

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

AI-accelerator-course-preparation tutorial

AI-accelerator-course-preparation tutorial

Lab 1 Software platform Description

1.1 Vivado® Design Suite

- Introduction

- Component

 - Vivado High-Level Synthesis compiler

 - Vivado Simulator

 - Vivado IP Integrator

 - Vivado Tcl Store

1.2 Resource downloading

1.3 Final installation and licensing

- Windows version

- Linux version

- Licensing

Lab 2 Verilog & VHDL:

2.1 Design Flow

- Project creating

- Source Adding

- RTL Analysis

2.2 Testbench

- Timescale definition and variable declaration

- DUT Instantiating

- Generating clock

- Simulation

2.3 Lab Task(u)

Lab 3 Convolutional Layer

3.1 Introduction

- What is **convolution**?

 - Linear formula

 - Discrete formula

- Convolution in Image Processing

 - edge1

3.2 Task(U)

Lab 1 Software platform Description

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).

1.1 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.

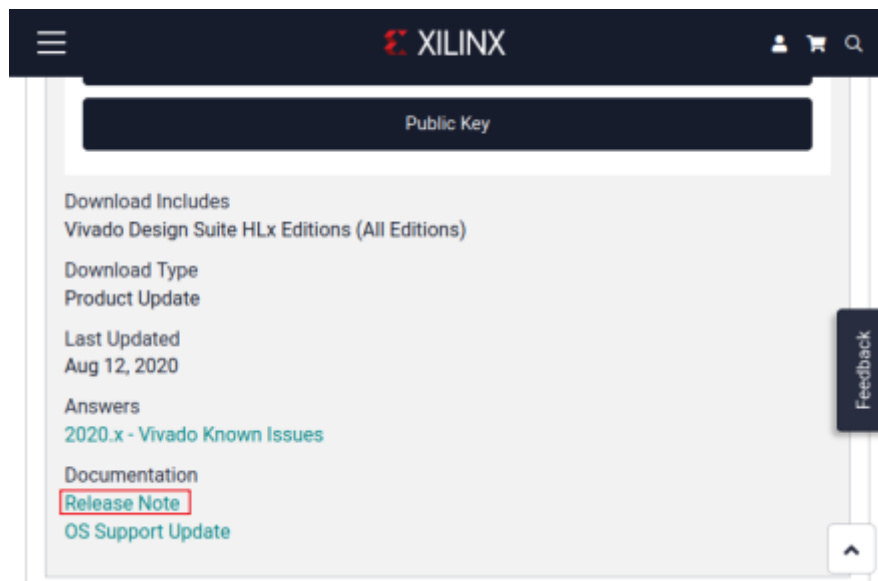
1.2 Resource downloading

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.



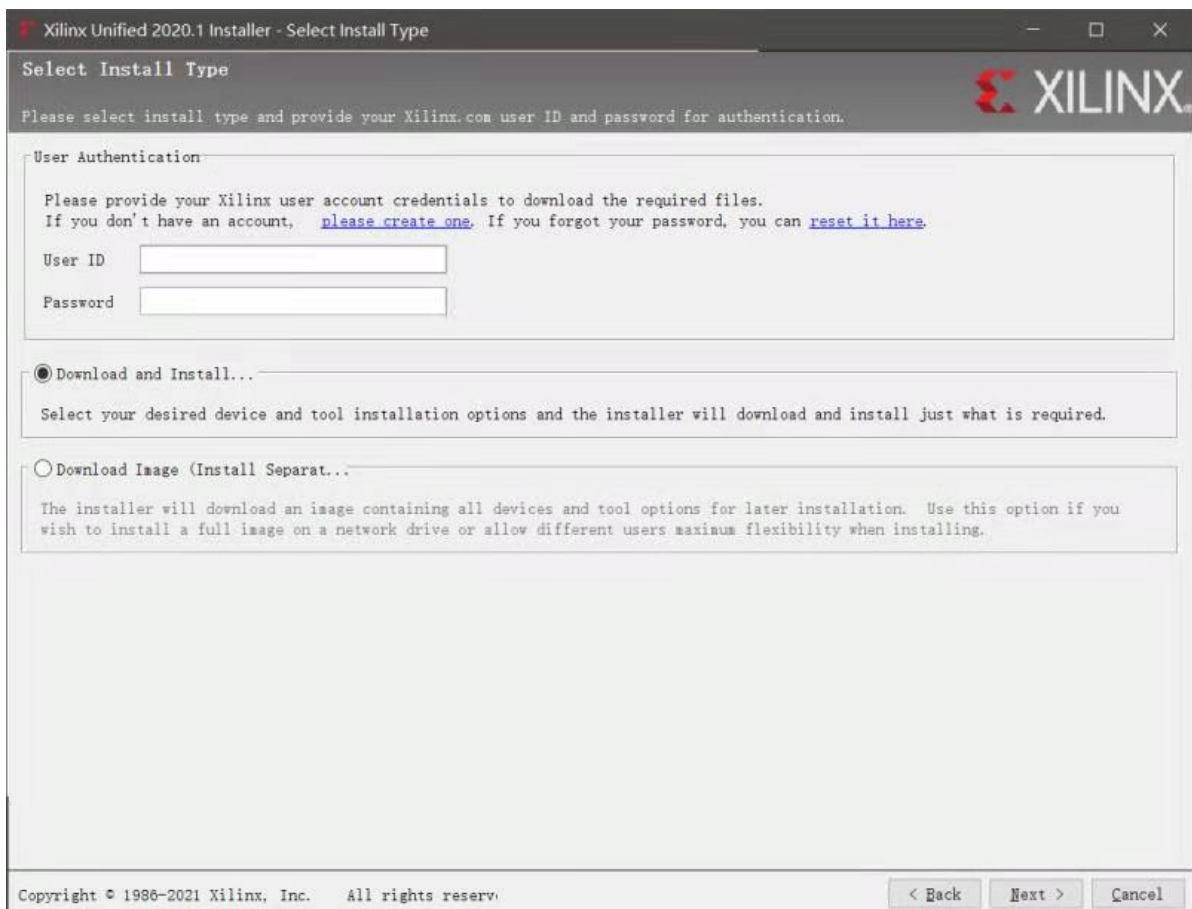
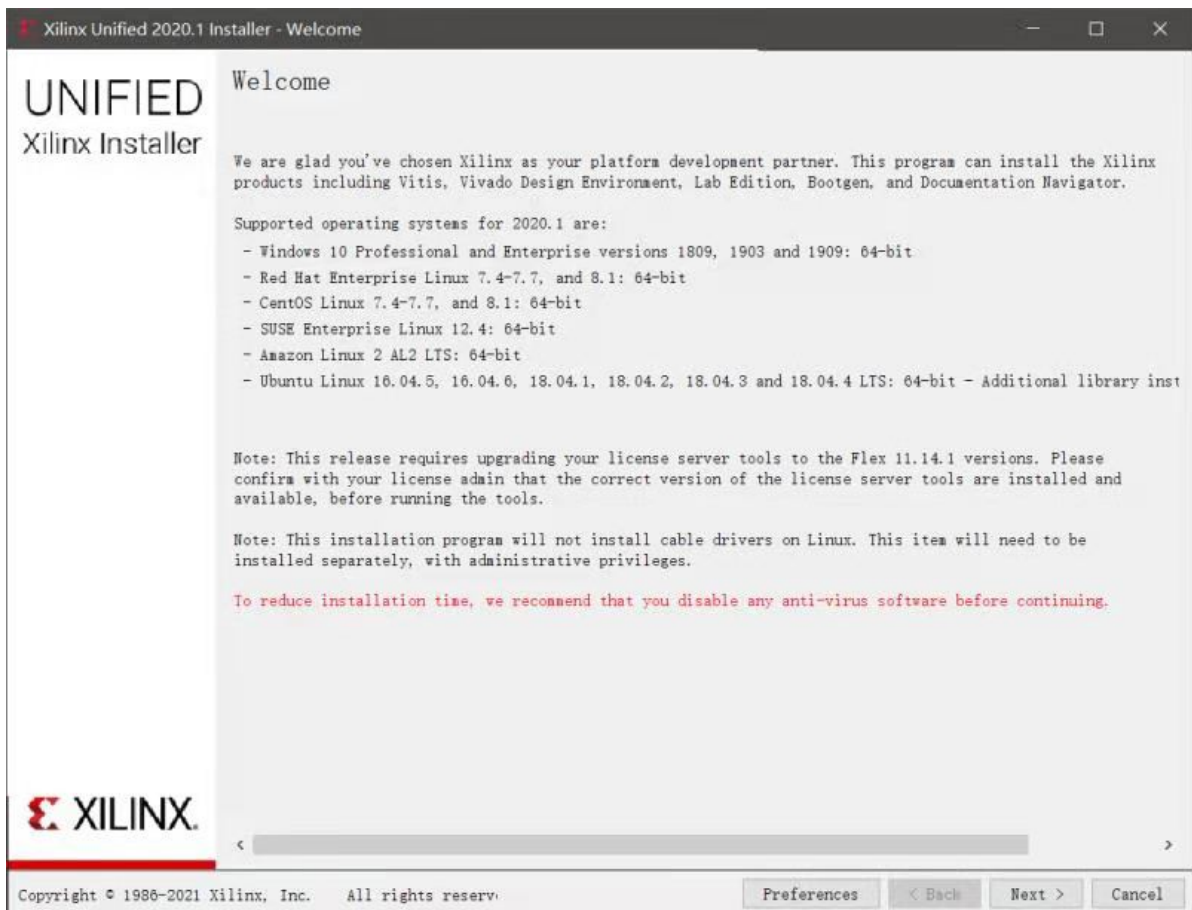
1.3 Final installation and licensing

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.



