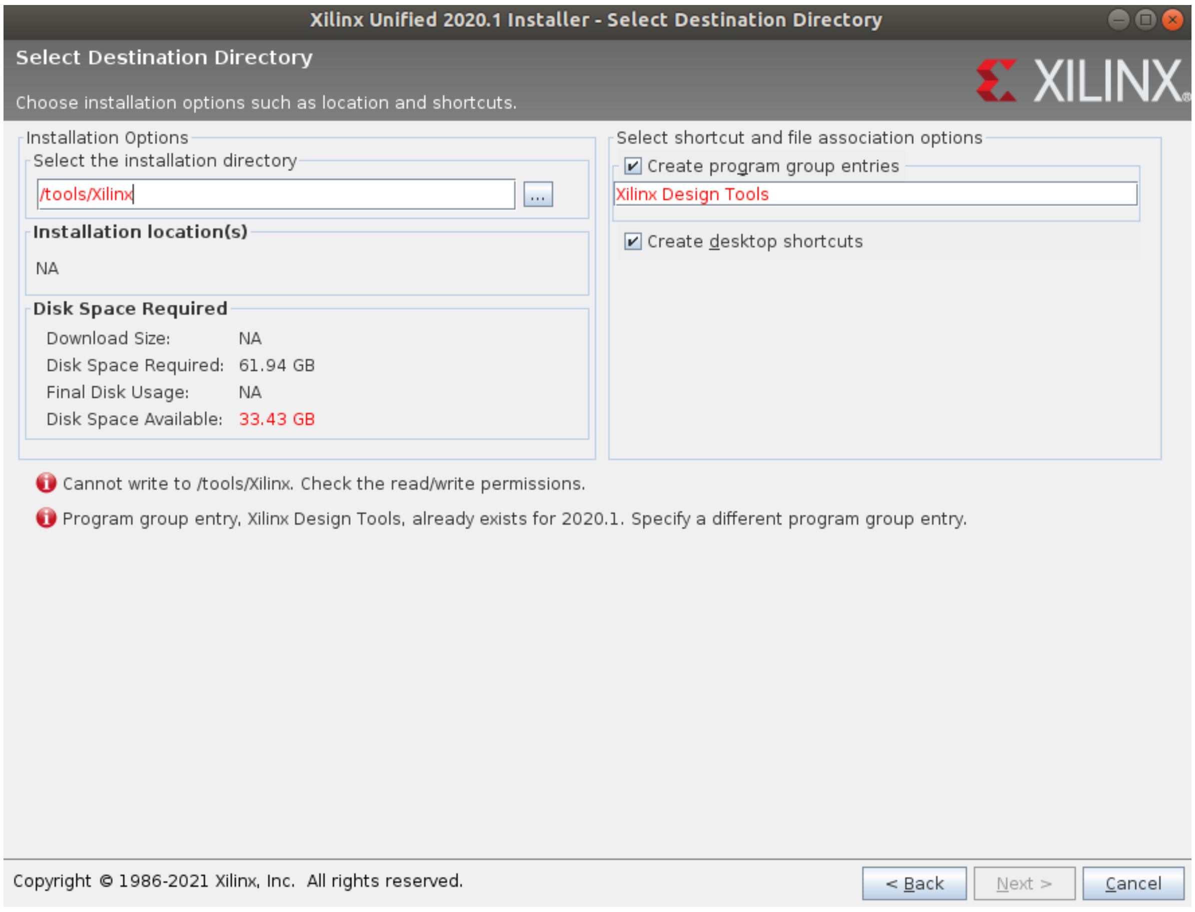
- Design Tools
  - ▣ Vivado Design Suite
  - ✓ DocNav
- Devices
  - ✓ Production Devices
    - ✓ SoCs
    - ✓ 7 Series
    - ✓ UltraScale
    - ✓ UltraScale+
  - ☐ Engineering Sample Devices
- Installation Options
  - ☐ NOTE: Cable Drivers are not installed on Linux. Please follow the instructions in UG973 to install Linux cable drivers
  - ✓ Acquire or Manage a License Key
  - ✓ Enable WebTalk for Vivado to send usage statistics to Xilinx (Always enabled for WebPACK license)

Download Size: NA
Disk Space Required: 59.23 GB

Copyright © 1986-2021 Xilinx, Inc. All rights reserved.

Figure 1.15

Choose the installation location.


Xilinx Unified 2020.1 Installer - Select Destination Directory

Select Destination Directory

Choose installation options such as location and shortcuts.

Installation Options

Select the installation directory

/tools/Xilinx

Installation location(s)

NA

Disk Space Required

| | |
|---|---|
| Download Size: | NA |
| Disk Space Required: | 61.94 GB |
| Final Disk Usage: | NA |
| Disk Space Available: | 33.43 GB |

Select shortcut and file association options

✓ Create program group entries

Xilinx Design Tools

✓ Create desktop shortcuts

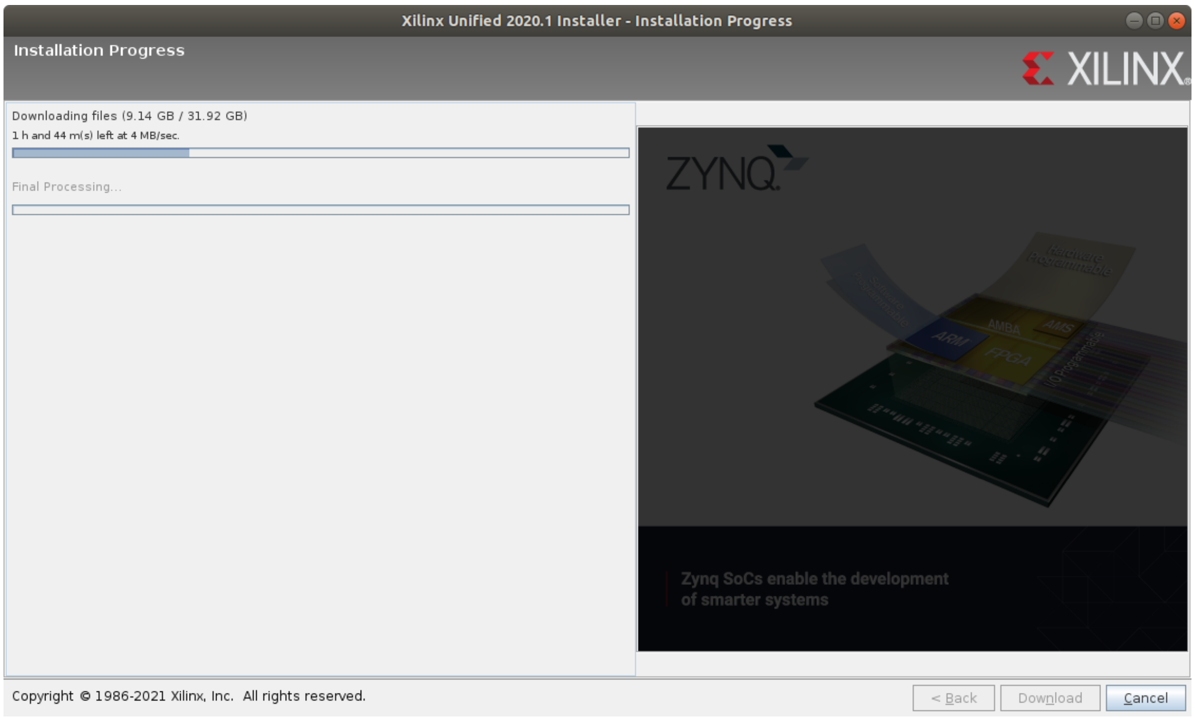
Cannot write to /tools/Xilinx. Check the read/write permissions.

Program group entry, Xilinx Design Tools, already exists for 2020.1. Specify a different program group entry.

Copyright © 1986-2021 Xilinx, Inc. All rights reserved.

Figure 1.16

Finally the installation will automatically progress and logically it will succeed. If not, the problem might link to the disk room for storage and processing or the system's compatibility.


Xilinx Unified 2020.1 Installer - Installation Progress

Installation Progress

Downloading files (9.14 GB / 31.92 GB)

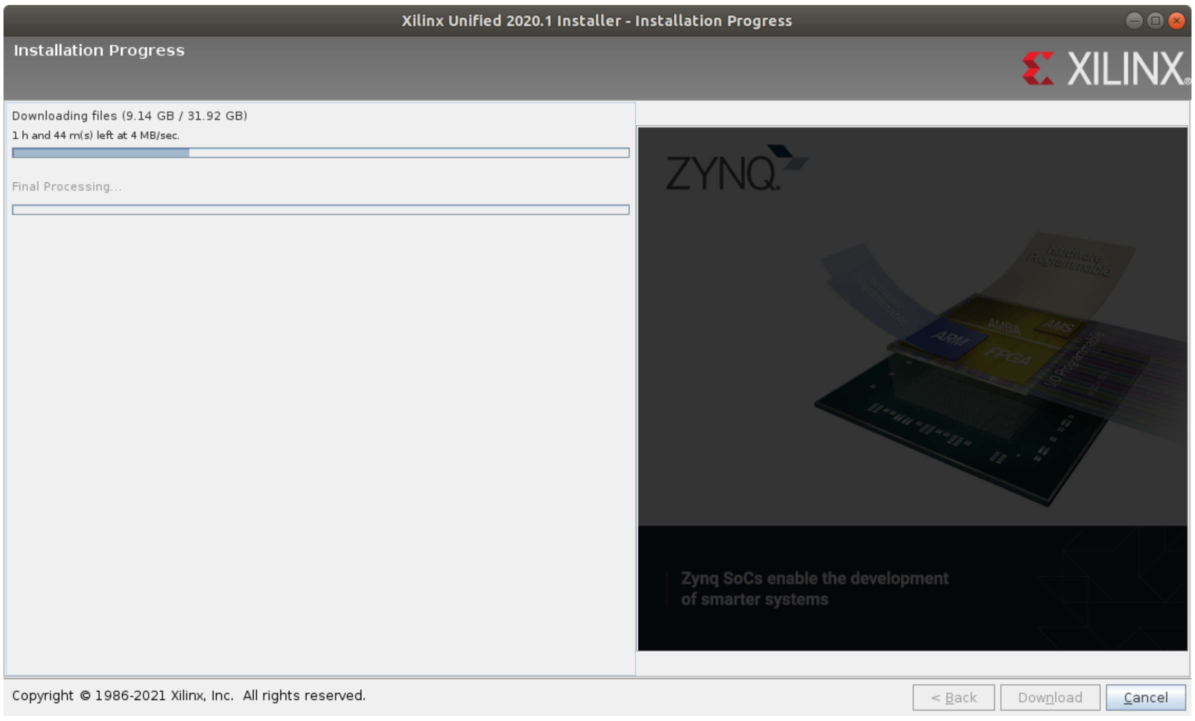
1 h and 44 m(s) left at 4 MB/sec.

Final Processing...

Zynq SoCs enable the development of smarter systems

Copyright © 1986-2021 Xilinx, Inc. All rights reserved.

Figure 1.17


Xilinx Unified 2020.1 Installer - Installation Progress
Installation Progress
XILINX
ZYNQ
Zynq SoCs enable the development of smarter systems
< Back
Download
Cancel

Downloading files (9.14 GB / 31.92 GB)
1 h and 44 m(s) left at 4 MB/sec.

Final Processing...

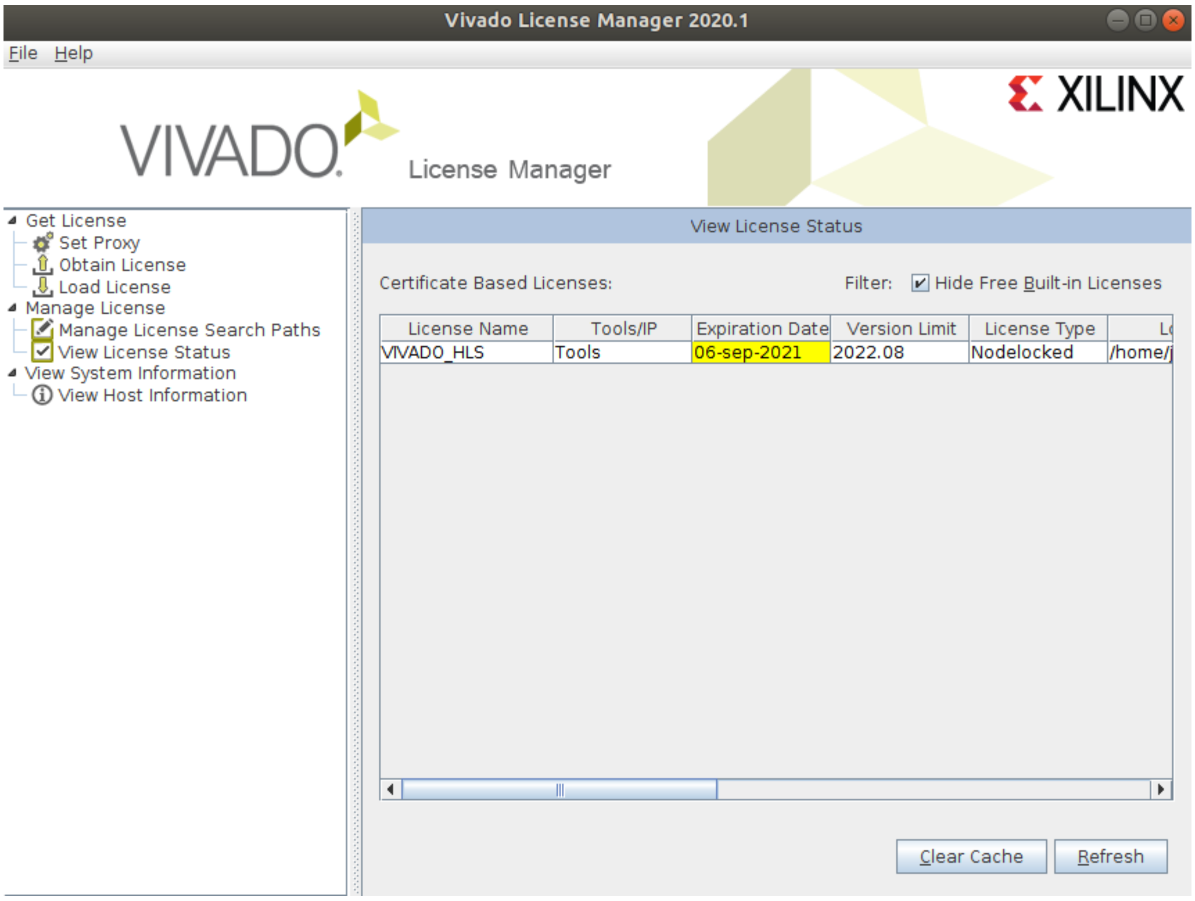
Copyright © 1986-2021 Xilinx, Inc. All rights reserved.

Figure 1.18

## Licensing

Xilinx charges for IP design and software and profits from the business. Currently, you can have one month's trial free of charge with your registered information.

After completing the installation of Vivado, SDx or ISE Design Suite, the Xilinx License Configuration Manager (XCLM) will start automatically and guide you through the licensing process,displayed in Figure 9. You may also go directly to the Xilinx Product Licensing Site to obtain licenses for free or evaluation products if you decided to skip this step during product installation.