Xilinx Unified 2020.1 Installer - Accept License Agreements

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 [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 <https://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 satisfy 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 [clicking here](#)

☒ I Agree

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

< Back Next > Cancel

Xilinx Unified 2020.1 Installer - Select Product to Install

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

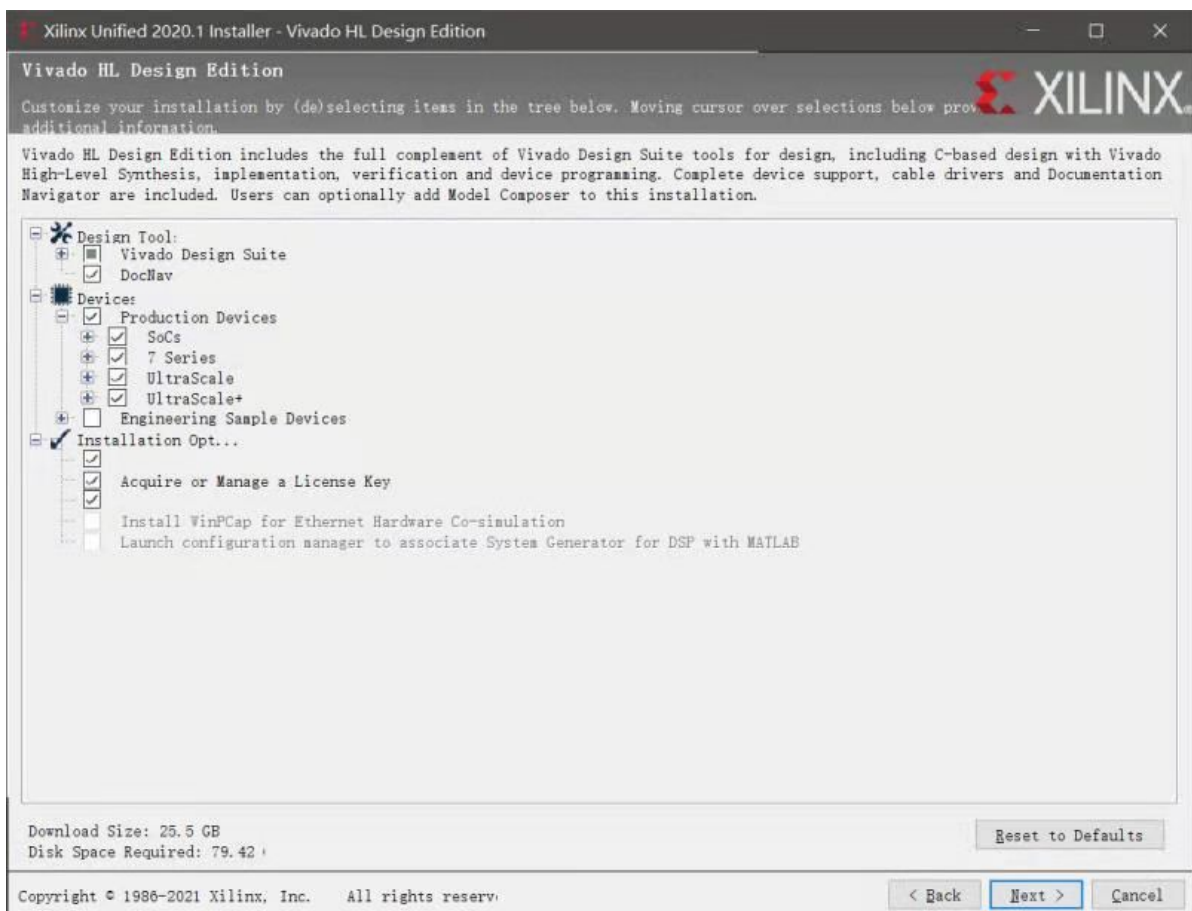
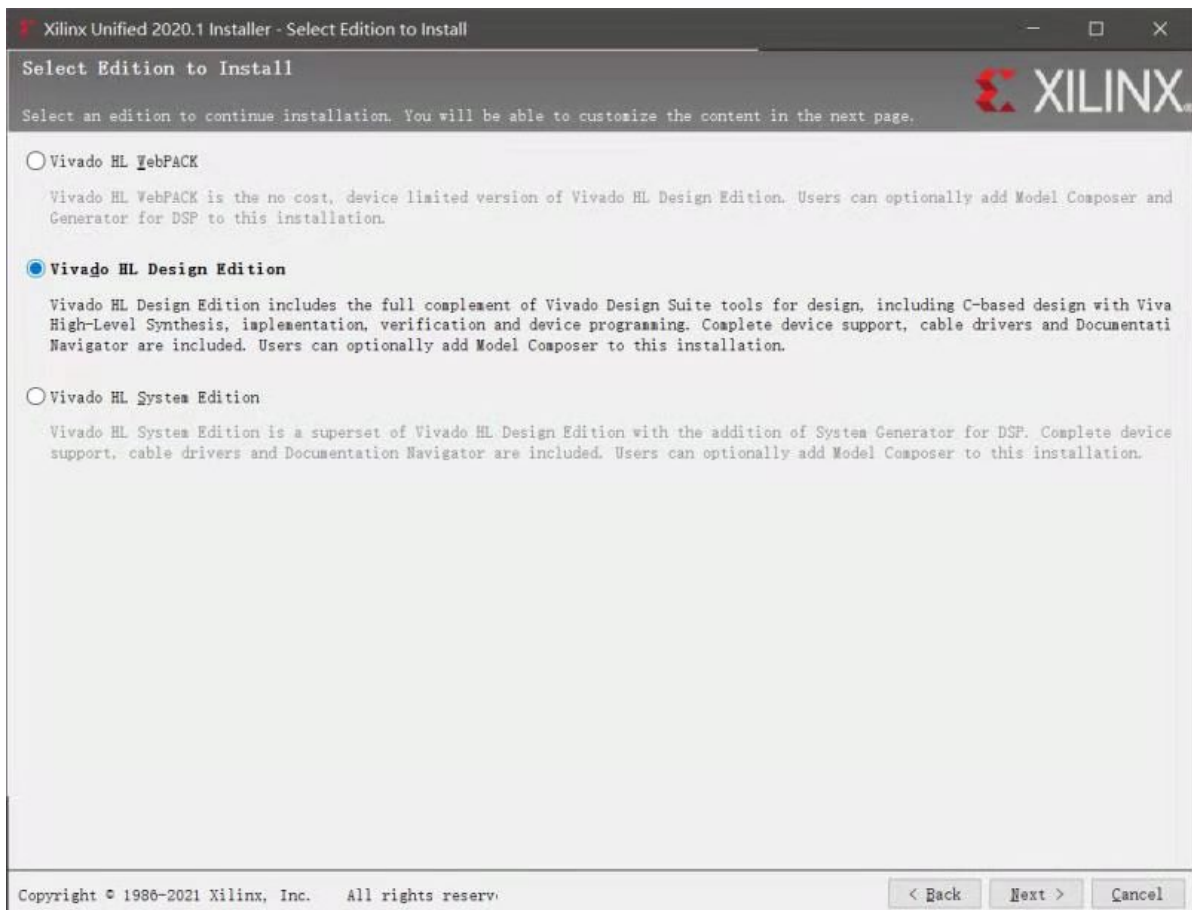
Installs hardware server and JTAG cable drivers for remote debugging.

☐ 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.

< Back Next > Cancel



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

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

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

< Back Next > Cancel

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

XILINX

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

Preferences < Back **Install** Cancel

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][Figure 1] .

Please ensure that you have changed the file permissions to execute.

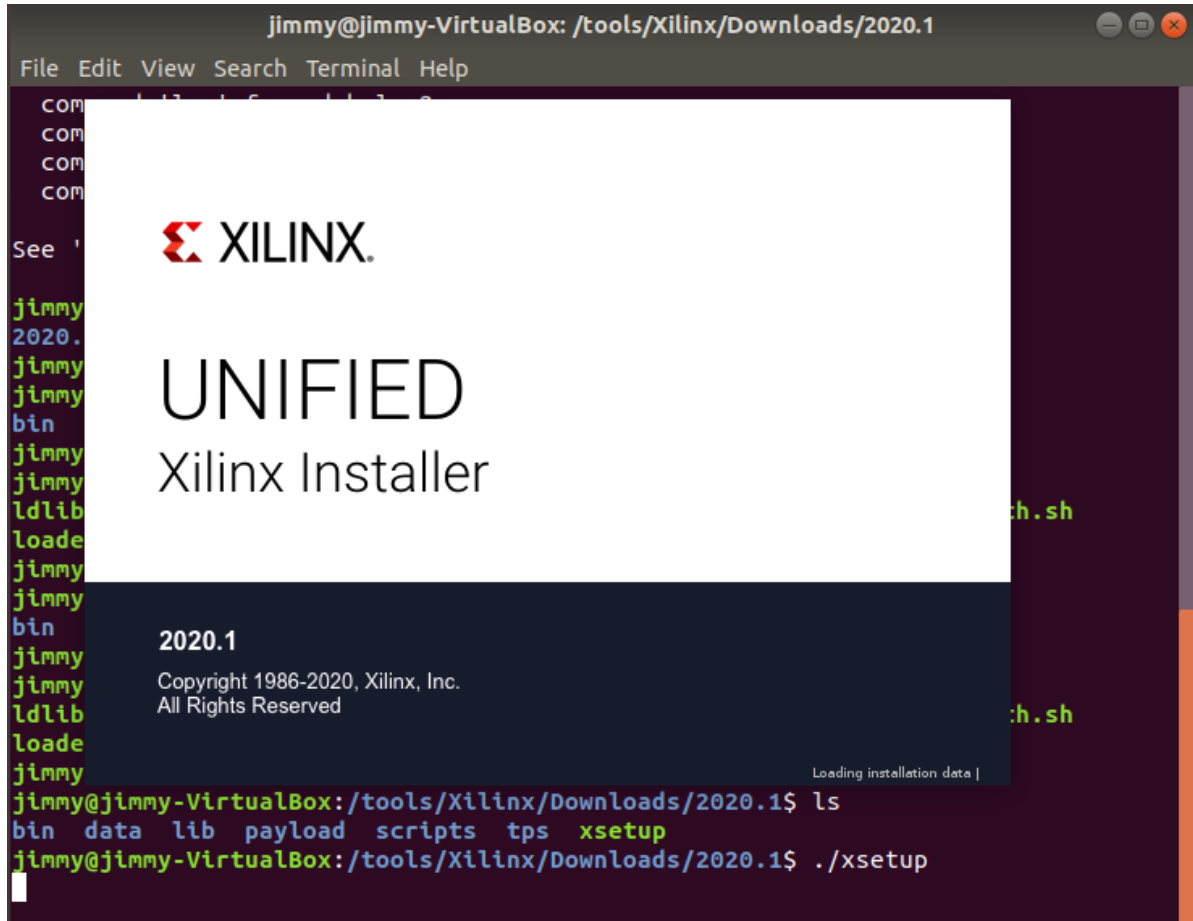


Figure 1

Following the initializing of the installer, a welcome window will appear with the necessary information about the supported operating systems, shown in [Figure 2][setup3].

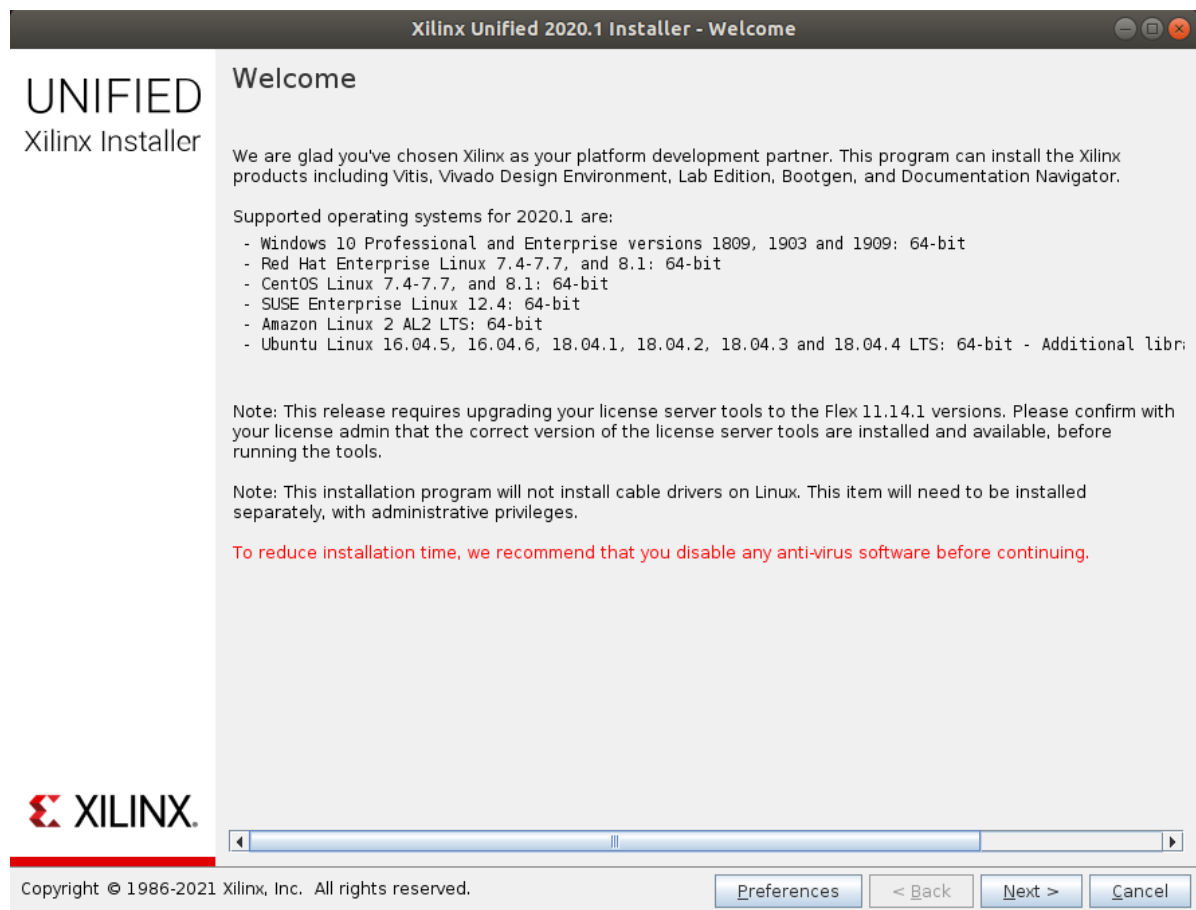


Figure 2

Accept all the license agreement, otherwise the installation will not proceed, displayed in Figure 3.