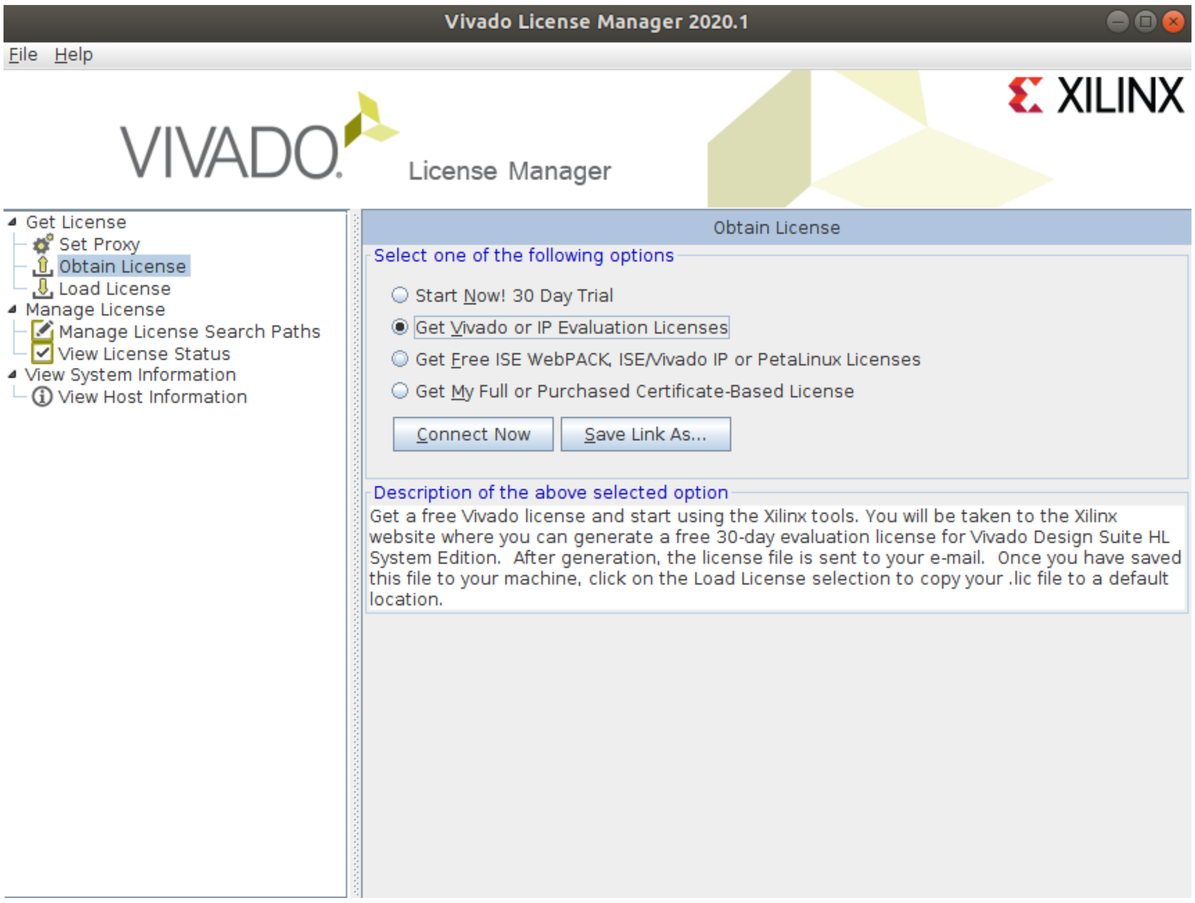


Certificate Based Licenses: Filter: [✓] Hide Free Built-in Licenses

| License Name | Tools/IP | Expiration Date | Version Limit | License Type | Lo |
|---|---|---|---|---|---|
| VIVADO_HLS | Tools | 06-sep-2021 | 2022.08 | Nodelocked | /home/j |

Figure 1.19

Click the **obtain license** part on the left side, choose the second option **Get Vivado or IP Evalution Licenses**, you will have 30 days' free use of Vivado Design Suite. And confirm with clicking the **Connect Now**.



Select one of the following options

- ○ Start Now! 30 Day Trial
- ● Get Vivado or IP Evaluation Licenses
- ○ Get Free ISE WebPACK, ISE/Vivado IP or PetaLinux Licenses
- ○ Get My Full or Purchased Certificate-Based License

Description of the above selected option

Get a free Vivado license and start using the Xilinx tools. You will be taken to the Xilinx website where you can generate a free 30-day evaluation license for Vivado Design Suite HL System Edition. After generation, the license file is sent to your e-mail. Once you have saved this file to your machine, click on the Load License selection to copy your .lic file to a default location.

Figure 1.20

The program will ask you to sign in with your pre-registered user information, including e-mail address and password.

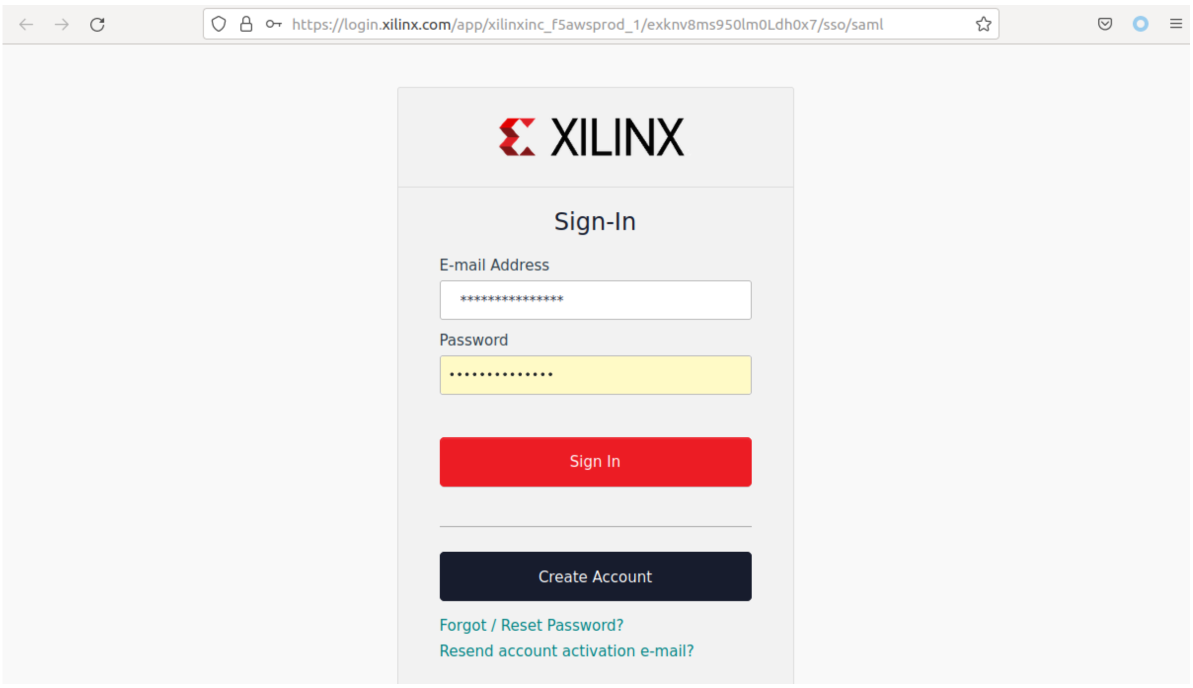
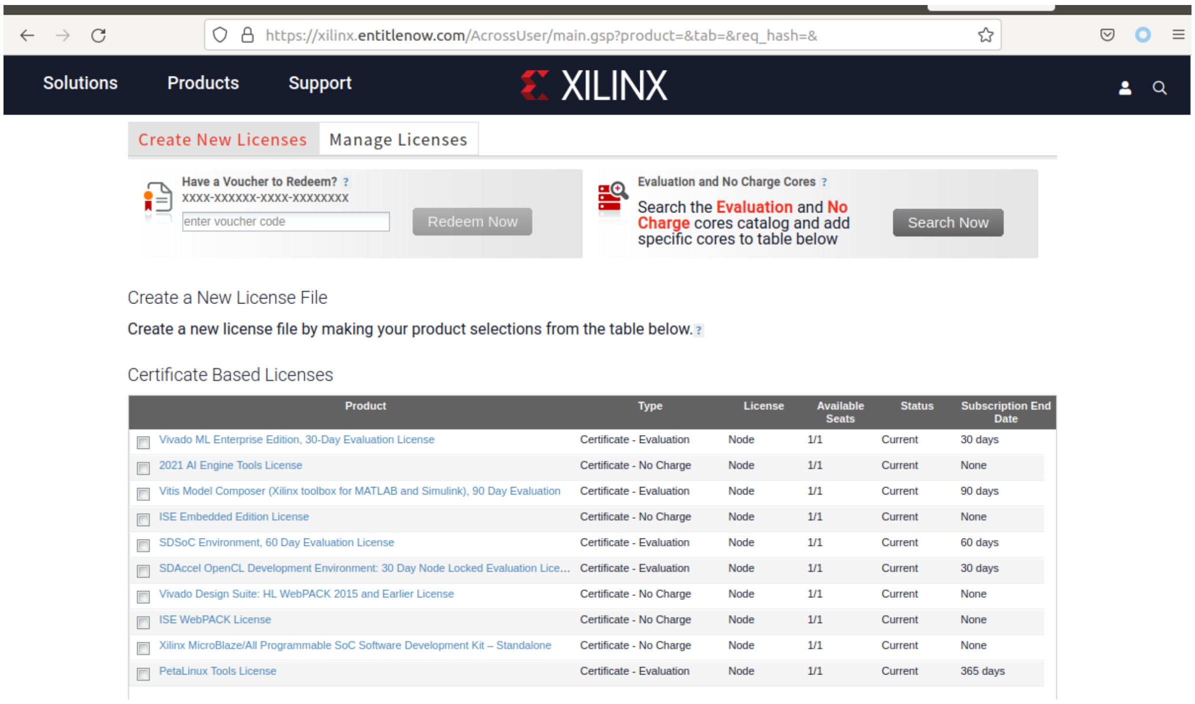

https://login.xilinx.com/app/xilinxinc_f5awsprod_1/exknv8ms950lm0Ldh0x7/sso/saml
XILINX
Sign-In
E-mail Address
Password
Sign In
Create Account
Forgot / Reset Password?
Resend account activation e-mail?

Figure 1.21

After successful login or signup, you will see a list of certificate based licenses, shown in Figure . Find the Vivado HLS Evalution Evalution, select it and confirm the generation. The license with file name **Xilinx.lic** will be downloaded to you local computer.


https://xilinx.entitlenow.com/AcrossUser/main.gsp?product=&tab=&req_hash=&
Solutions
Products
Support
XILINX
Create New Licenses
Manage Licenses
Have a Voucher to Redeem? ?
XXXX-XXXXXX-XXXX-XXXXXXXX
enter voucher code
Redeem Now
Evaluation and No Charge Cores ?
Search the Evaluation and No Charge cores catalog and add specific cores to table below
Search Now

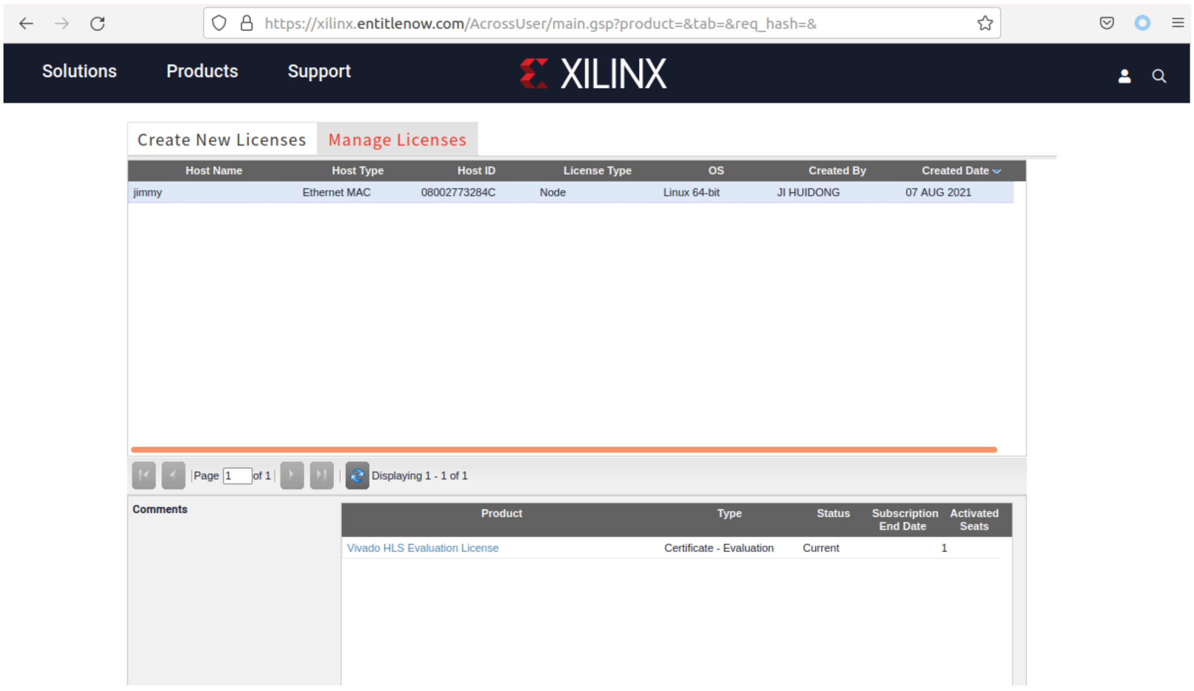
Create a New License File

Create a new license file by making your product selections from the table below. ?

Certificate Based Licenses

| Product | Type | License | Available Seats | Status | Subscription End Date |
|---|---|---|---|---|---|
| Vivado ML Enterprise Edition, 30-Day Evaluation License | Certificate - Evaluation | Node | 1/1 | Current | 30 days |
| 2021 AI Engine Tools License | Certificate - No Charge | Node | 1/1 | Current | None |
| Vitis Model Composer (Xilinx toolbox for MATLAB and Simulink), 90 Day Evaluation | Certificate - Evaluation | Node | 1/1 | Current | 90 days |
| ISE Embedded Edition License | Certificate - No Charge | Node | 1/1 | Current | None |
| SDSoC Environment, 60 Day Evaluation License | Certificate - Evaluation | Node | 1/1 | Current | 60 days |
| SDAccel OpenCL Development Environment: 30 Day Node Locked Evaluation Lice... | Certificate - Evaluation | Node | 1/1 | Current | 30 days |
| Vivado Design Suite: HL WebPACK 2015 and Earlier License | Certificate - No Charge | Node | 1/1 | Current | None |
| ISE WebPACK License | Certificate - No Charge | Node | 1/1 | Current | None |
| Xilinx MicroBlaze/All Programmable SoC Software Development Kit – Standalone | Certificate - No Charge | Node | 1/1 | Current | None |
| PetaLinux Tools License | Certificate - Evaluation | Node | 1/1 | Current | 365 days |

Figure 1.22



https://xilinx.entitlenow.com/AcrossUser/main.gsp?product=&tab=&req_hash=&

| Host Name | Host Type | Host ID | License Type | OS | Created By | Created Date |
|---|---|---|---|---|---|---|
| jimmy | Ethernet MAC | 08002773284C | Node | Linux 64-bit | JI HUIDONG | 07 AUG 2021 |

Page 1 of 1 Displaying 1 - 1 of 1

Comments

| Product | Type | Status | Subscription End Date | Activated Seats |
|---|---|---|---|---|
| Vivado HLS Evaluation License | Certificate - Evaluation | Current | | 1 |

Figure 1.23

Shown in Figure , **Load license** with the selected license file in your local machine. The License manager will verify the file to check the credential. If the verification passed, the license status window will show the license's detail.

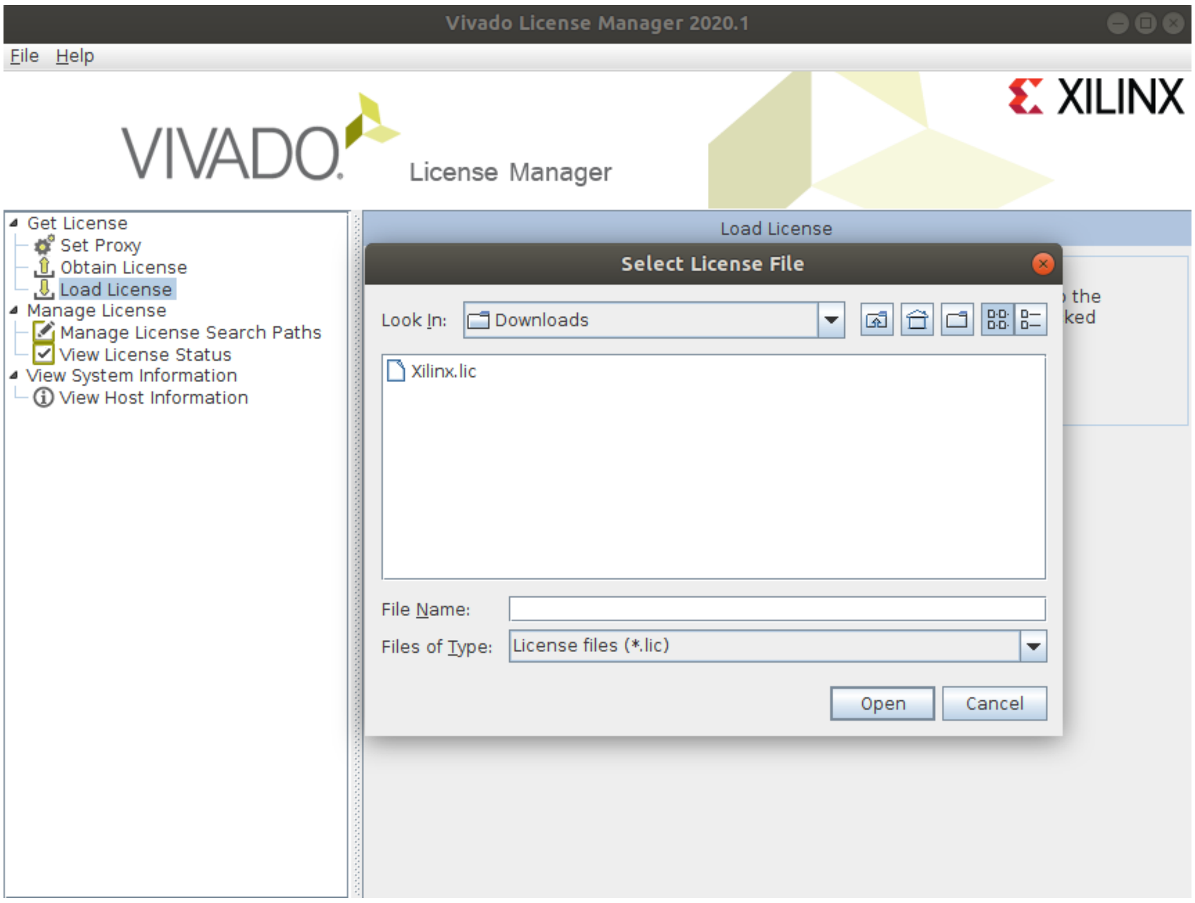
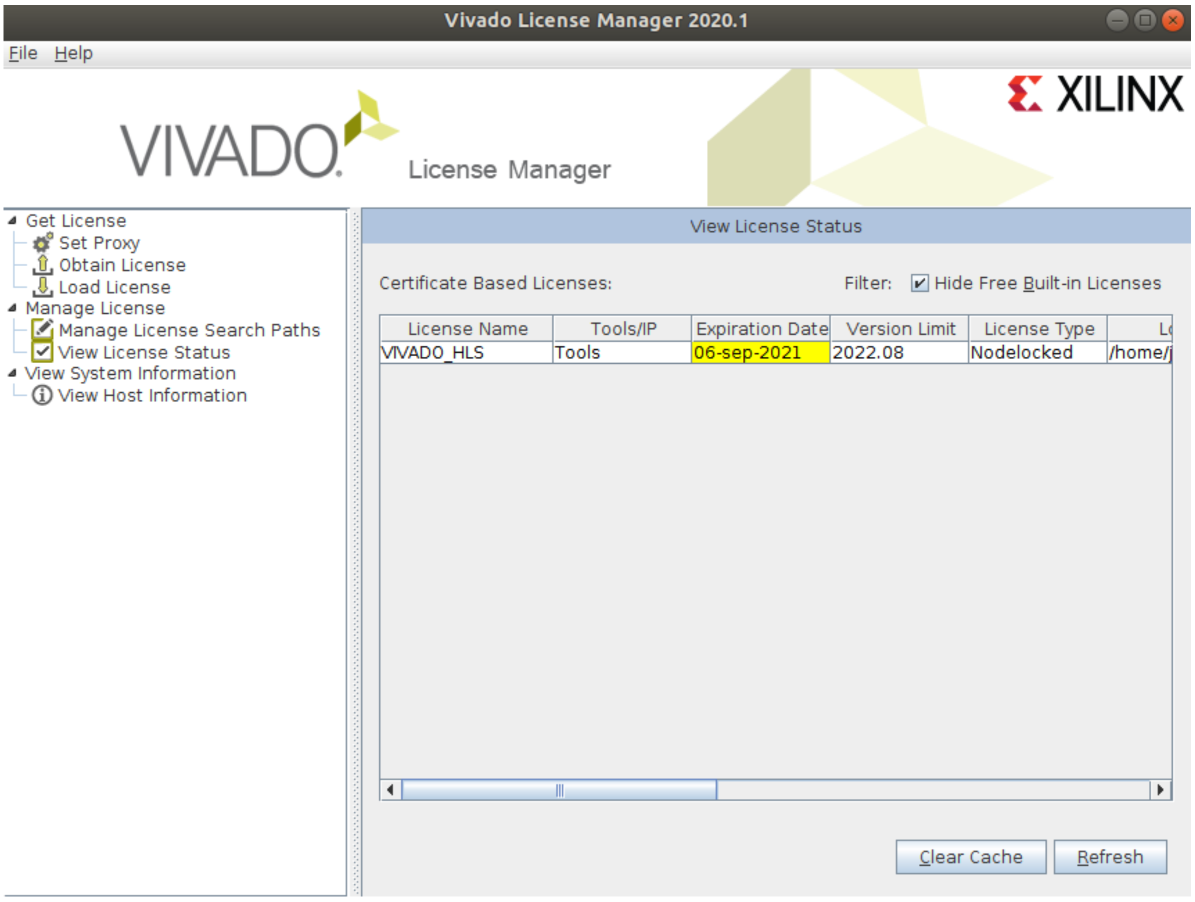


Figure 1.24



Certificate Based Licenses:

Filter: ☑ Hide Free Built-in Licenses

| License Name | Tools/IP | Expiration Date | Version Limit | License Type | Lo |
|---|---|---|---|---|---|
| VIVADO_HLS | Tools | 06-sep-2021 | 2022.08 | Nodelocked | /home/j |

Figure 1.25

Any problems concerning the installation, please feel free to contact me.

In the next lab session, we will write the first project on Vivado Design Suite, therefore, before the start of the lab, please learn yourself the basics of VHDL and Verilog.

Maybe there are students who are not familiar with Verilog and VHDL, two popular hardware description languages. Don't worry, Here are some websites helping you get start with these.

# Lab 2 Verilog & VHDL:

## 2.1 Goal

This part of the documentation gives a step by step instruction about how to create a project and verify the result. It is expected the users can learn how to create a simple digital circuit using Verilog HDL and finish the synthesis and generate the bitstream for the designated hardware.

## 2.1 Design Flow

Shown in Figure 2.1, a typical design flow consists of model(s) creating, user constraint creating file(s), Vivado project creating, the created models importing, assigning created constraint file(s), optionally running behavioral simulation, synthesizing the design, implementing the design, generating the bitstream, and finally verifying the functionality in the hardware by downloading the generated bitstream file.

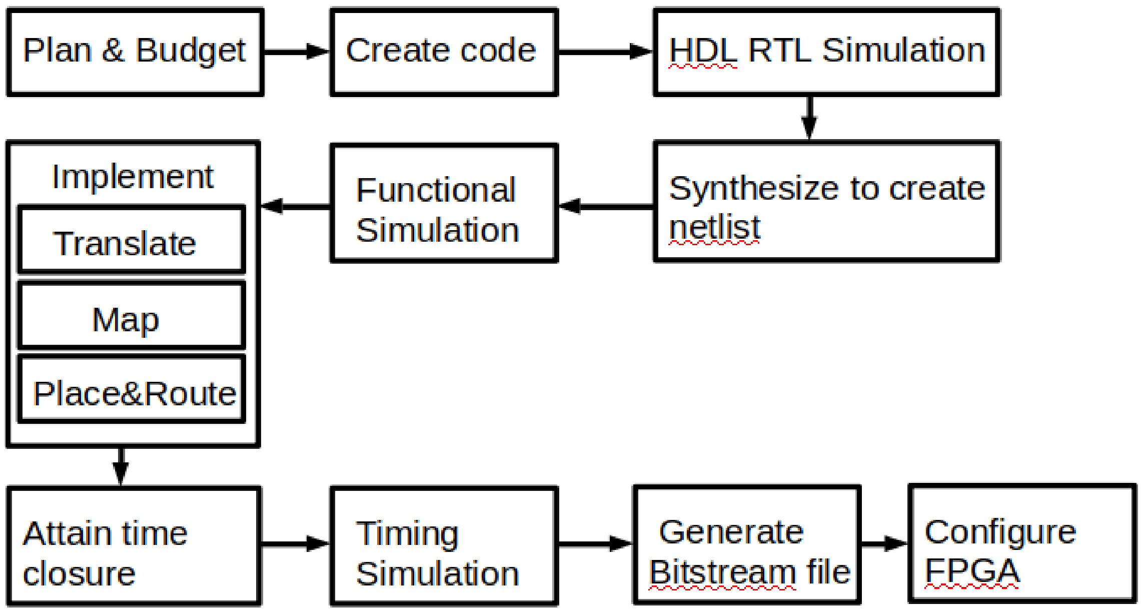
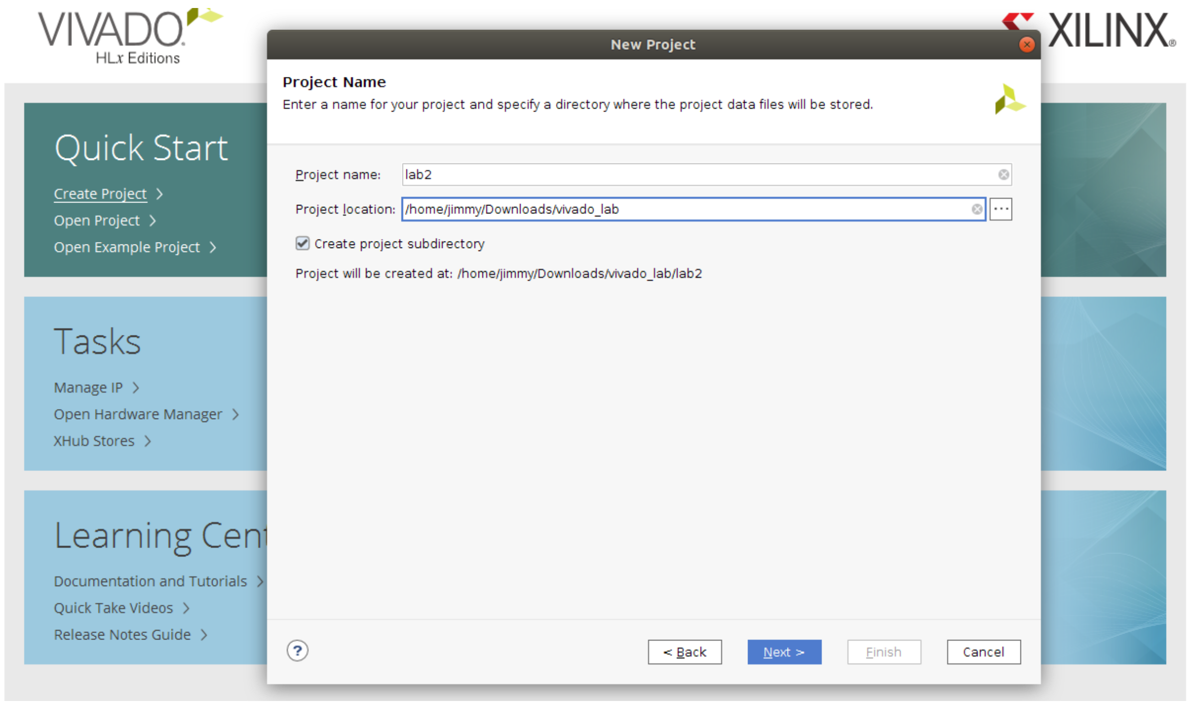


Figure 2.1

### Project creating

Figure to Figure is the new project, lab2's creating process. You can locate the project on the **vivado_lab** folder, which is provided and can be downloaded from the github.


New Project

**Project Name**

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

Project name: lab2

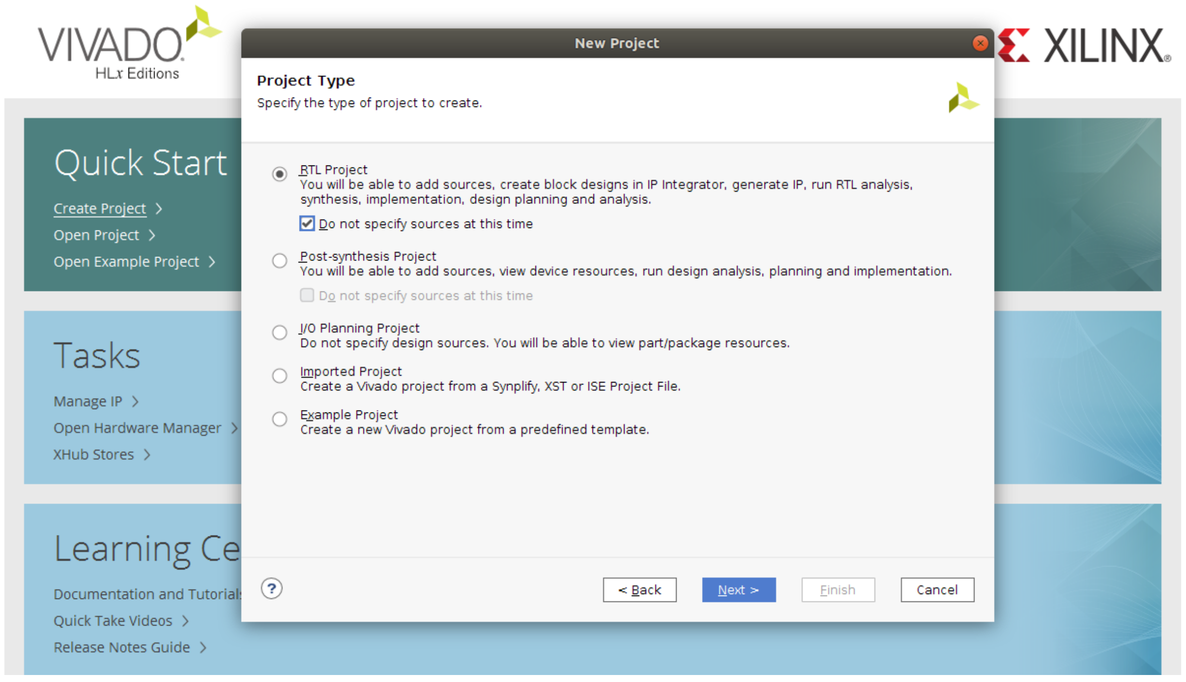
Project location: /home/jimmy/Downloads/vivado_lab

[✓] Create project subdirectory

Project will be created at: /home/jimmy/Downloads/vivado_lab/lab2

Figure 2.2

Choose RTL project as the project type.


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

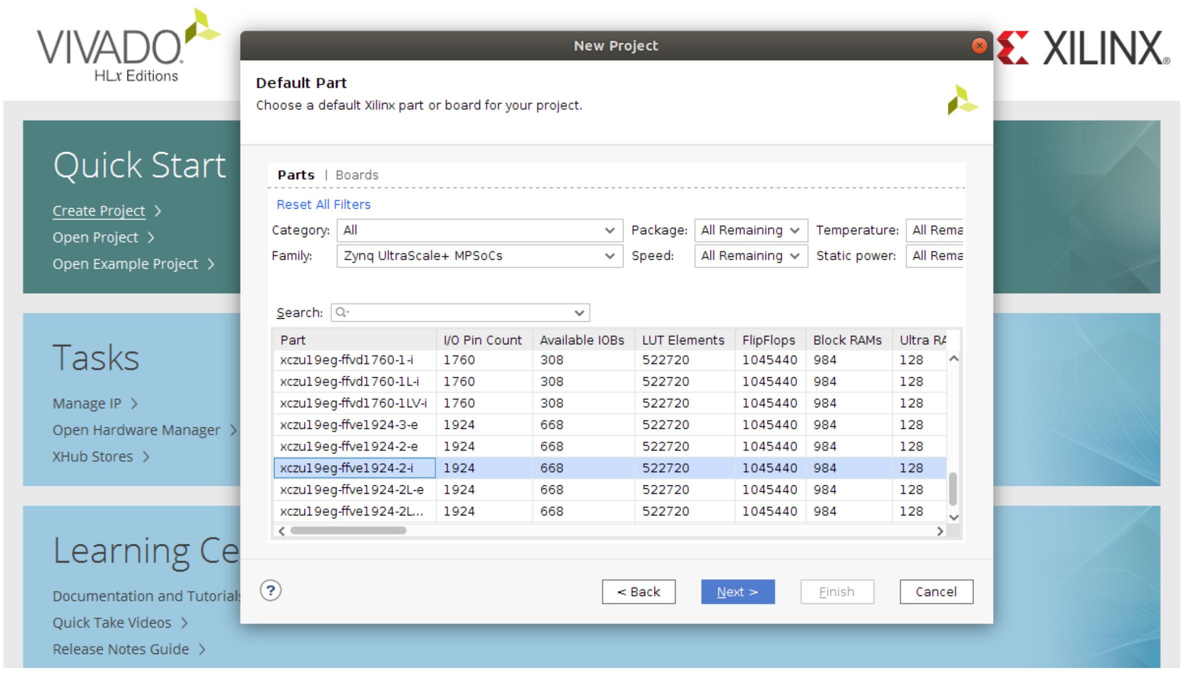
○ I/O Planning Project
Do not specify design sources. You will be able to view part/package resources.