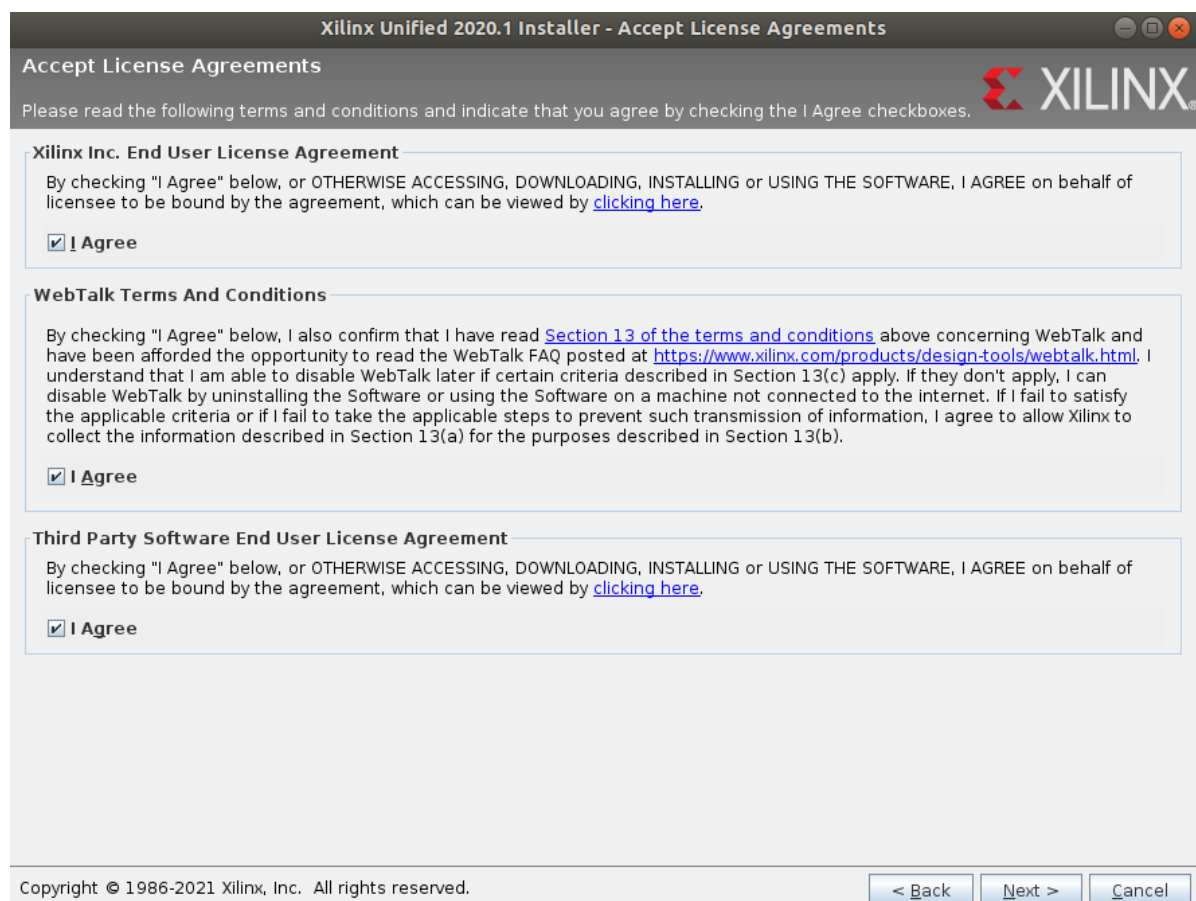


Figure 3

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.

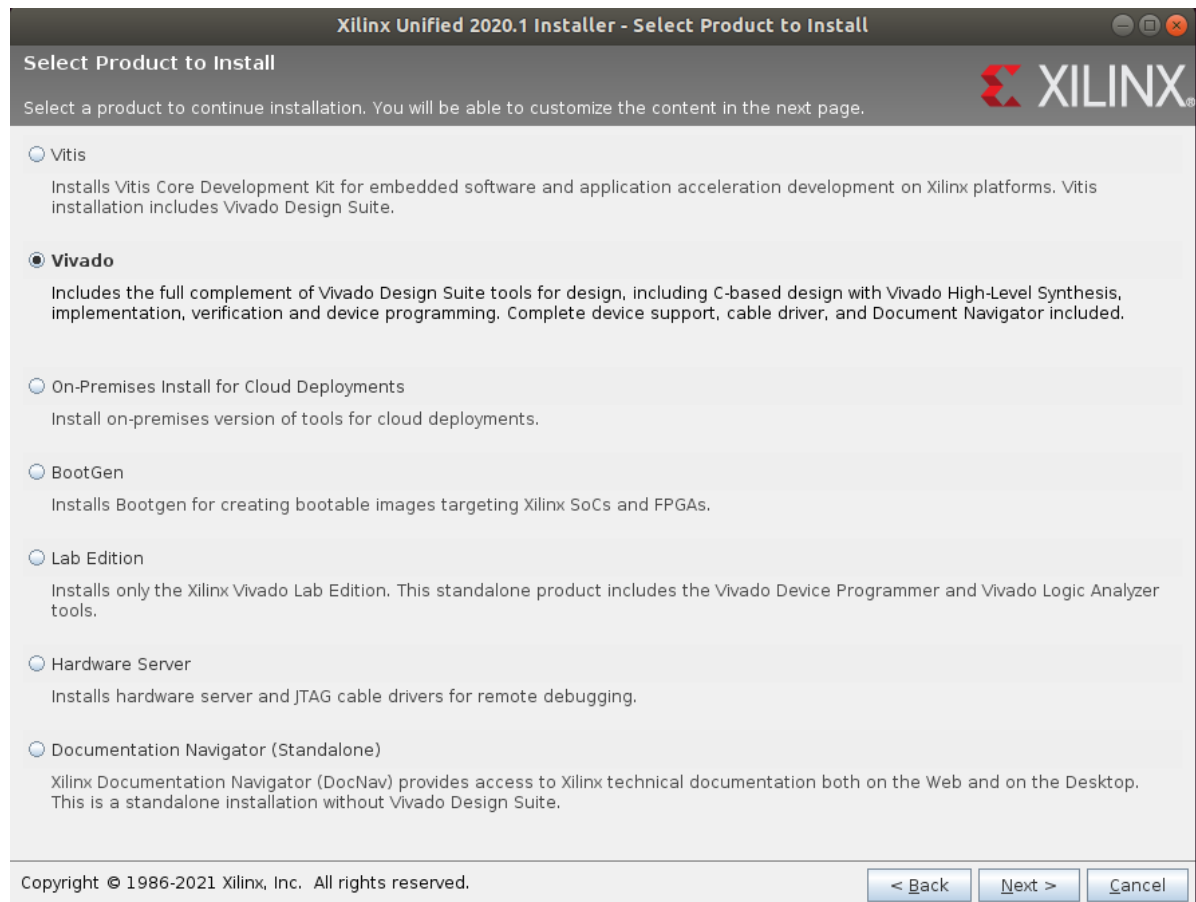


Figure 4

Choose the Vivado HL System Edition.

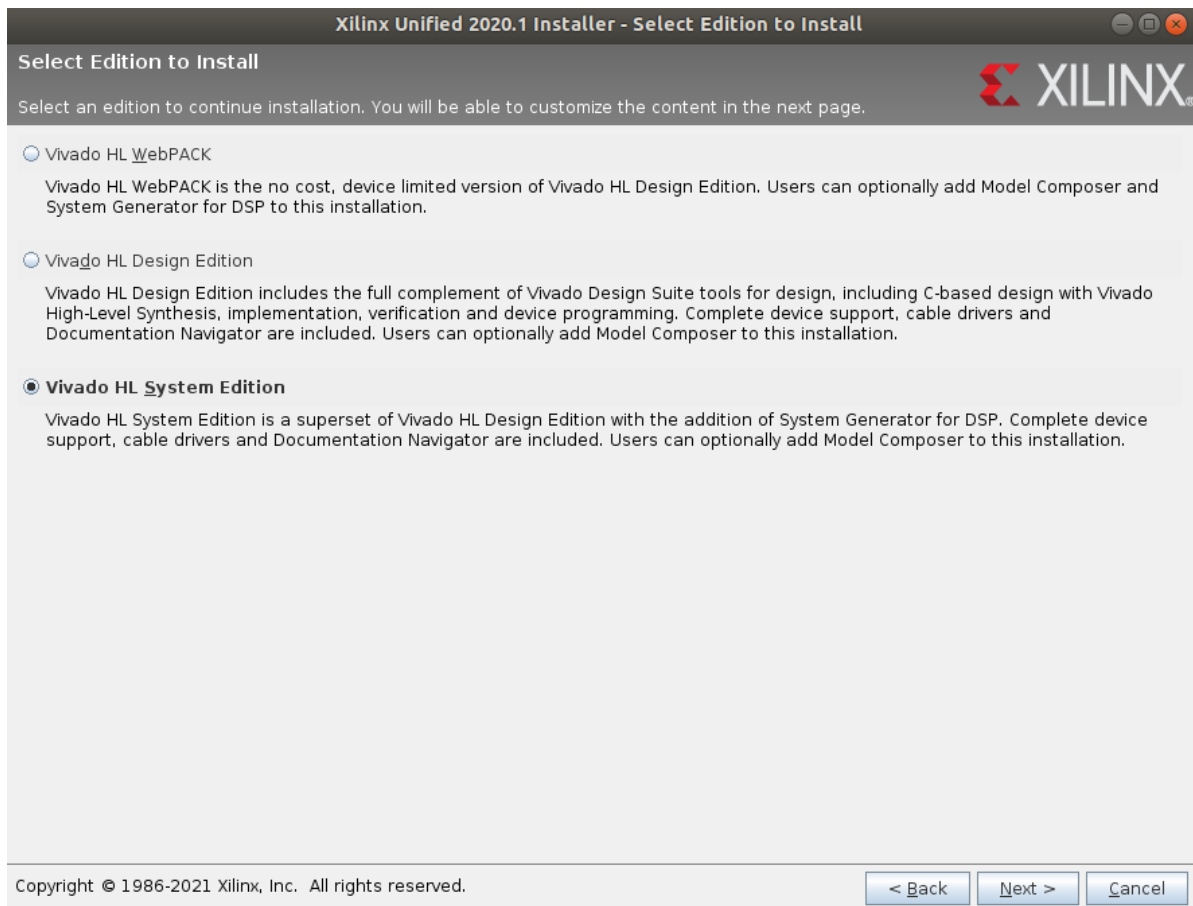


Figure 5

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.