○ 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.

Figure 2.3

After the type selection, decide the board part.



**Default Part**

Choose a default Xilinx part or board for your project.

Category: All — Package: All Remaining — Temperature: All Rema

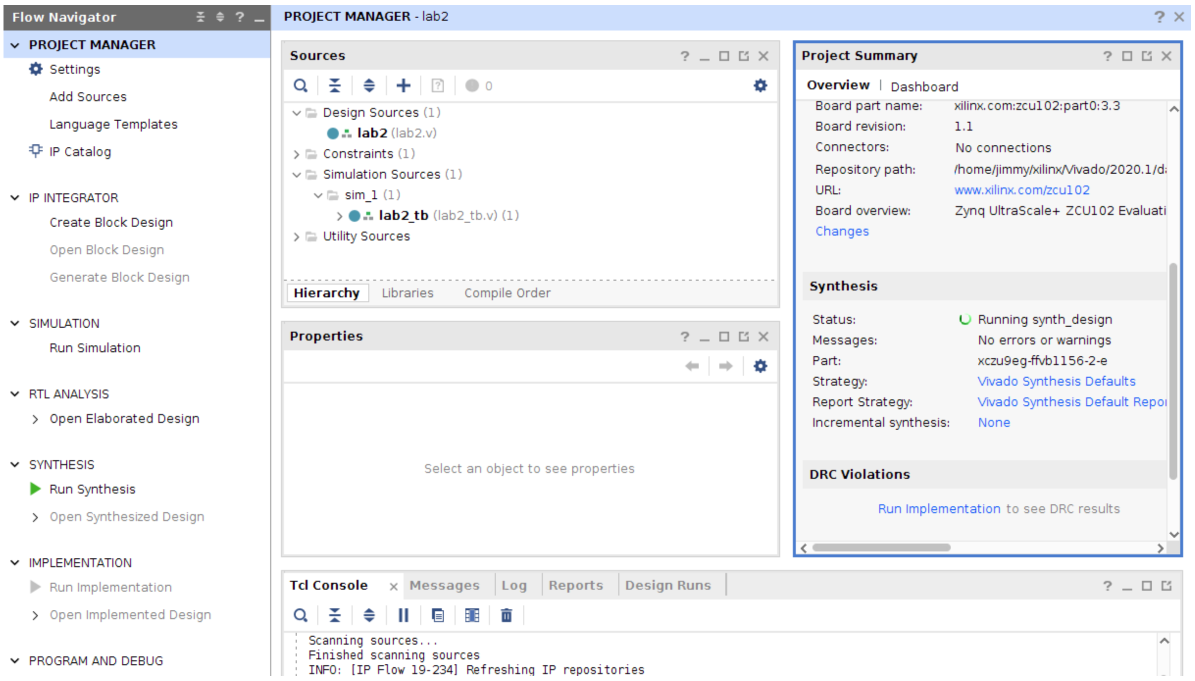
Family: Zynq UltraScale+ MPSoCs — Speed: All Remaining — Static power: All Rema

Search:

| Part | I/O Pin Count | Available IOBs | LUT Elements | FlipFlops | Block RAMs | Ultra RA |
|---|---|---|---|---|---|---|
| xczu19eg-ffvd1760-1-i | 1760 | 308 | 522720 | 1045440 | 984 | 128 |
| xczu19eg-ffvd1760-1L-i | 1760 | 308 | 522720 | 1045440 | 984 | 128 |
| xczu19eg-ffvd1760-1LV-i | 1760 | 308 | 522720 | 1045440 | 984 | 128 |
| xczu19eg-ffve1924-3-e | 1924 | 668 | 522720 | 1045440 | 984 | 128 |
| xczu19eg-ffve1924-2-e | 1924 | 668 | 522720 | 1045440 | 984 | 128 |
| xczu19eg-ffve1924-2-i | 1924 | 668 | 522720 | 1045440 | 984 | 128 |
| xczu19eg-ffve1924-2L-e | 1924 | 668 | 522720 | 1045440 | 984 | 128 |
| xczu19eg-ffve1924-2L... | 1924 | 668 | 522720 | 1045440 | 984 | 128 |

Figure 2.4

We can see the layout of the project platform and the sources hierarchy.



Sources:
- Design Sources (1)
  - lab2 (lab2.v)
- Constraints (1)
- Simulation Sources (1)
  - sim_1 (1)
    - lab2_tb (lab2_tb.v) (1)
- Utility Sources

Properties: Select an object to see properties

Project Summary

| | |
|---|---|
| Board part name: | xilinx.com:zcu102:part0:3.3 |
| Board revision: | 1.1 |
| Connectors: | No connections |
| Repository path: | /home/jimmy/xilinx/Vivado/2020.1/d |
| URL: | www.xilinx.com/zcu102 |
| Board overview: | Zynq UltraScale+ ZCU102 Evaluati |

Changes

**Synthesis**

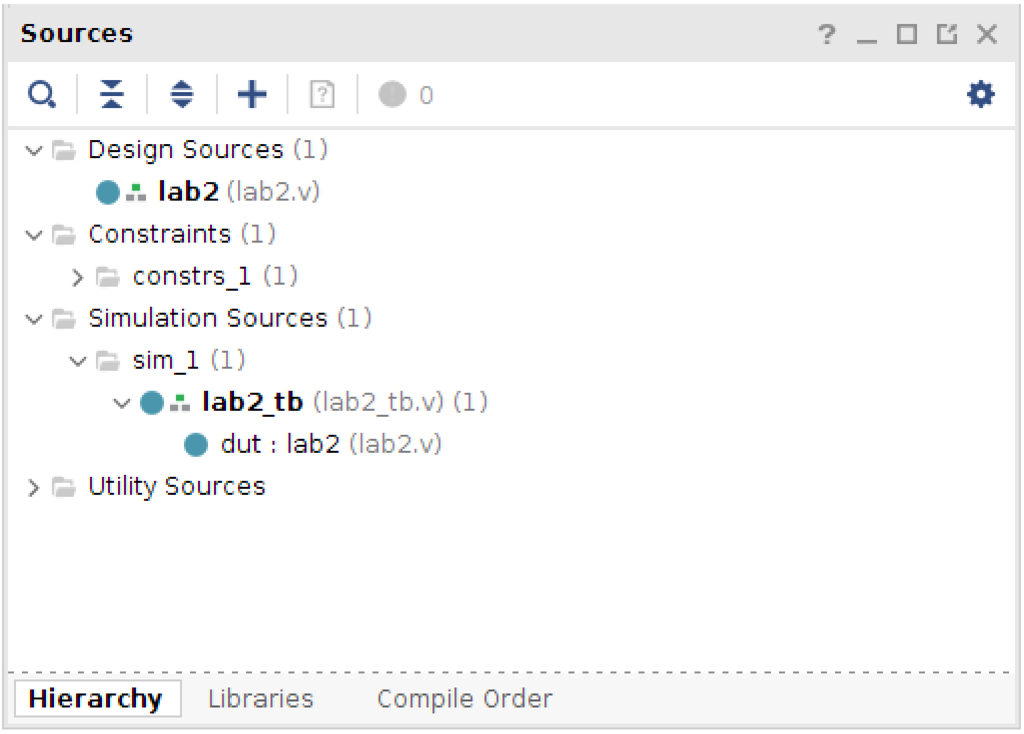
| | |
|---|---|
| Status: | Running synth_design |
| Messages: | No errors or warnings |
| Part: | xczu9eg-ffvb1156-2-e |
| Strategy: | Vivado Synthesis Defaults |
| Report Strategy: | Vivado Synthesis Default Repo |
| Incremental synthesis: | None |

**DRC Violations**

Run Implementation to see DRC results

```
Scanning sources...
Finished scanning sources
INFO: [IP_Flow 19-234] Refreshing IP repositories
```

Figure 2.5



Sources

Design Sources (1)
- lab2 (lab2.v)

Constraints (1)
- constrs_1 (1)

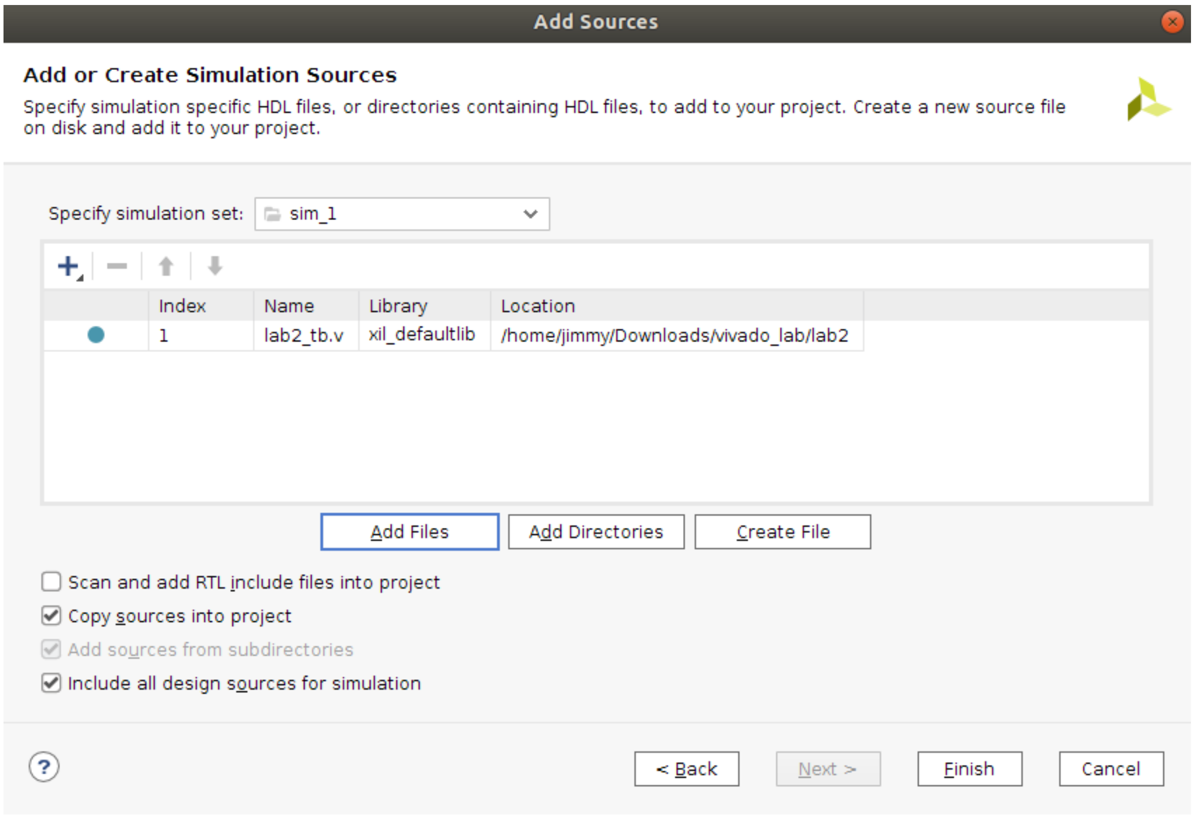
Simulation Sources (1)
- sim_1 (1)
  - lab2_tb (lab2_tb.v) (1)
    - dut : lab2 (lab2.v)

Utility Sources

Hierarchy | Libraries | Compile Order

Figure 2.6

## Source Adding



Add Sources

**Add or Create Simulation Sources**

Specify simulation specific HDL files, or directories containing HDL files, to add to your project. Create a new source file on disk and add it to your project.

Specify simulation set: sim_1

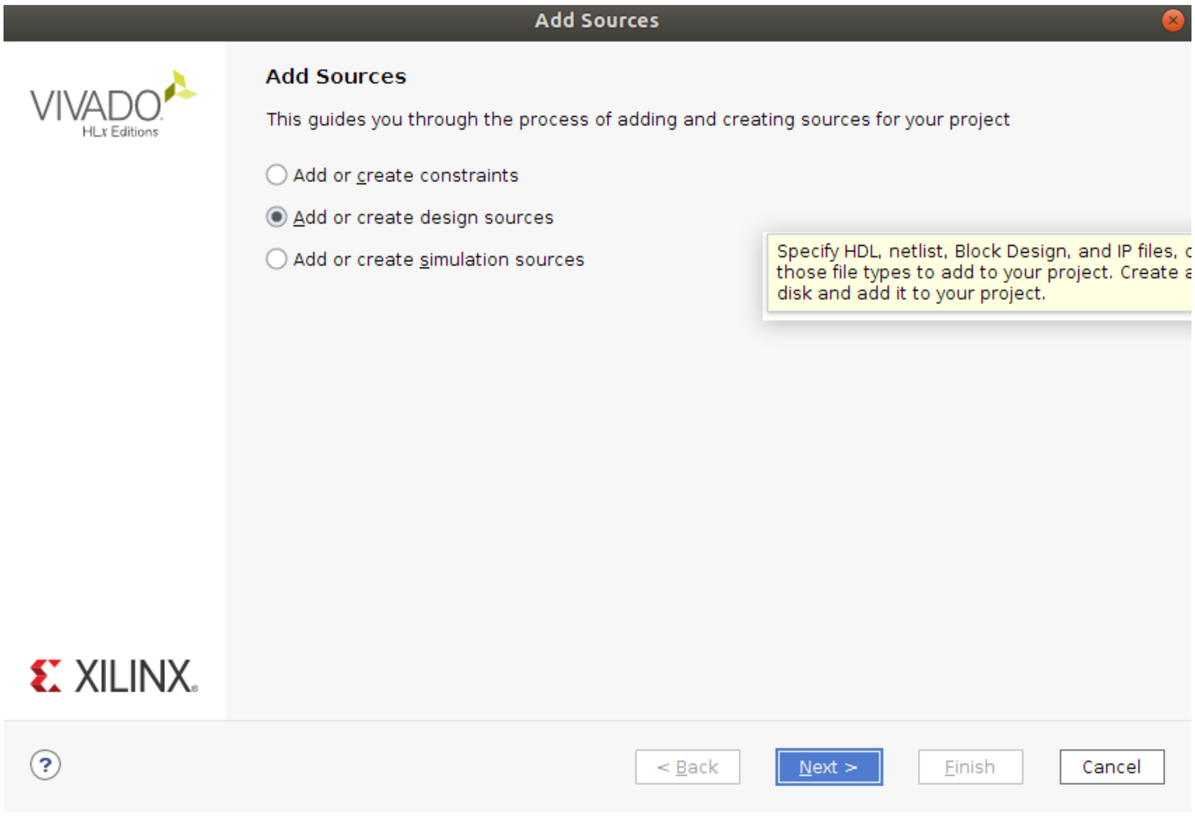
| | Index | Name | Library | Location |
|---|---|---|---|---|
| ● | 1 | lab2_tb.v | xil_defaultlib | /home/jimmy/Downloads/vivado_lab/lab2 |

Add Files | Add Directories | Create File

- [ ] Scan and add RTL include files into project
- [x] Copy sources into project
- [x] Add sources from subdirectories
- [x] Include all design sources for simulation

< Back | Next > | Finish | Cancel

Figure 2.7



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

Specify HDL, netlist, Block Design, and IP files, o
those file types to add to your project. Create a
disk and add it to your project.

Figure 2.8

In the **source** window, we can find the verilog code for lab2 project. double click it and check the code, which is also pasted below.

```
`timescale 1ns / 1ps
////////////////////////////////////////////////////////////////////////////////
/
// Module Name: lab2
////////////////////////////////////////////////////////////////////////////////
/


module lab2(
    input [7:0] swt,
    output [7:0] led
    );

    assign led[0] = ~swt[0];
    assign led[1] = swt[1] & ~swt[2];
    assign led[3] = swt[2] & swt[3];
    assign led[2] = led[1] | led[3];

    assign led[7:4] = swt[7:4];

endmodule
```

1. The first line `` `timescale 1ns / 1ps `` defines the timescale directive specifies the time unit and precision for the modules for the later functional and timing simulaton. Please note: it has no effect on the **synthesis** and also **hardware implementation**.

```
module lab2()
   /*main content*/
endmodule
```

2. The `module` and `endmodule` defines the beginning and end of a module.

```
input [7:0] swt,
output [7:0] led
```

3. The input and output define the I/O port of the module, `[7:0]` is the length of data array passing through the port, for example `0x00101011`.
4. The follow part defines the logical relationship between the input **swt** and the output **led**. The first bit of the output,**led[0]**, is linked with the value of the first bit of the input, **swt[0]** with **assign**. `~` in the verilog code is the bitwise operator for negation.
5. `&` and `|` are also the bitwise operator working as AND and OR logics.