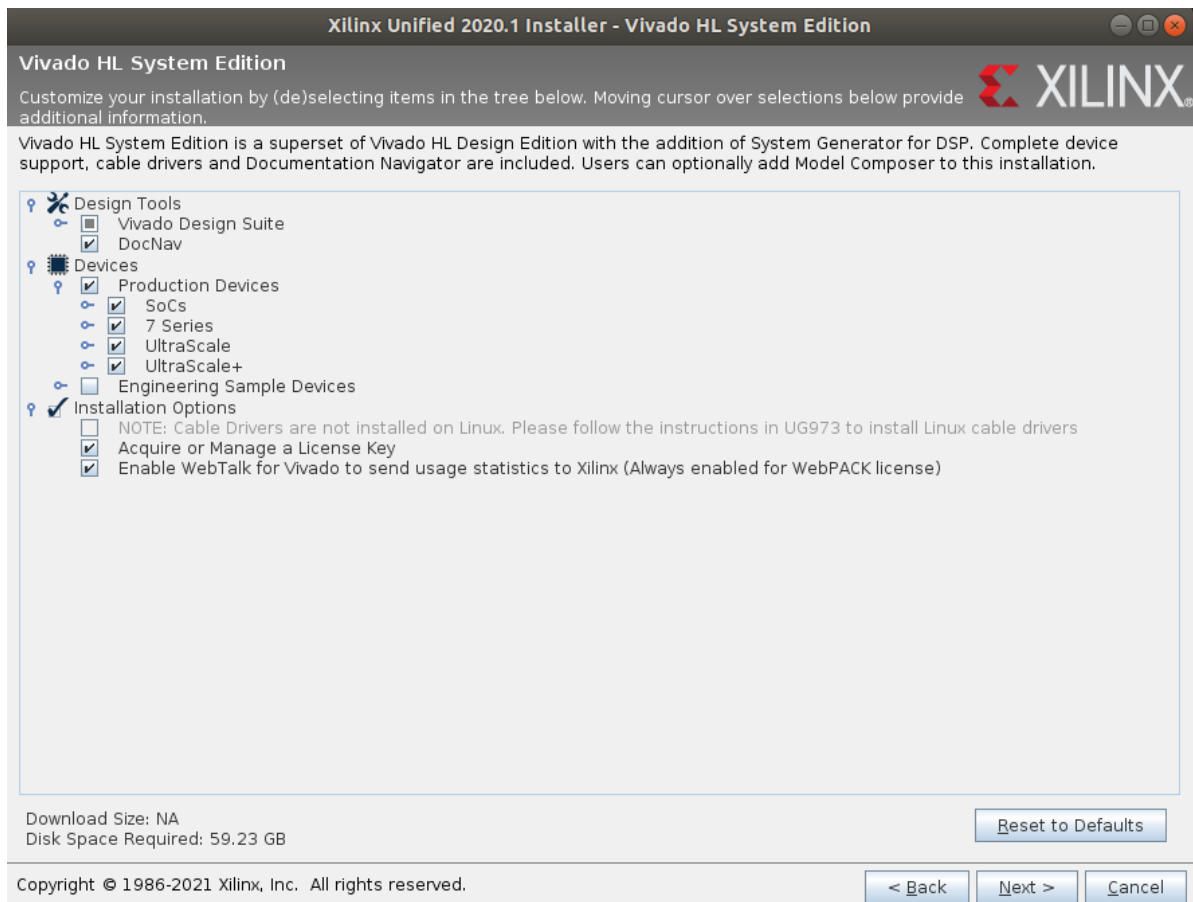


Figure 6

Choose the installation location.

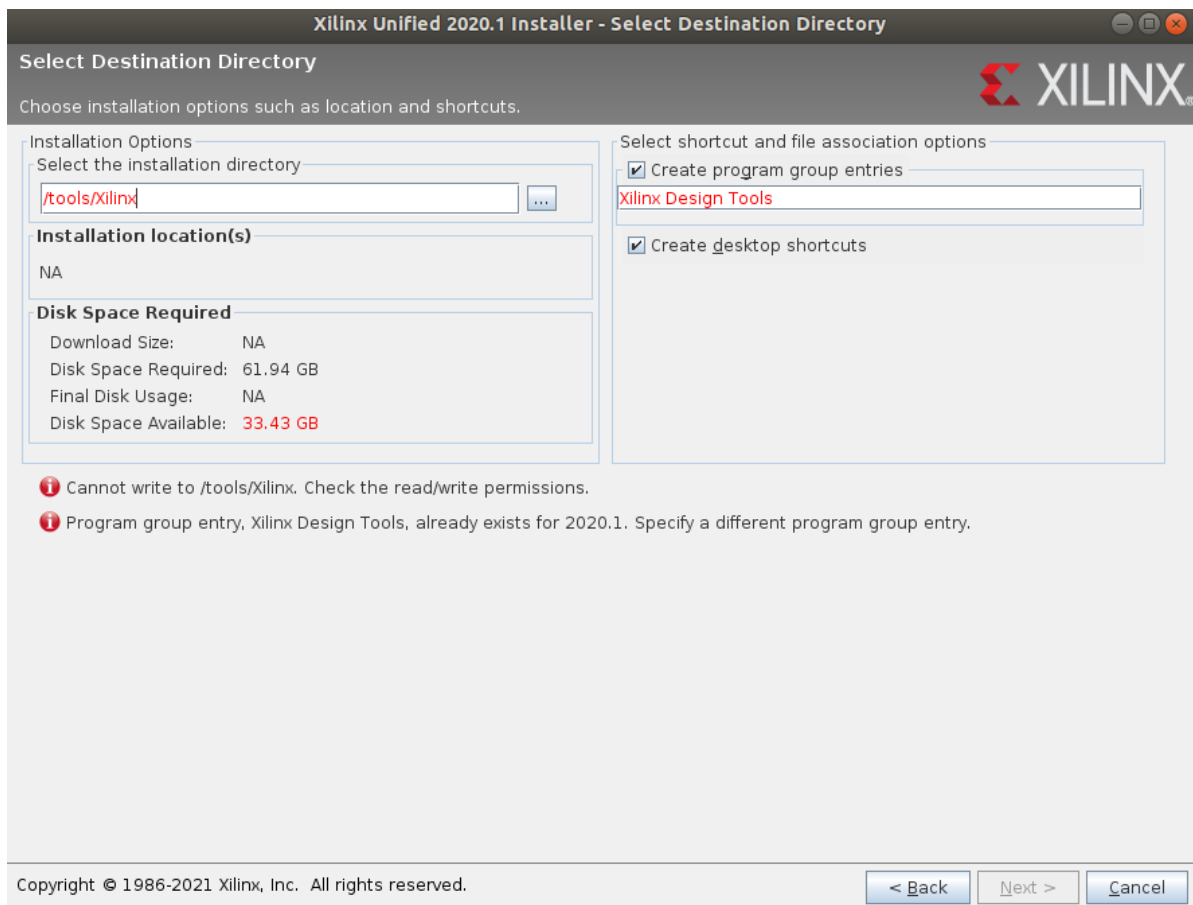


Figure 7

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.