## RTL Analysis

After the code detail explanation, we can get a functional circuit with the output array of **led** control by the input **swt**. To get out of the abstract description, we can use RTL analysis to get the schematic drawing for the logic circuit, shown in Figure 2.9.

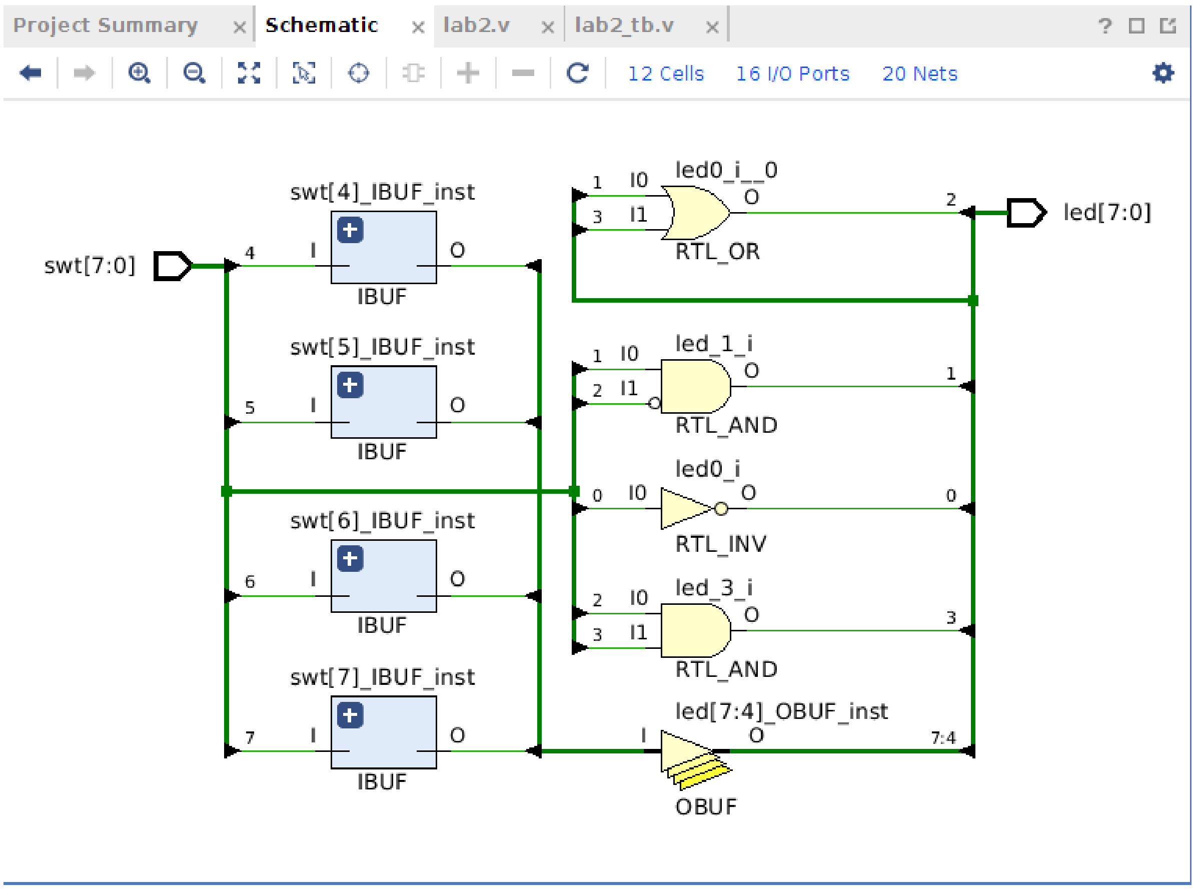


Figure 2.9

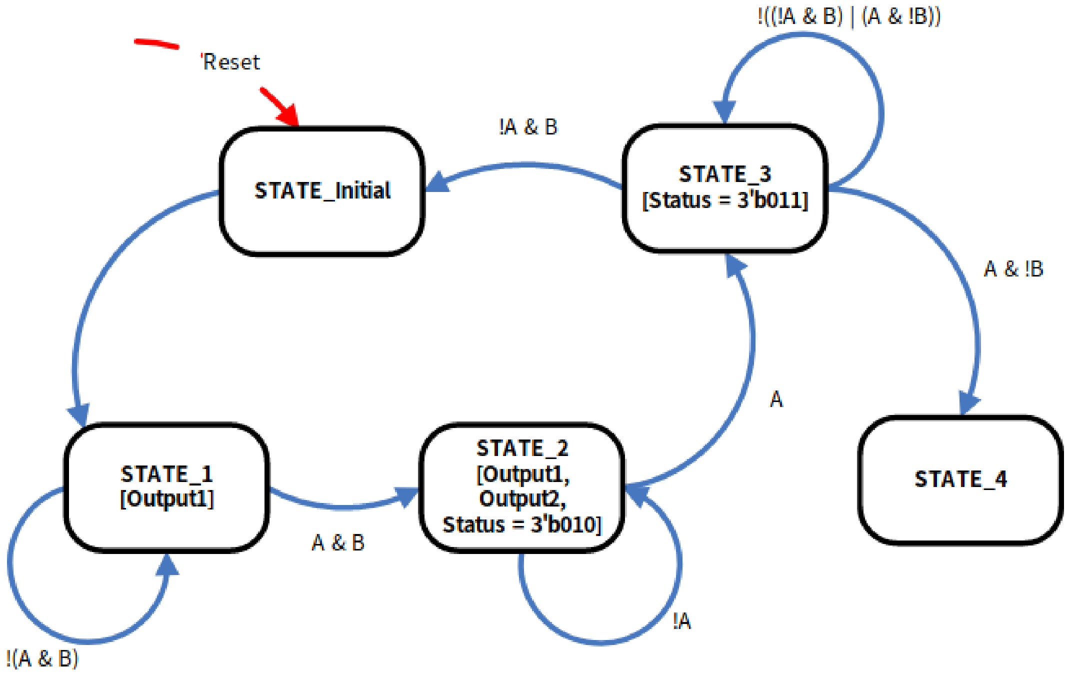
# 2.3 Finite State Machine

Generally circuits can be divided into two modes, combinational and sequential mode. In combinational circuits, the output depends on the current values of inputs only. Whereas in sequential circuits, the output depends on the current values of the inputs along with the previously stored states.

Shown in Figure 2.10, the FSM consists of:

1. Initial state (**STATE_Initial**).
2. States with conditional outward transitions(**STATE_1/STATE_2/STATE_3**).
3. States with non-conditional outward transitions(**STATE_Initial**).
4. States looping back into themselves(**STATE_1/STATE_2/STATE_3**).
5. States with no outward transitions(**STATE_4**).

All the individual state in some cases share more than one characteristics.



| | |
|---|---|
| & | logical and |
| \| | logical or |
| ! | logical not |

Figure 2.10

To construct the logics in verilog, we need to:

1. A state representation for each state.
2. A mechanism for keeping track of the current state.
3. Transitions or jumps from state to state.
4. Output values based on the current state.

Below is the template code for your reference.

```
module BasicFsm(
//-------Inputs & Outputs declaration------------------------
 input wire Clock ,
 input wire Reset ,
 input wire A,
 input wire B,
 output wire Output1 ,
 output wire Output2 ,
 output reg [2:0] Status
// ------------------------------------------------------------
);
//-------- State Encoding-------------------------------------
localparam STATE_Initial = 3’d0 ,
           STATE_1 = 3’d1 ,
           STATE_2 = 3’d2 ,
           STATE_3 = 3’d3 ,
           STATE_4 = 3’d4 ,
           STATE_5_PlaceHolder = 3’d5 ,
           STATE_6_PlaceHolder = 3’d6 ,
           STATE_7_PlaceHolder = 3’d7;
// --------------------------------------------------------------------
// --------------------State reg Declarations----------------------
reg [2:0] CurrentState;
reg [2:0] NextState;
// --------------------------------------------------------------------
// ---------------------Outputs----------------------------------------
// ------------1-bit outputs-------------------------------------------
assign Output1 = (CurrentState == STATE_1) | (CurrentState == STATE_2);
assign Output2 = (CurrentState == STATE_2);
// ------------multi -bit outputs--------------------------------------
always@( * ) begin
   Status = 3’b000;
   case (CurrentState)
     STATE_2: begin
       Status = 3’b010;
     end
     STATE_3: begin
       Status = 3’b011;
     end
   endcase
   end
// --------------------------------------------------------------------
// Synchronous State -Transition always@(posedge Clock) block
always@(posedge Clock) begin
  if (Reset) CurrentState <= STATE_Initial;
  else CurrentState <= NextState;
  end
// --------------------------------------------------------------------
// Conditional State -Transition always@( * ) block
always@( * ) begin
  NextState = CurrentState;
  case (CurrentState)
    STATE_Initial: begin
      NextState = STATE_1;
    end
    STATE_1: begin
      if (A & B) NextState = STATE_2;
    end
    STATE_2: begin
      if (A) NextState = STATE_3;
    end
    STATE_3: begin
      if (!A & B) NextState = STATE_Initial;
```

```
        else if (A & !B) NextState = STATE_4;
      end
      STATE_4: begin
      end
      STATE_5_PlaceHolder: begin
        NextState = STATE_Initial;
      end
      STATE_6_PlaceHolder: begin
        NextState = STATE_Initial;
      end
      STATE_7_PlaceHolder: begin
        NextState = STATE_Initial;
      end
    endcase
  end
// --------------------------------------------------------------------
endmodule
```

In the code, the declaration part describes the output variables and input variables, which correcponds to the flow diagram of the FSM.

To output the value of register **Status**, the program implements **case** statement to accomplish the conditional selection. Also the same method of **case** statement applies to the state regsiter value of **NextState**.

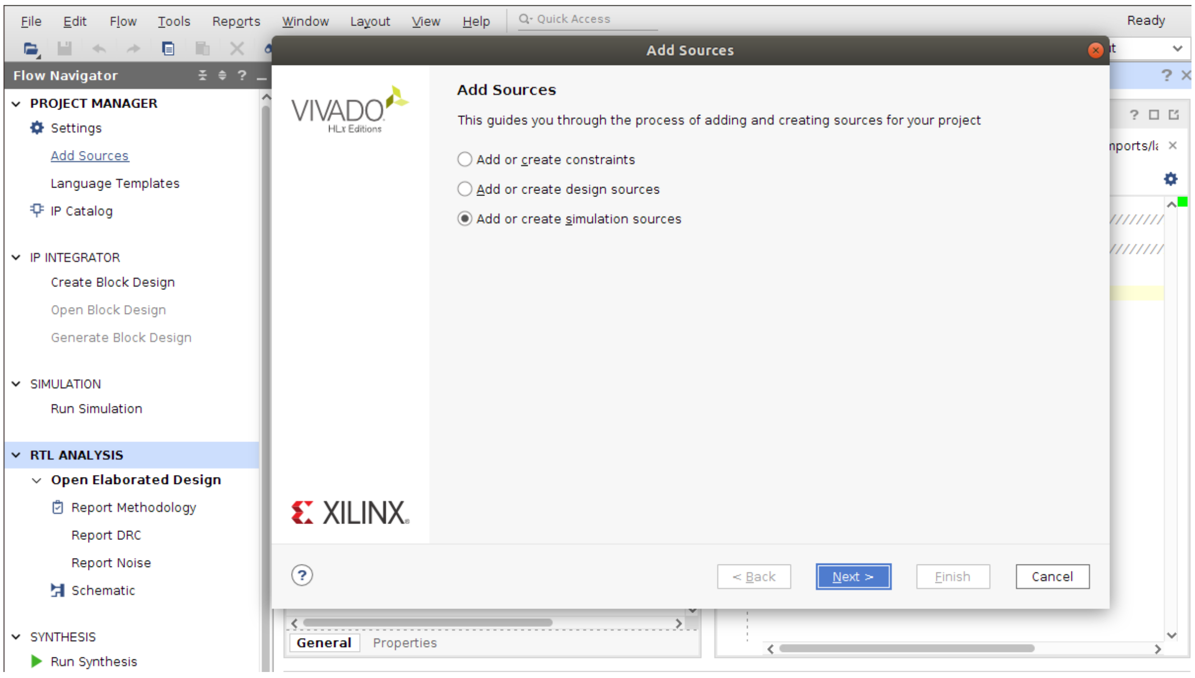
In Synchronous State part, the value of **CurrentState** is determined only by the input **reset** value.

## 2.4 Testbench

To validate the performance of the module, a testbench script file is needed.

Testbench is a program or model written in any language for the purposes of exercising and verifying the functional correctness of a hardware model during the simulation.

Add the testbench verilog file into the source as the create simulation sources.



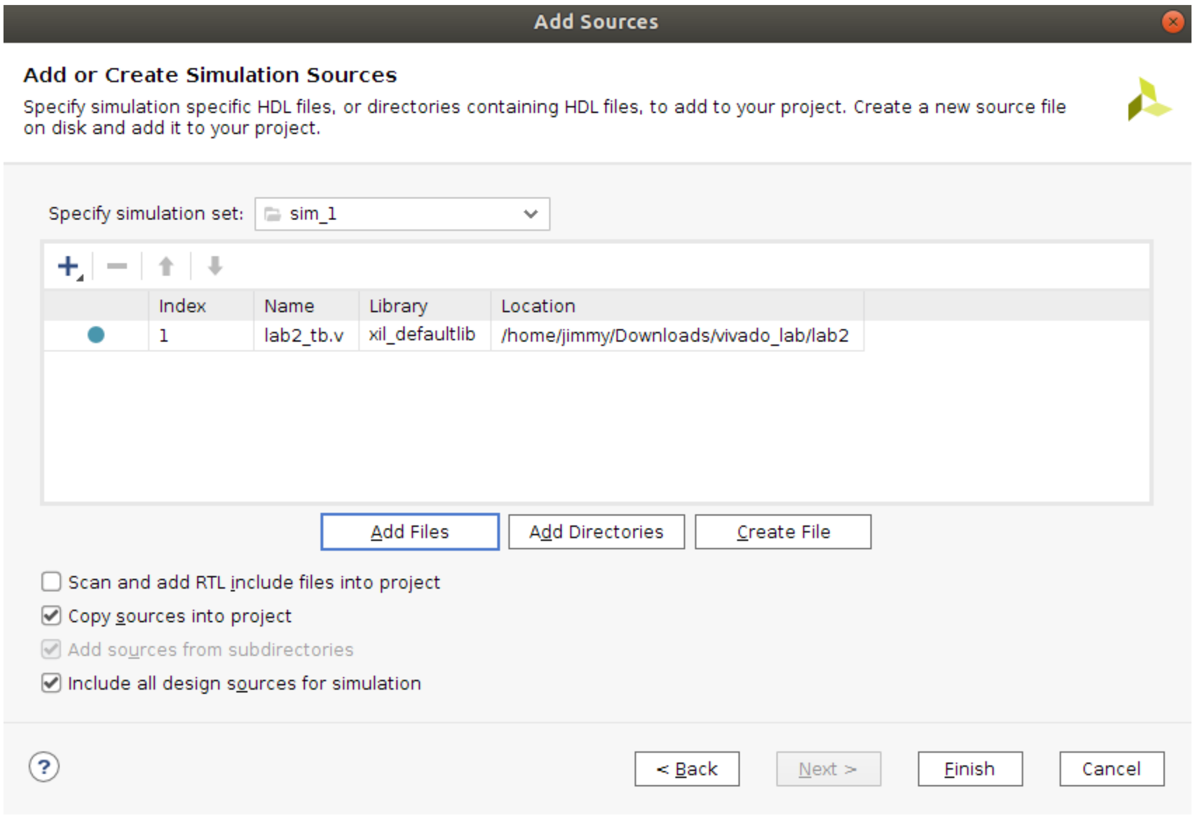
Add Sources

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]

Figure 2.11



**Add or Create Simulation Sources**

Specify simulation specific HDL files, or directories containing HDL files, to add to your project. Create a new source file on disk and add it to your project.