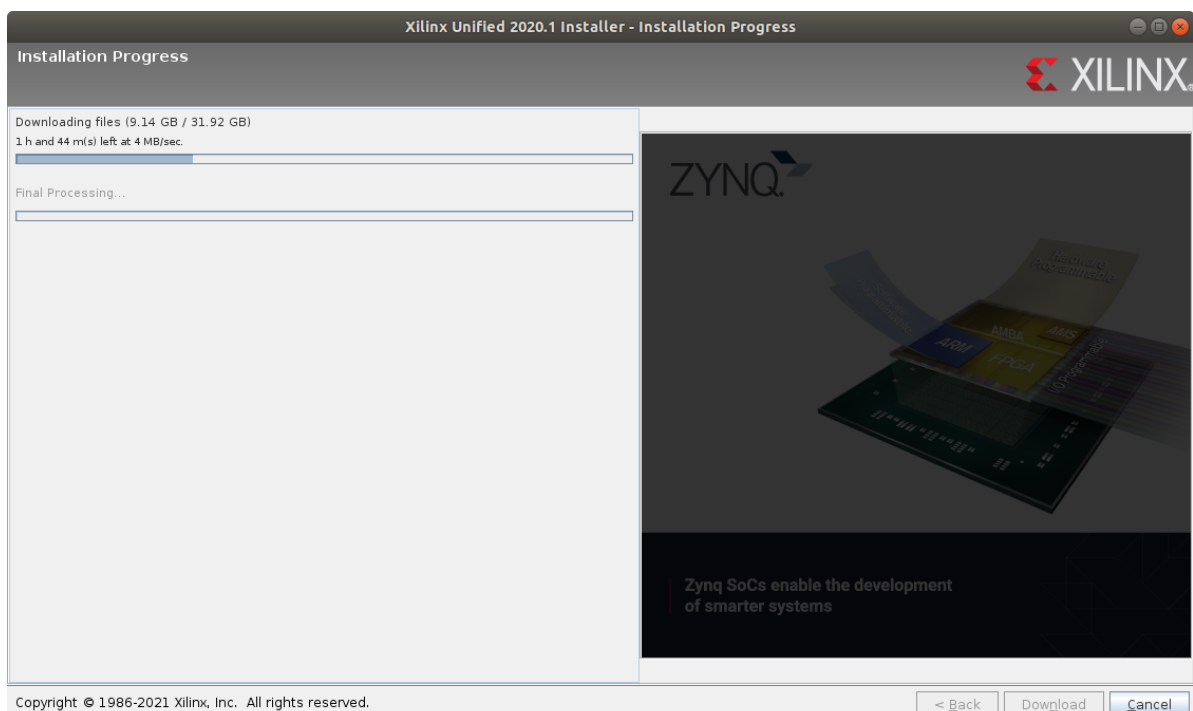
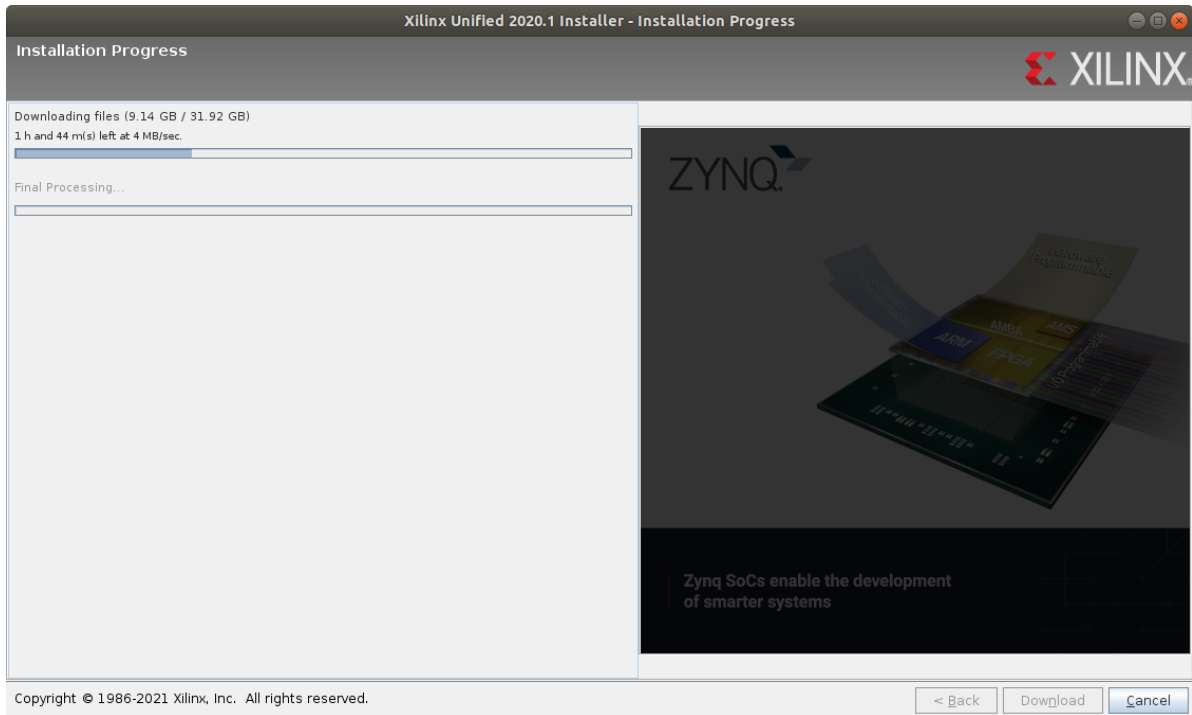