Specify simulation set: sim_1

| | Index | Name | Library | Location |
|---|---|---|---|---|
| ● | 1 | lab2_tb.v | xil_defaultlib | /home/jimmy/Downloads/vivado_lab/lab2 |

Add Files

☐ Scan and add RTL include files into project
☑ Copy sources into project
☑ Add sources from subdirectories
☑ Include all design sources for simulation

Figure 2.12

Below is the code block for your reference.

```
`timescale 1ns / 1ps
/////////////////////////////////////////////////////////////////
// Module Name: lab1_tb
/////////////////////////////////////////////////////////////////
module lab2_tb(

    );

    reg [7:0] switches;
    wire [7:0] leds;
    reg [7:0] e_led;

    integer i;

    lab2 dut(.led(leds),.swt(switches));

    function [7:0] expected_led;
       input [7:0] swt;
    begin
       expected_led[0] = ~swt[0];
       expected_led[1] = swt[1] & ~swt[2];
       expected_led[3] = swt[2] & swt[3];
       expected_led[2] = expected_led[1] | expected_led[3];
       expected_led[7:4] = swt[7:4];
    end
    endfunction

    initial
    begin
        for (i=0; i < 255; i=i+2)
        begin
            #50 switches=i;
            #10 e_led = expected_led(switches);
            if(leds == e_led)
                $display("LED output matched at", $time);
            else
                $display("LED output mis-matched at ",$time,": expected: %b, 
actual: %b", e_led, leds);
        end
    end

endmodule
```

## Timescale definition and variable declaration

```
`timescale 1ns / 1ps

reg [7:0] switches;
wire [7:0] leds;
reg [7:0] e_led;
integer i;
```

Comparing with the functional model module, the Testbench module usually has no port. And in the first part, the Testbench defines the variables for future references and assignments.

## DUT Instantiating

```
lab2 dut(.led(leds),.swt(switches));
```

Following the variables' definition, the Testbench instantiating a DUT(Design Under Design), which is the duplicates of the functional module waiting to be tested(in our project, the function module is **lab2**). and DUT regulates that the input variable must be **reg** type, and the output the **wire** type, because the **reg** type can be assigned with value, while **wire** type can not.

## Generating clock

```
initial
    begin
        for (i=0; i < 255; i=i+2)
        begin
            #50 switches=i;
            #10 e_led = expected_led(switches);
            if(leds == e_led)
                $display("LED output matched at", $time);
            else
                $display("LED output mis-matched at ",$time,": expected: %b, 
actual: %b", e_led, leds);
        end
    end
```

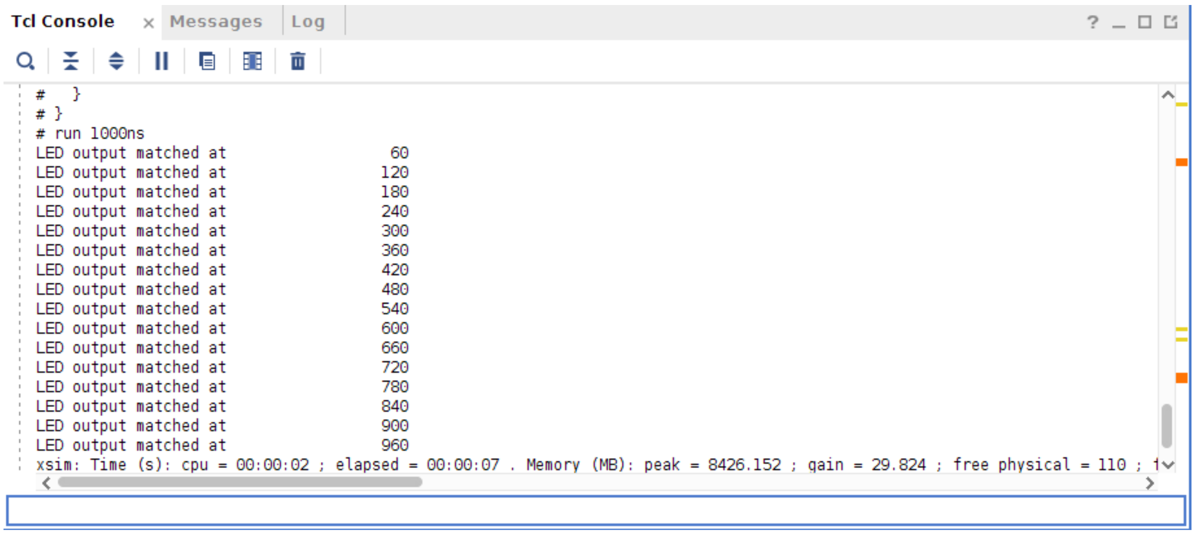
This Testbench program uses a for loop to counter the frequency or clocks of running. `#50` and `#10` are the delay controls the for-loop used to generate the signal.

In between the clock generating and DUT instantiating, the program starts another function block to repeat the logic of the lab2 circuit and the link the **swt** with output **expected_led**.

```
function [7:0] expected_led;
        input [7:0] swt;
    begin
        expected_led[0] = ~swt[0];
        expected_led[1] = swt[1] & ~swt[2];
        expected_led[3] = swt[2] & swt[3];
        expected_led[2] = expected_led[1] | expected_led[3];
        expected_led[7:4] = swt[7:4];
    end
    endfunction
```

The value of **expected_led** will be used to compare with the DUT output to check the correctness of the lab2 function. The result will be published with `$display()` function on the **Tcl concole**.

## Simulation



```
#   }
# }
# run 1000ns
LED output matched at                  60
LED output matched at                 120
LED output matched at                 180
LED output matched at                 240
LED output matched at                 300
LED output matched at                 360
LED output matched at                 420
LED output matched at                 480
LED output matched at                 540
LED output matched at                 600
LED output matched at                 660
LED output matched at                 720
LED output matched at                 780
LED output matched at                 840
LED output matched at                 900
LED output matched at                 960
xsim: Time (s): cpu = 00:00:02 ; elapsed = 00:00:07 . Memory (MB): peak = 8426.152 ; gain = 29.824 ; free physical = 110 ; f
```

Figure 2.13

To verify the logic function's correctness, the waveform generator from simulation also can provide the result,shown in Figure 2.14.

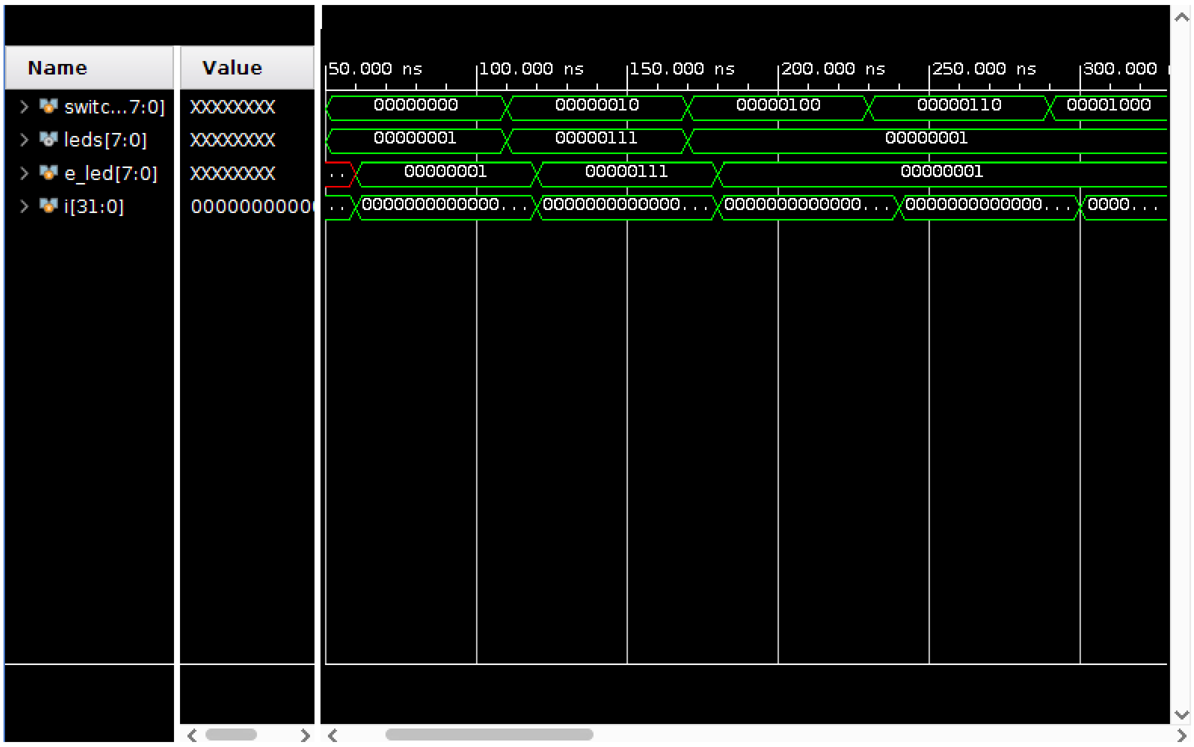


Figure 2.14

## 2.5 Assignment

### Assignment 1

Please explain the two types of FSM, and analyze the difference.

### Assignment 2

Please generate the waveform with Vivado Design Suite for the template code in section 2.2, and store it with the form **wcfg**.

### Assignment 3

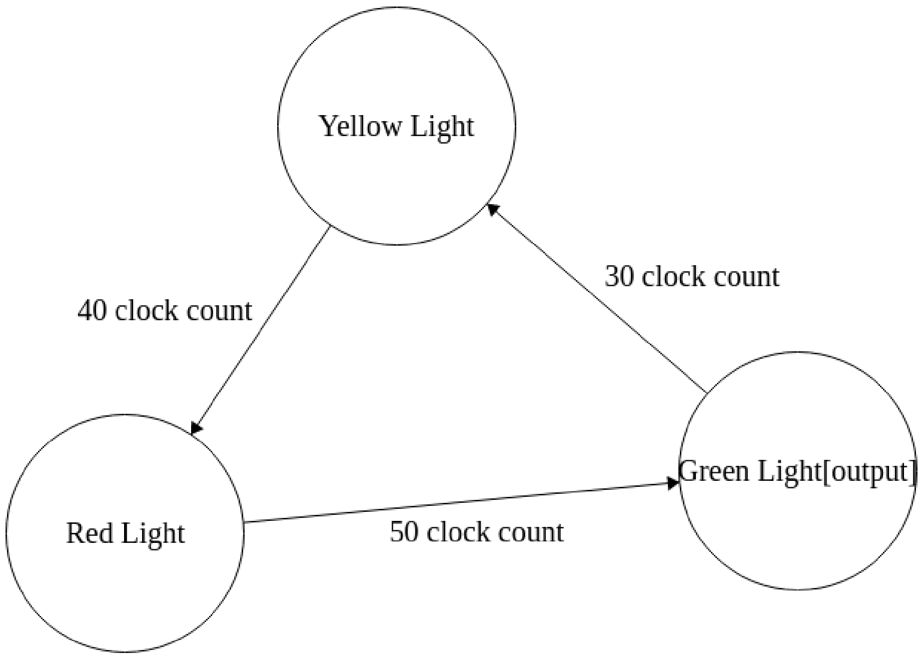


Figure 2.15

The flow diagram shows the sequential logic of the traffic lights, red,yello,green. The transisition from one light to another depends on the time delay, such as the 40 clock count. When the light turns green, the output will turn 1.

In this assignment, you are required to:

1. write a verilog module using FSM to realize the traffic light system, where the Yellow Light is the initial state.
2. write a verilog testbench to check the correctness of the module, generate the waveform with Vivado Design Suite, and store it with the form **wcfg**.

# Lab 3 Convolutional Layer

## 3.1 Goal

Design a module functioning as a layer,which operates convolution on the input data array.

## 3.2 Introduction

### What is convolution?

Convolution is function that outputs an integral that expresses the amount of **overlap** of one function *g* as it is shifted over another function *f*. It therefore "blends" one function with another.

#### Linear formula

In mathematics, a convolution is defined as a product of functions *f* and *g* that are objects in the algebra of Schwartz functions in $\mathbb{R}^n$. Convolution of two functions *f(t)* and *g(t)* over a finite range $[0, t]$ is given by

$$f * g = \int_{-\infty}^{+\infty} f(\tau)g(t-\tau)d\tau$$
$$= \int_{-\infty}^{+\infty} g(\tau)f(t-\tau)d\tau$$

***Formula 1***

And the convolution $f * g$ can also be expressed as $f \otimes g$ .

#### Discrete formula

In many engineering fields, such as the computer science, the input signals come with the discrete form. Convolution of two discrete functions *x(n)* and *h(n)* over a finite count *i* is given by

$$x * h = \sum_{i=-\infty}^{\infty} x(i)h(n-i)$$
$$= \sum_{i=-\infty}^{\infty} h(i)f(n-i)$$

***Formula 2***

#### 2-Dimensional formula

The 2-D Convolution computes the two-dimensional convolution of two input 2-D arrays, in another word, matrix .  Assume that *matrix A* has dimensions $(M_a, N_a)$and matrix B has dimensions $(M_b, N_b)$. When the convolution calculates the full output size, the equation for the 2-D discrete convolution is:

$$C(i,j) = \sum_{m=0}^{M_a-1} \sum_{n=0}^{N_a-1} A(m,n) * B(i-m, j-n)$$

***Formula 3***

### Convolution in Image Processing

In image processing, convolutional filtering can be used to implement algorithms such as edge detection, image sharpening, and image blurring.

With the convolution process, many important features can be extracted from the edges of an image (e.g., corners, lines, curves). In image process, Features which distinguish them from the surrounding display the color intensity change. The feature can later be used for further deep learning training or traditional computer vision object recognition and tracking.

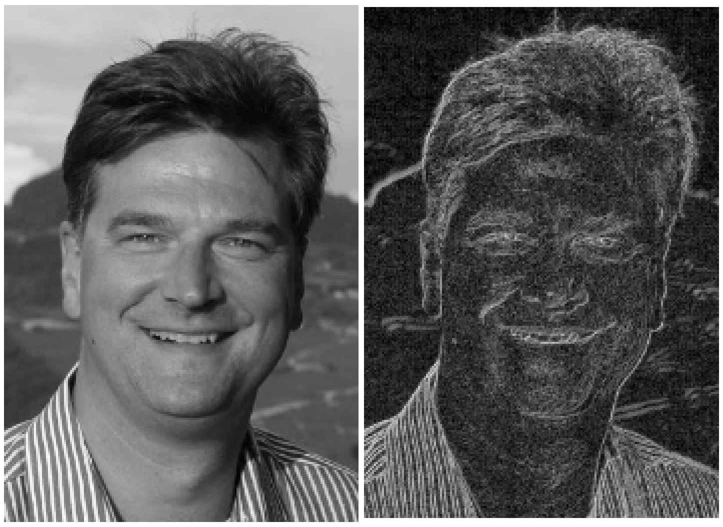


Figure 3.1

## 3.3 Assignment

### Assignment 1

In this lab, we plan to build with **Verilog** based on Vivado Design Suite to realize a 3 by 3 matrix's convolutional operation.

Imagine ***Matrix A*** is the pixel value for the upper left corner

$$A = \begin{bmatrix} 3 & 3 & 1 \\ 0 & 1 & 0 \\ 1 & 4 & 2 \end{bmatrix}$$

Matrix B is the image kernel for convolution

$$B = \begin{bmatrix} 3 & 1 & 2 \\ 2 & 0 & 2 \\ 1 & 1 & 2 \end{bmatrix}$$

Following the *Formula 3* above, $A * B$ can be separated into two parts:

1. Element-wise Multiplication
   $M11 = A_{11} * B_{11}, M12 = A_{12} * B_{12}, M13 = A_{13} * B_{13};$
   $M21 = A_{21} * B_{21}, M22 = A_{22} * B_{22}, M23 = A_{23} * B_{23};$
   $M31 = A_{31} * B_{31}, M32 = A_{32} * B_{32}, M33 = A_{33} * B_{33};$
2. Summation of the multiplication

$$S = M_{11} + M_{12} + M_{13} + M_{21} + M_{22} + M_{23} + M_{31} + M_{32} + M_{33}$$

1. Please calculate the result
2. Please simulate the whole convolution process using the mathematical operators in hardware mode, such as the **multipliers** and **adders**. Write the verilog-based module and testbench, store the waveform.

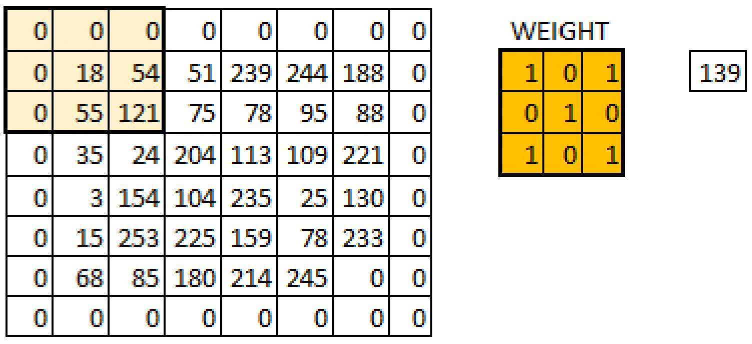
Hint

1. Vivado Design Suite has already provided functional IPs for use, such as the **RAM** or **ROM** for memory reading and writing, **Adder** for mathematical adding and **Multiplier** for multiplication. Of course you can also build your own IP module with verilog and Vivado, this is encouraged.
2. As is known in the lectures, the convolution layer in deep learning or machine learning in effect is the result of continous convolution between matrixs. To accelerate the computing, please take parallel computing in consideration, and take use of the advantage of Xilinx FPGA's reconfigurable hardware to realize it.
3. FSM(Finite State Machine) or Behavioral Tree can be used to create case-based function flow, which helps the project management.

## Assignment 2

As shown in Ass , the matrix's convolution will reduce the size and information, which meets the requirement of subsampling.

For the input matrix's size outnumber the kernel's, the output will be a matrix also:



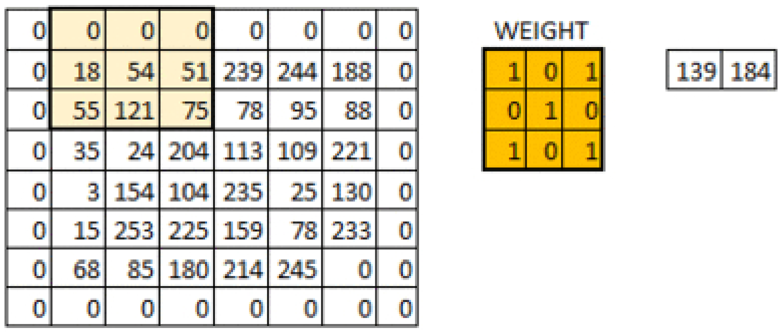
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 18 | 54 | 51 | 239 | 244 | 188 | 0 |
| 0 | 55 | 121 | 75 | 78 | 95 | 88 | 0 |
| 0 | 35 | 24 | 204 | 113 | 109 | 221 | 0 |
| 0 | 3 | 154 | 104 | 235 | 25 | 130 | 0 |
| 0 | 15 | 253 | 225 | 159 | 78 | 233 | 0 |
| 0 | 68 | 85 | 180 | 214 | 245 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

WEIGHT

| | | |
|---|---|---|
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

| |
|---|
| 139 |

**Step 1**



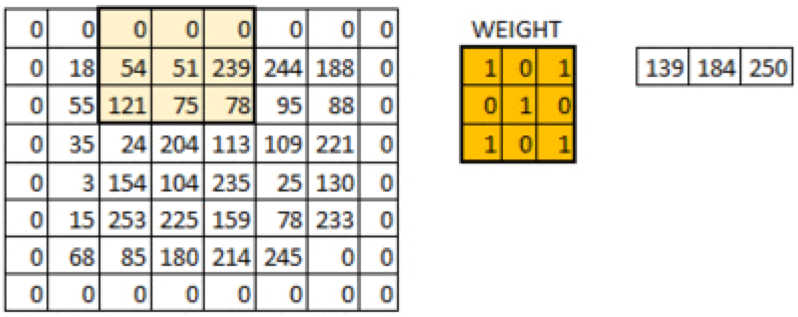
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 18 | 54 | 51 | 239 | 244 | 188 | 0 |
| 0 | 55 | 121 | 75 | 78 | 95 | 88 | 0 |
| 0 | 35 | 24 | 204 | 113 | 109 | 221 | 0 |
| 0 | 3 | 154 | 104 | 235 | 25 | 130 | 0 |
| 0 | 15 | 253 | 225 | 159 | 78 | 233 | 0 |
| 0 | 68 | 85 | 180 | 214 | 245 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

WEIGHT

| | | |
|---|---|---|
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

| | |
|---|---|
| 139 | 184 |

**Step 2**



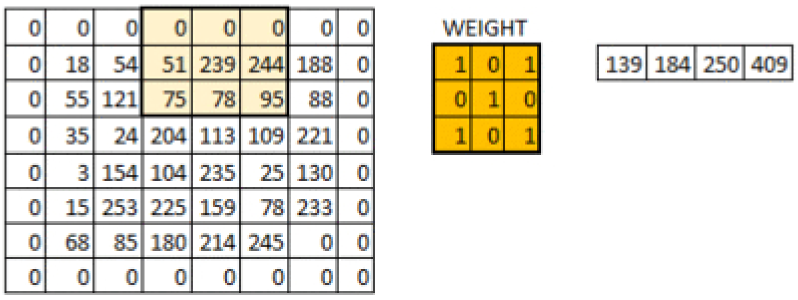
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 18 | 54 | 51 | 239 | 244 | 188 | 0 |
| 0 | 55 | 121 | 75 | 78 | 95 | 88 | 0 |
| 0 | 35 | 24 | 204 | 113 | 109 | 221 | 0 |
| 0 | 3 | 154 | 104 | 235 | 25 | 130 | 0 |
| 0 | 15 | 253 | 225 | 159 | 78 | 233 | 0 |
| 0 | 68 | 85 | 180 | 214 | 245 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

WEIGHT

| | | |
|---|---|---|
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

| | | |
|---|---|---|
| 139 | 184 | 250 |

**Step 3**



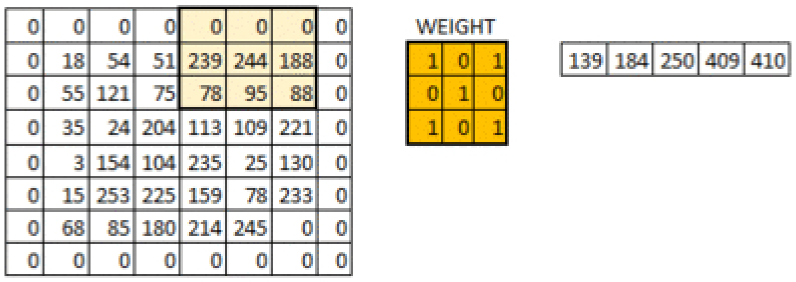
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 18 | 54 | 51 | 239 | 244 | 188 | 0 |
| 0 | 55 | 121 | 75 | 78 | 95 | 88 | 0 |
| 0 | 35 | 24 | 204 | 113 | 109 | 221 | 0 |
| 0 | 3 | 154 | 104 | 235 | 25 | 130 | 0 |
| 0 | 15 | 253 | 225 | 159 | 78 | 233 | 0 |
| 0 | 68 | 85 | 180 | 214 | 245 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

WEIGHT

| | | |
|---|---|---|
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

| | | | |
|---|---|---|---|
| 139 | 184 | 250 | 409 |

**Step 4**



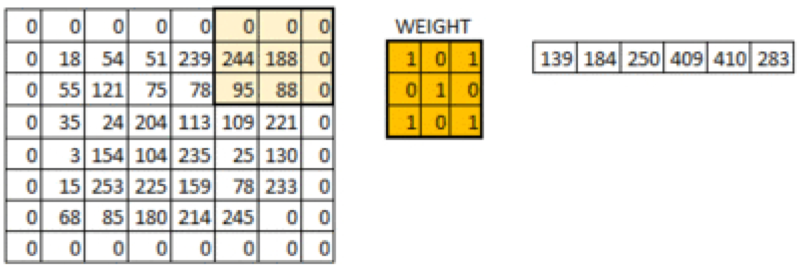
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 18 | 54 | 51 | 239 | 244 | 188 | 0 |
| 0 | 55 | 121 | 75 | 78 | 95 | 88 | 0 |
| 0 | 35 | 24 | 204 | 113 | 109 | 221 | 0 |
| 0 | 3 | 154 | 104 | 235 | 25 | 130 | 0 |
| 0 | 15 | 253 | 225 | 159 | 78 | 233 | 0 |
| 0 | 68 | 85 | 180 | 214 | 245 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

WEIGHT

| | | |
|---|---|---|
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

| | | | | |
|---|---|---|---|---|
| 139 | 184 | 250 | 409 | 410 |

**Step 5**



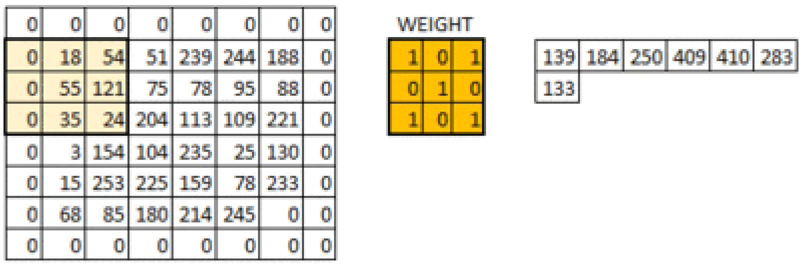
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 18 | 54 | 51 | 239 | 244 | 188 | 0 |
| 0 | 55 | 121 | 75 | 78 | 95 | 88 | 0 |
| 0 | 35 | 24 | 204 | 113 | 109 | 221 | 0 |
| 0 | 3 | 154 | 104 | 235 | 25 | 130 | 0 |
| 0 | 15 | 253 | 225 | 159 | 78 | 233 | 0 |
| 0 | 68 | 85 | 180 | 214 | 245 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

WEIGHT

| | | |
|---|---|---|
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

| | | | | | |
|---|---|---|---|---|---|
| 139 | 184 | 250 | 409 | 410 | 283 |

**Step 6**

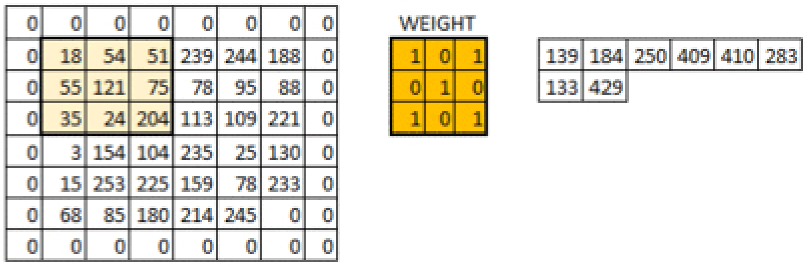


|  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 18 | 54 | 51 | 239 | 244 | 188 | 0 |
| 0 | 55 | 121 | 75 | 78 | 95 | 88 | 0 |
| 0 | 35 | 24 | 204 | 113 | 109 | 221 | 0 |
| 0 | 3 | 154 | 104 | 235 | 25 | 130 | 0 |
| 0 | 15 | 253 | 225 | 159 | 78 | 233 | 0 |
| 0 | 68 | 85 | 180 | 214 | 245 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

|  |  |  |
|---|---|---|
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

|  |  |  |  |  |  |
|---|---|---|---|---|---|
| 139 | 184 | 250 | 409 | 410 | 283 |
| 133 |  |  |  |  |  |

**Step 7**

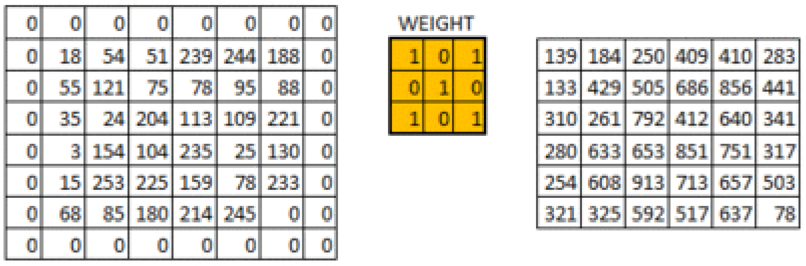


|  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 18 | 54 | 51 | 239 | 244 | 188 | 0 |
| 0 | 55 | 121 | 75 | 78 | 95 | 88 | 0 |
| 0 | 35 | 24 | 204 | 113 | 109 | 221 | 0 |
| 0 | 3 | 154 | 104 | 235 | 25 | 130 | 0 |
| 0 | 15 | 253 | 225 | 159 | 78 | 233 | 0 |
| 0 | 68 | 85 | 180 | 214 | 245 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

|  |  |  |
|---|---|---|
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

|  |  |  |  |  |  |
|---|---|---|---|---|---|
| 139 | 184 | 250 | 409 | 410 | 283 |
| 133 | 429 |  |  |  |  |

**Step 8**



|  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 18 | 54 | 51 | 239 | 244 | 188 | 0 |
| 0 | 55 | 121 | 75 | 78 | 95 | 88 | 0 |
| 0 | 35 | 24 | 204 | 113 | 109 | 221 | 0 |
| 0 | 3 | 154 | 104 | 235 | 25 | 130 | 0 |
| 0 | 15 | 253 | 225 | 159 | 78 | 233 | 0 |
| 0 | 68 | 85 | 180 | 214 | 245 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

|  |  |  |
|---|---|---|
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

|  |  |  |  |  |  |
|---|---|---|---|---|---|
| 139 | 184 | 250 | 409 | 410 | 283 |
| 133 | 429 | 505 | 686 | 856 | 441 |
| 310 | 261 | 792 | 412 | 640 | 341 |
| 280 | 633 | 653 | 851 | 751 | 317 |
| 254 | 608 | 913 | 713 | 657 | 503 |
| 321 | 325 | 592 | 517 | 637 | 78 |

**Step 9**

In Step 1, the kernel mapping the first 9 element in the top left corner and calculates the output integer. Following it, from Step 2 to Step 5 , the kernel moves rightward to finish the mapping until the last effective index. After that, the kernel jump to the starting index of the **second** row of the input, and operates the same as in the first row.

In the end, the final convolution will be like in Step 9.

Here is the definition for the images and kernel:

O: The size of output image,

I:  The size of input image,

K:  The size of the kernel,

S:  The stride value of the kernel's single step moving. In the example above, the stride is 1, therefore, the kernel moves one index every step.

P:  The padding number,

The output size is calculated with formula:

$$O = \frac{I - K + 2P}{S} + 1$$

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 3 & 3 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 4 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 3 & 1 & 2 \\ 2 & 0 & 2 \\ 1 & 1 & 2 \end{bmatrix}$$

In this assignment, the input Matrix A will be with size 5 by 5, and the kernel matrix B will remain the same with the size 3 by 3. Please calculate the convoluted output with ***0 padding*** and ***stride = 1***.

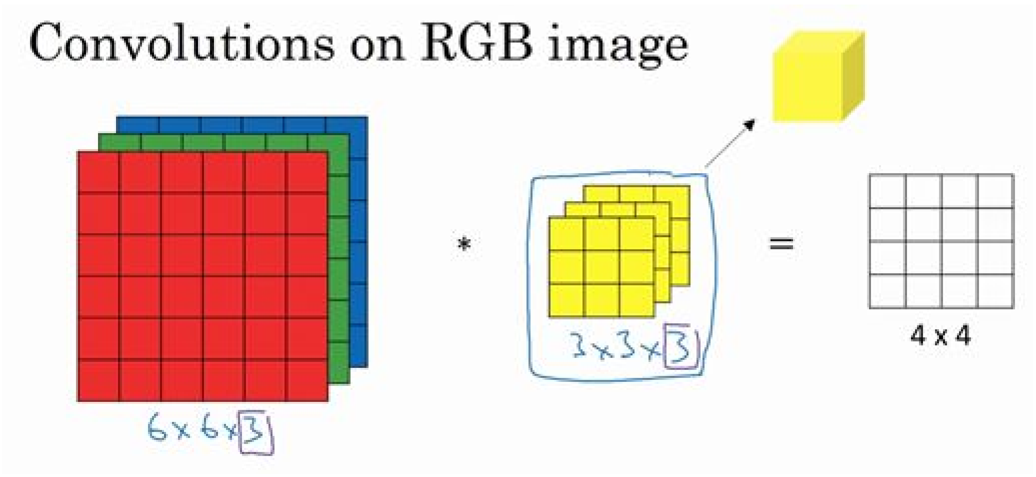
Again with verilog and Vivado Design Suite, write the code for the convolution function, and store the waveform of the final matrix result.