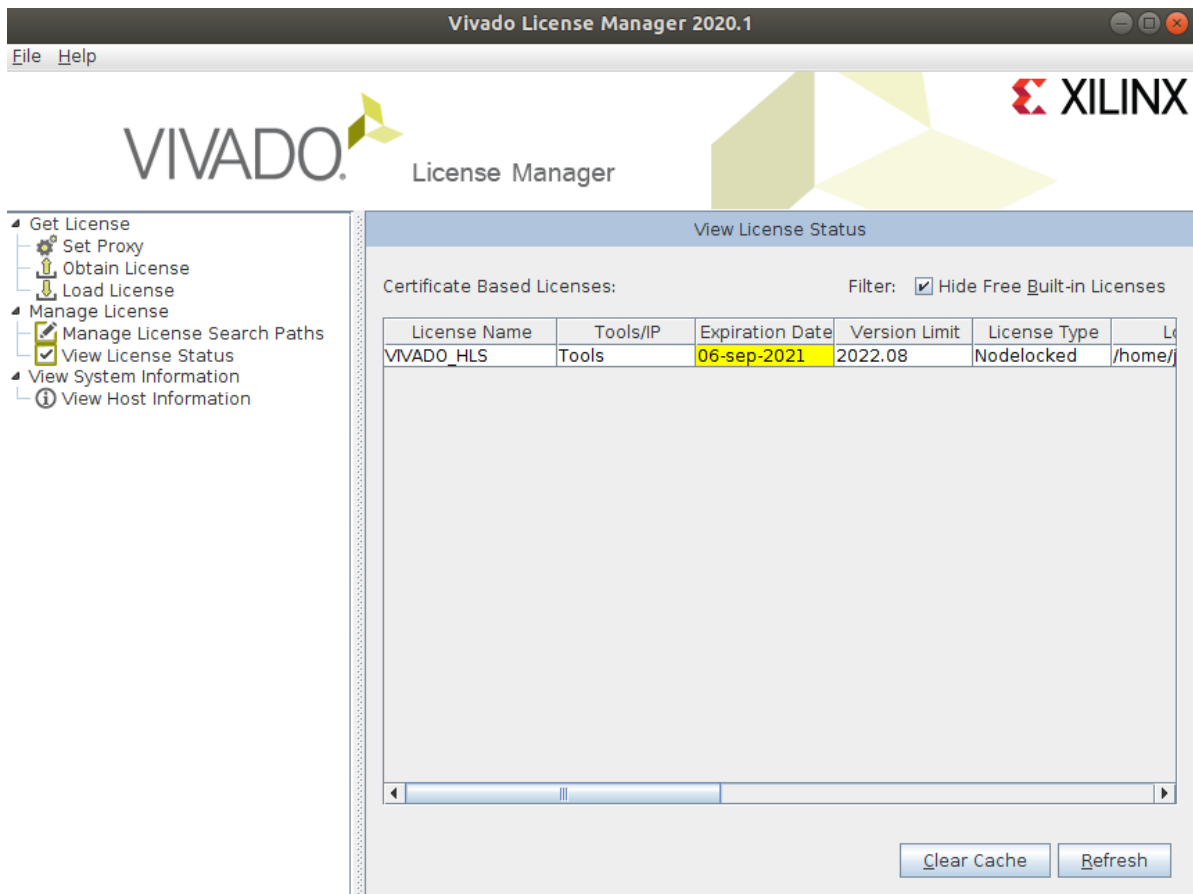
Figure 8



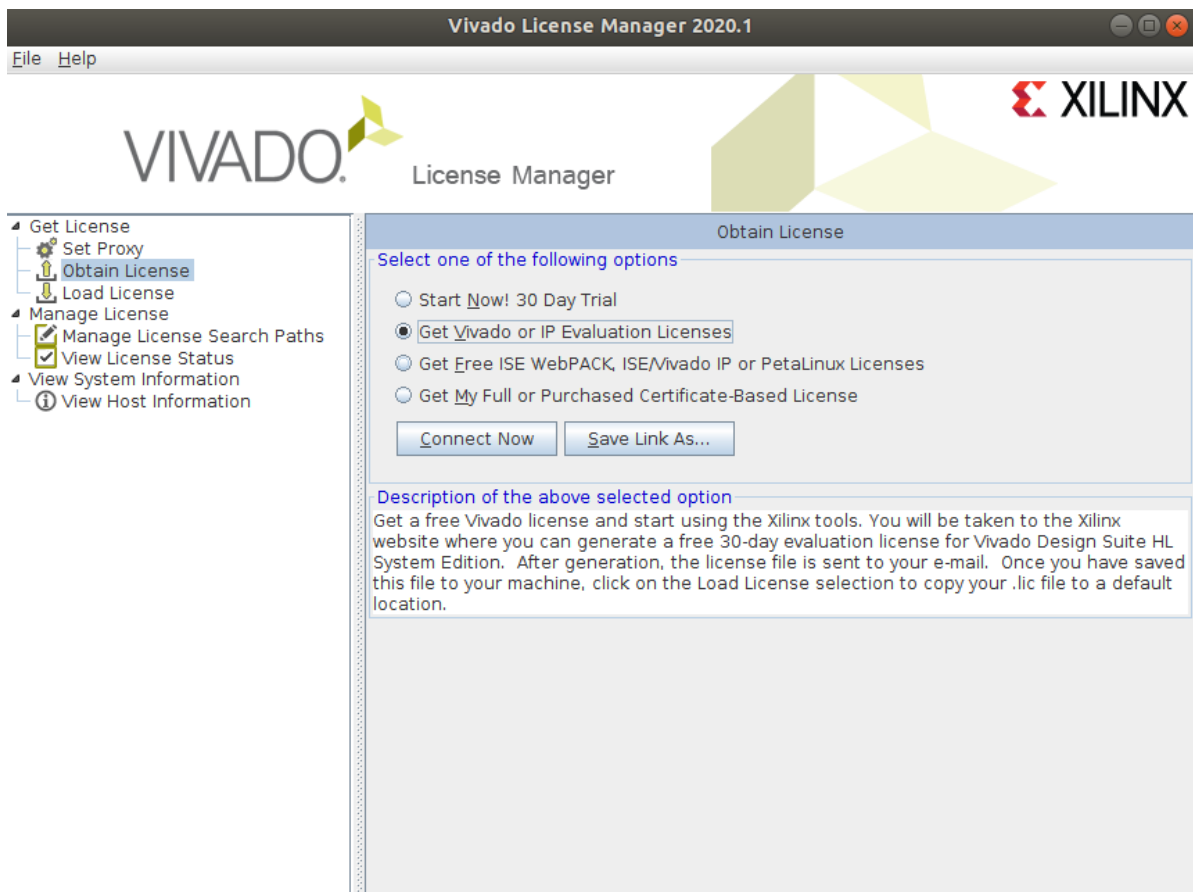
Licensing

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.

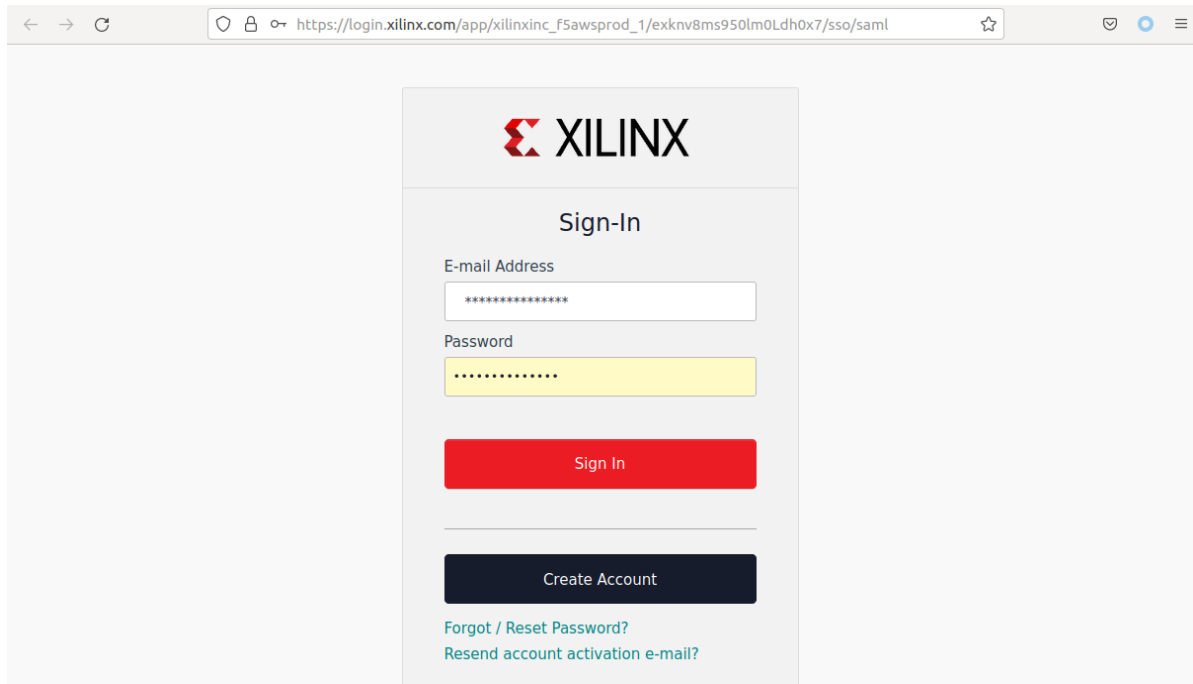
After completing the installation of Vivado, SDx or ISE Design Suite, the Xilinx License Configuration Manager (XCLM) will start automatically and guide you through the licensing process, displayed in Figure 9. You may also go directly to the [Xilinx Product Licensing Site](#) to obtain licenses for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



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

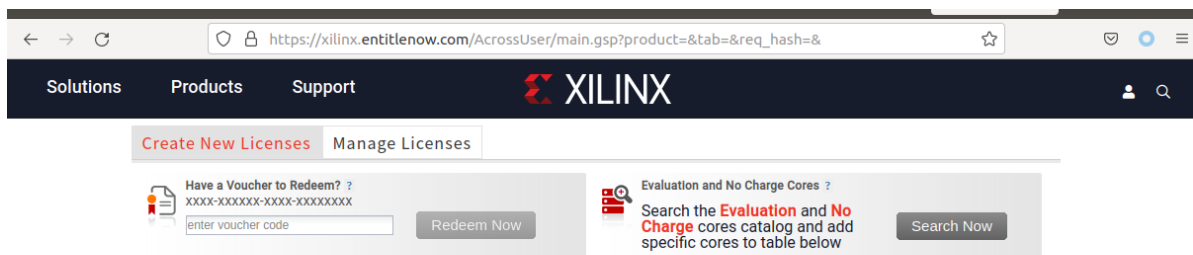


The program will ask you to sign in with your pre-registered user information, including e-mail address and password.



The image shows a web browser window displaying the Xilinx Sign-In page. The URL in the address bar is https://login.xilinx.com/app/xilinxinc_f5awsprod_1/exknv8ms950lm0Ldh0x7/sso/saml. The page features the Xilinx logo at the top, followed by the heading "Sign-In". Below the heading are two input fields: "E-mail Address" and "Password", both containing masked text (asterisks). A red "Sign In" button is positioned below the password field. Below the button is a dark blue "Create Account" button. At the bottom, there are two links: "Forgot / Reset Password?" and "Resend account activation e-mail?".