## Assignment 3

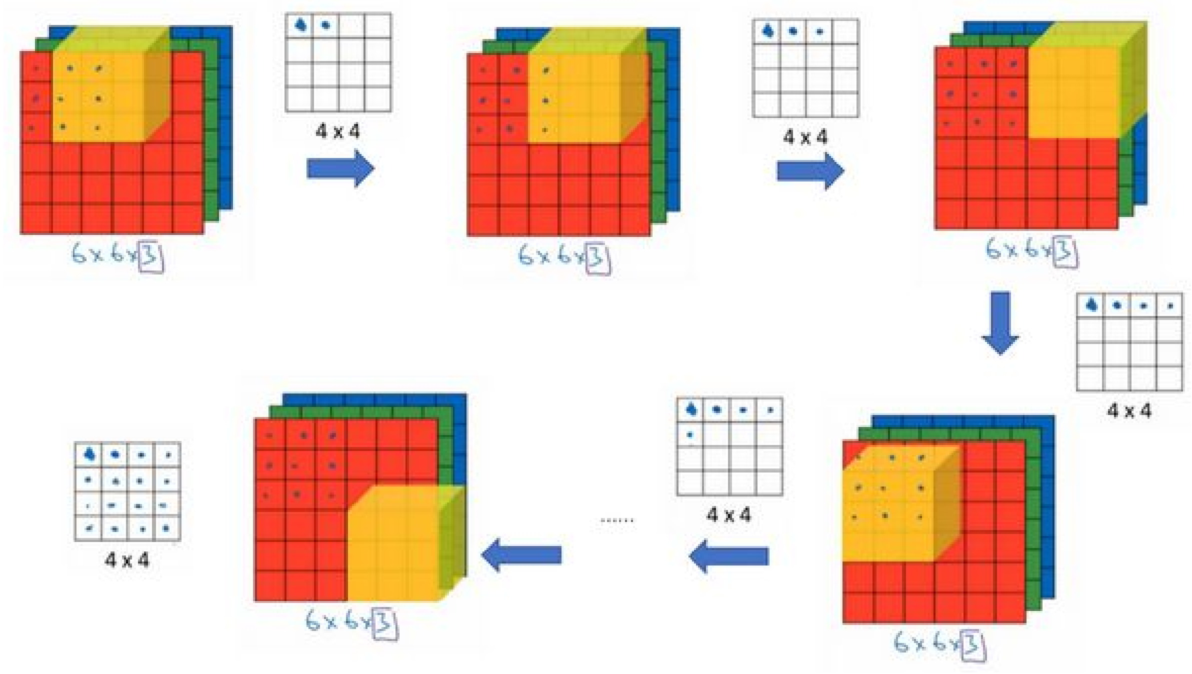
In deep learning, the input images will be more than 2 dimensions, containing the width, height and the depth or the channels of the images. Another term to introduce the multi-dimensional input image is call **tensor**.

Also for the kernel matrix, to get the features for one channel of the input images, kernel with multiple channels will be designed. To get wanted features, such as horizontal edges or vertical edges, multiple kernels will be implemented on the same input image for convolution.

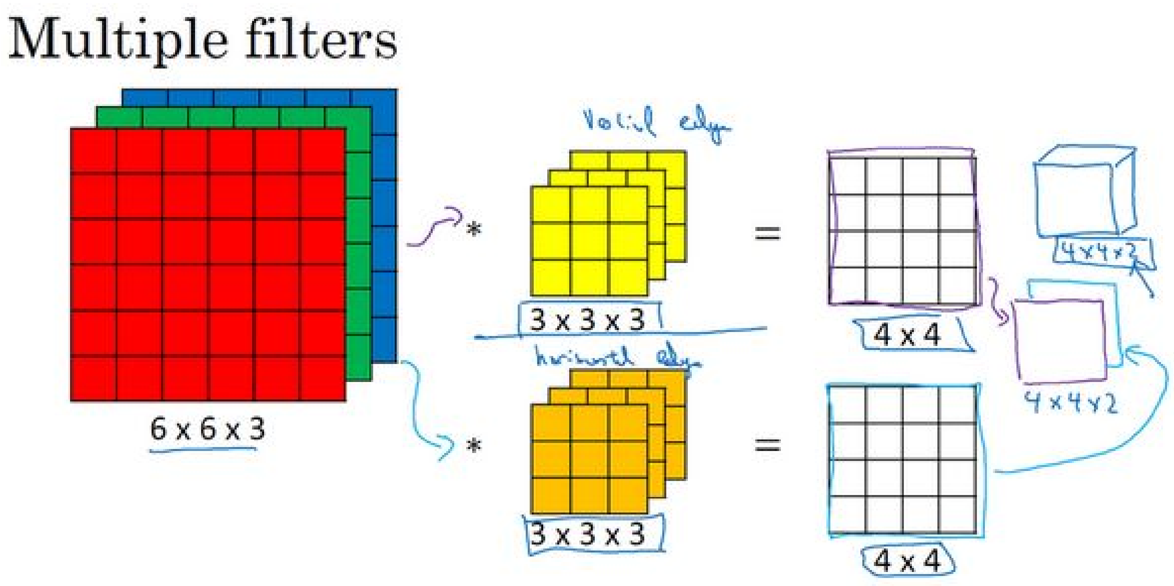
For example, there is a 3D matrix with the size 3 x 3 x 3. Still the same with the 2D convolution in Assignment 2 , the total **27** weight parameters of the kernel is fetched and implemented on the **top left 3 x 3 corner**, and the convolution will output an integer, which is the first element for the final 4 x 4 matrix.



Repeating the above operation, the 3D convolution will finally output the 4 x 4 matrix, whose size can be determined by the **formular**



As mentioned above, multiple filters can be implemented to get different features, such as vertical edges and horizontal edges. If the input image is with the size 6 x 6 x 3, and the kernel size is 3 x 3 x 3 x 2, where the last number mean the number of filters, then the output convolution size is 4 x 4 x 4. Therefore, the number of the output matrix equals the number of the kernel.



In the task, you are required to perform a convolution on Matrix A with the size: 6 x 6 x 3 with the kernel with the size: 3 x 3 x 3 x 2. Calculate the output matrix.

Again with verilog and Vivado Design Suite, write the code for the convolution function, and store the waveform of the final matrix result.

$$A0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 3 & 3 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 4 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad A_1 = \begin{bmatrix} 1 & 3 & 3 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 4 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad A_2 = \begin{bmatrix} 1 & 1 & 4 & 2 & 0 \\ 0 & 3 & 3 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$Kernel B_0: \quad B_{00} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix} \quad B_{01} = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \quad B_{02} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}$$

$$Kernel B_1: \quad B_{10} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad B_{11} = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix} \quad B_{12} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

# Lab 4 HLS

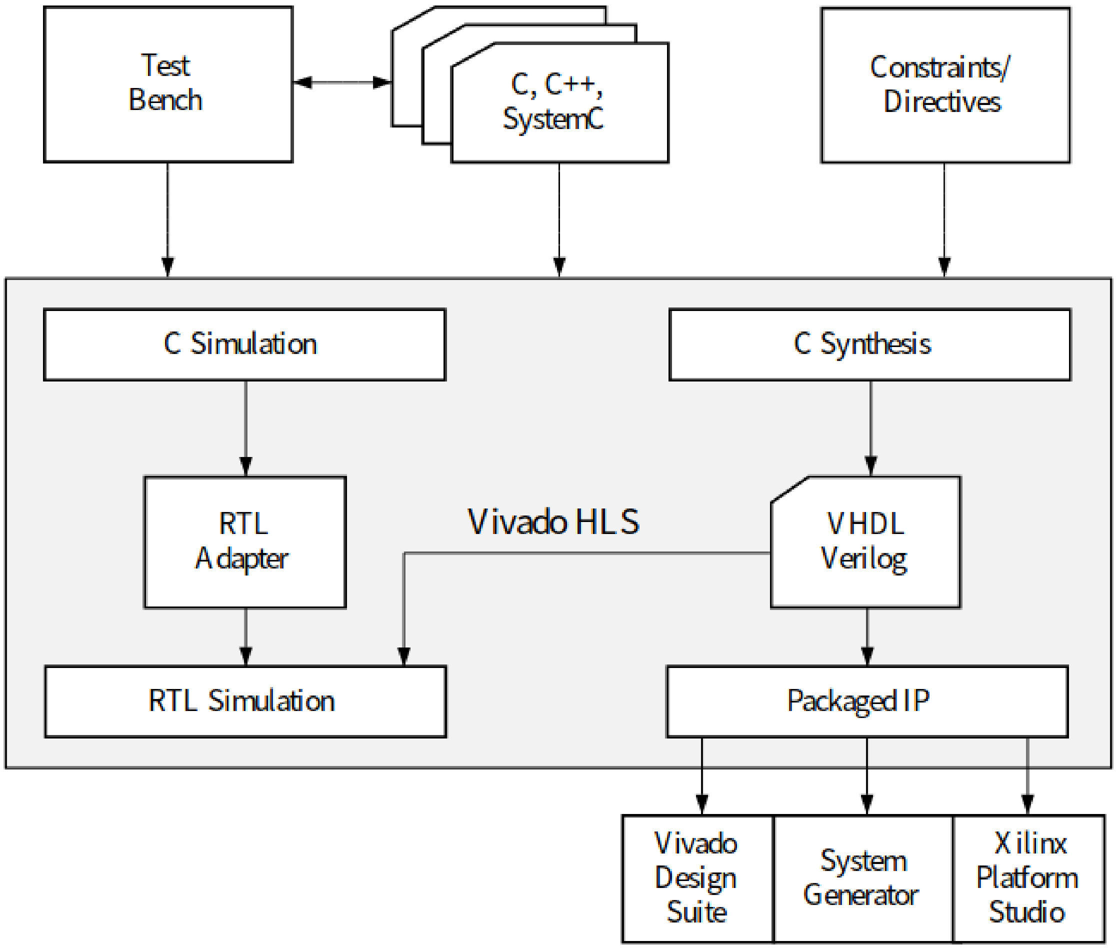
## 4.1 Goal

## 4.2 Introduction

### HLS introduction

HLS, abbreviated for High-Level-Synthesis, is a high level abstraction tool to automatically create RTL(Register Transfer Level) design.

The following figure shows an overview of the Vivado HLS input and output files. The Xilinx Vivado HLS tool synthesizes a C function into an IP block that you can integrate into a hardware system. It is tightly integrated with the rest of the Xilinx design tools and provides comprehensive language support and features for creating the optimal implementation for your C algorithm.



In two parts, HLS shows its advantage:

1. HLS syhthesizes the function's content into RTL statement with it's algorithm synthesis.
2. HLS helps transform the function's paramenters, inputs and outputs into ports(**output**, **input**, **reg**, **wire**, **RAM**).

In all, HLS save the developers from the details design of the circuit with verilog/VHDL.

## Hardware description-AVNET Ultra96-V2

Ultra96-V2 is an Arm-based, Xilinx Zynq UltraScale+™ MPSoC  development board based on the Linaro 96Boards Consumer Edition (CE)  specification.the Ultra96-V2 boots from the provided Delkin 16 GB microSD card.  Engineers have options of connecting to Ultra96-V2 through a Webserver  using integrated wireless access point capability or to use the provided PetaLinux desktop environment which can be viewed on the integrated  Mini DisplayPort video output.

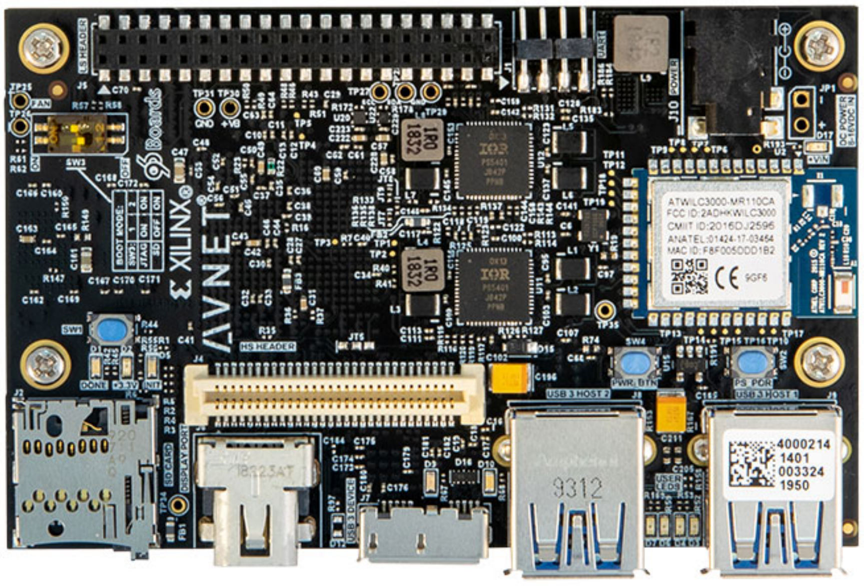
Here the feature table and the application fields are given below. For more details, you can refer to the product's homepage.

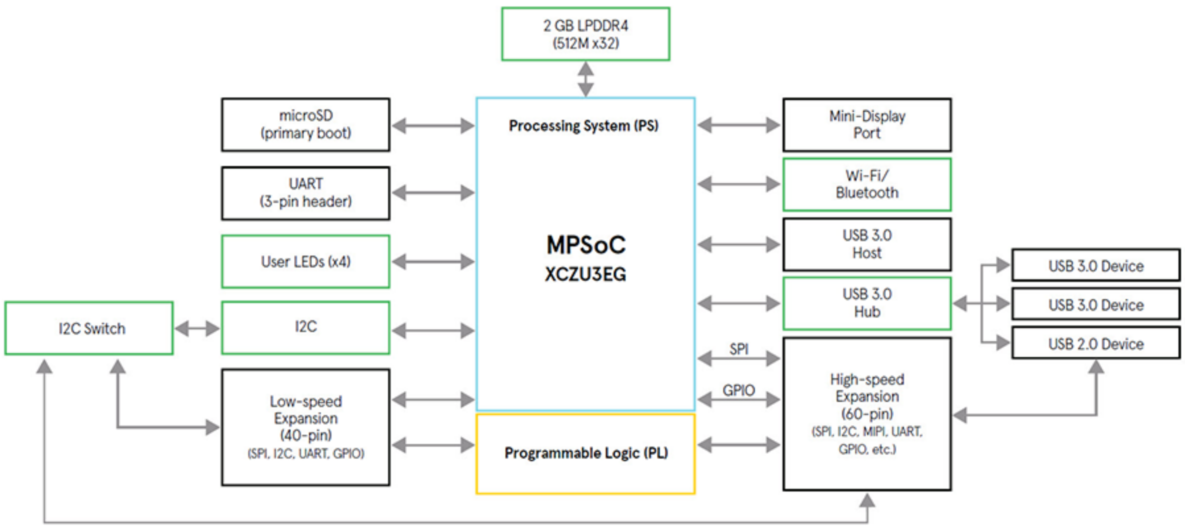
| Feature Table |
|---|
| Xilinx Zynq UltraScale+ MPSoC |
| ZU3EG A484 Micron 2 GB (512M x32) LPDDR4 |
| Memory Delkin 16 GB microSD card + adapter |
| PetaLinux environment available for download |
| Microchip Wi-Fi / Bluetooth |
| IDT VersaClock 6E clock generator |
| Infineon high efficiency power management |
| Aavid/Boyd Heatsink |
| Mini DisplayPort (MiniDP or mDP) |
| 1x USB 3.0 Type Micro-B upstream port |

Table 4.1

| Target Applications |
|---|
| Artificial Intelligence |
| Machine Learning |
| IoT/Cloud connectivity for add-on sensors |
| Embedded Computing |
| Robotic |
| Entry level Zynq UltraScale+ MPSoC development environment |
| Training, prototyping and proof-of-concept demo platform |
| Wireless design and demonstrations using Wi-Fi and Bluetooth |

Table 4.2





## Software development kit

The Deep Neural Network Development Kit (**DNNDK**) package can be freely downloaded after registration from the Xilinx website:

https://www.xilinx.com/products/design-tools/ai-inference/edge-ai-platform.html.

Deep Neural Network Development Kit (DNNDK) is a full-stack deep learning SDK for the Deep-learning Processor Unit (DPU). It provides a unified solution for deep neural network inference applications by providing pruning, quantization, compilation, optimization, and run time support.
Key
highlights and features are listed below:
Innovative full-stack solution for deep learning inference application development
• A complete set of optimized tool chains, including compression, compilation and runtime.
• Lightweight C/C++ and Python programming APIs.
• Easy-to-use with gradual learning curve.

As shown in the figure below, DNNDK framework is composed of Deep Compression Tool
(DECENT),
Deep Neural Network Compiler (DNNC), Deep Neural Network Assembler (DNNAS), Neural
Network
Runtime (N2Cube), DPU Simulator, Profiler, DExplorer and DDump.
Since v3.0, DNNDK supports both Caffe and TensorFlow with a unified framework. The toolchain
flow
for deploying Caffe and TensorFlow network models are almost the same, which delivers a
familiar user
experience.

4.3