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



The image shows the Xilinx Entitlement Management page. The URL in the address bar is https://xilinx.entitlenow.com/AcrossUser/main.gsp?product=&tab=&req_hash=&. The page has a dark blue header with the Xilinx logo and navigation links for "Solutions", "Products", and "Support". Below the header, there are two tabs: "Create New Licenses" (active) and "Manage Licenses". Under the "Create New Licenses" tab, there are two sections. The first section is titled "Have a Voucher to Redeem?" and includes a text input field for "enter voucher code" and a "Redeem Now" button. The second section is titled "Evaluation and No Charge Cores" and includes a "Search the Evaluation and No Charge cores catalog and add specific cores to table below" button.

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
<input type="checkbox"/> Vivado ML Enterprise Edition, 30-Day Evaluation License	Certificate - Evaluation	Node	1/1	Current	30 days
<input type="checkbox"/> 2021 AI Engine Tools License	Certificate - No Charge	Node	1/1	Current	None
<input type="checkbox"/> Vitis Model Composer (Xilinx toolbox for MATLAB and Simulink), 90 Day Evaluation	Certificate - Evaluation	Node	1/1	Current	90 days
<input type="checkbox"/> ISE Embedded Edition License	Certificate - No Charge	Node	1/1	Current	None
<input type="checkbox"/> SDSoC Environment, 60 Day Evaluation License	Certificate - Evaluation	Node	1/1	Current	60 days
<input type="checkbox"/> SDAccel OpenCL Development Environment: 30 Day Node Locked Evaluation Lice...	Certificate - Evaluation	Node	1/1	Current	30 days
<input type="checkbox"/> Vivado Design Suite: HL WebPACK 2015 and Earlier License	Certificate - No Charge	Node	1/1	Current	None
<input type="checkbox"/> ISE WebPACK License	Certificate - No Charge	Node	1/1	Current	None
<input type="checkbox"/> Xilinx MicroBlaze/All Programmable SoC Software Development Kit – Standalone	Certificate - No Charge	Node	1/1	Current	None
<input type="checkbox"/> PetaLinux Tools License	Certificate - Evaluation	Node	1/1	Current	365 days

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

Solutions Products Support XILINX

Create New Licenses Manage Licenses

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

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

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.

Vivado License Manager 2020.1

File Help

VIVADO License Manager XILINX

- Get License
 - Set Proxy
 - Obtain License
 - Load License
- Manage License
 - Manage License Search Paths
 - View License Status
- View System Information
 - View Host Information

Load License

Select License File

Look In: Downloads

Xilinx.lic

File Name:

Files of Type: License files (*.lic)

Open Cancel

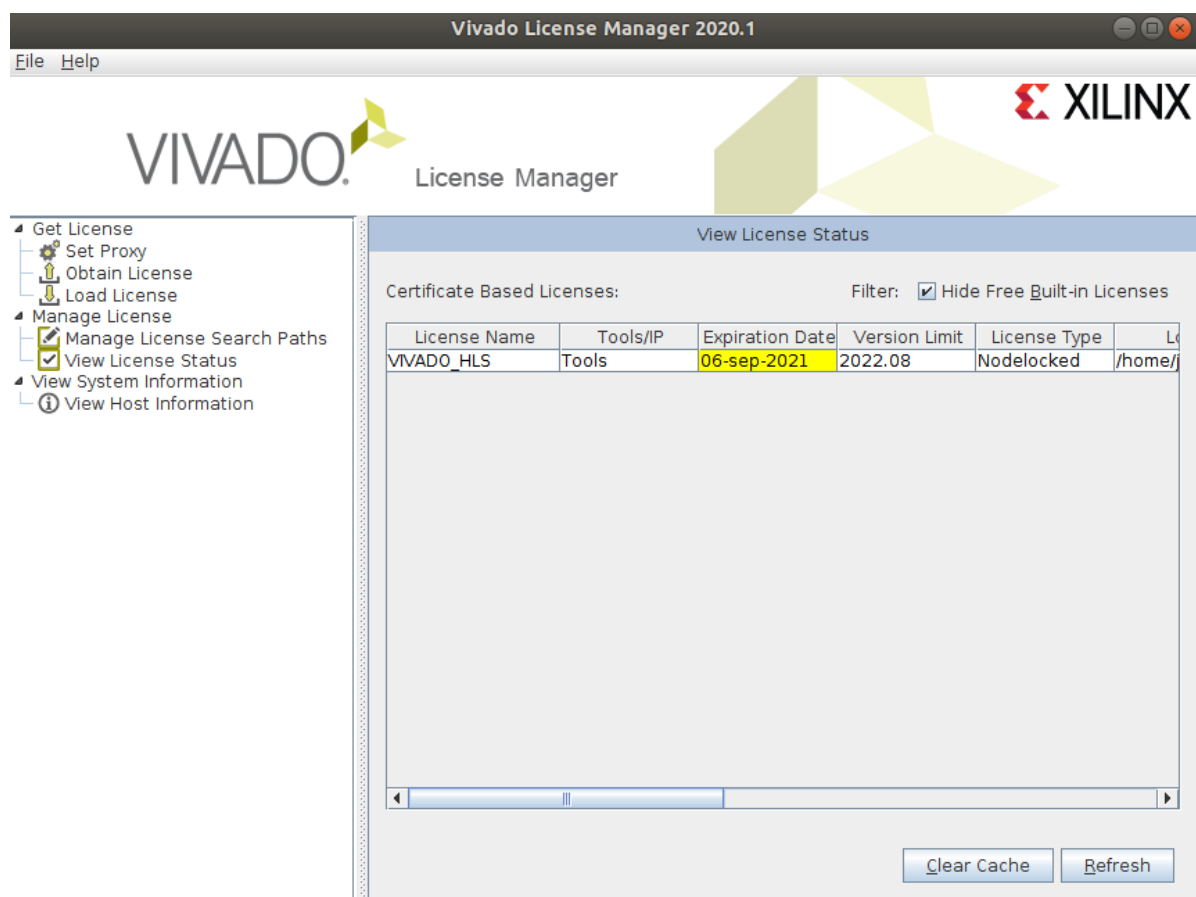


Figure 9

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.

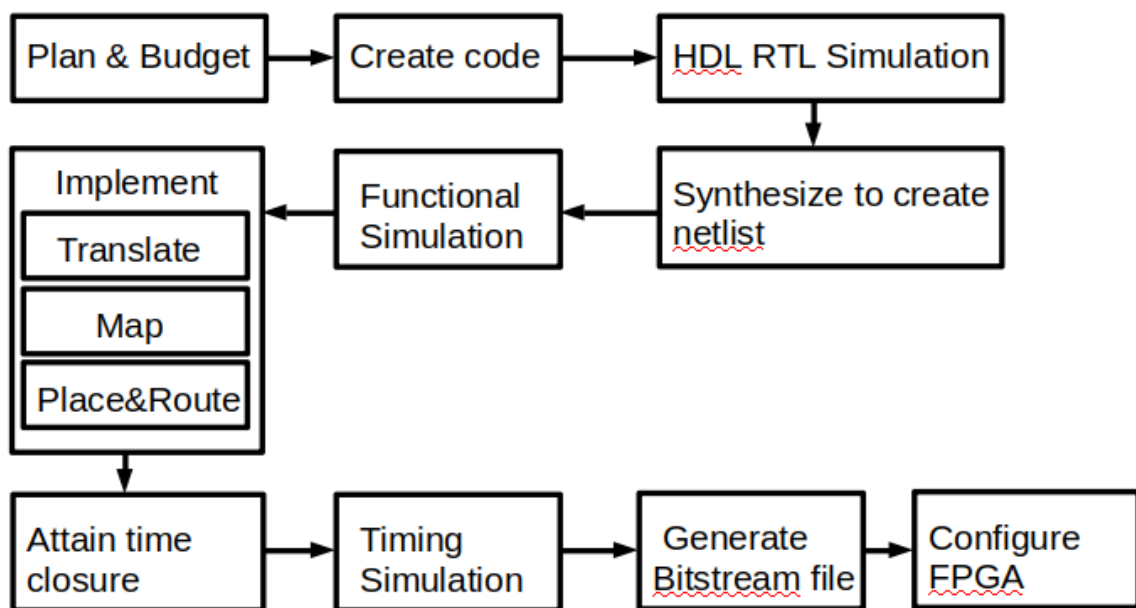
Maybe there are students who are not familiar with Verilog and VHDL, two popular hardware description languages. Don't worry, Here are some websites helping you get start with these.

Lab 2 Verilog & VHDL:

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

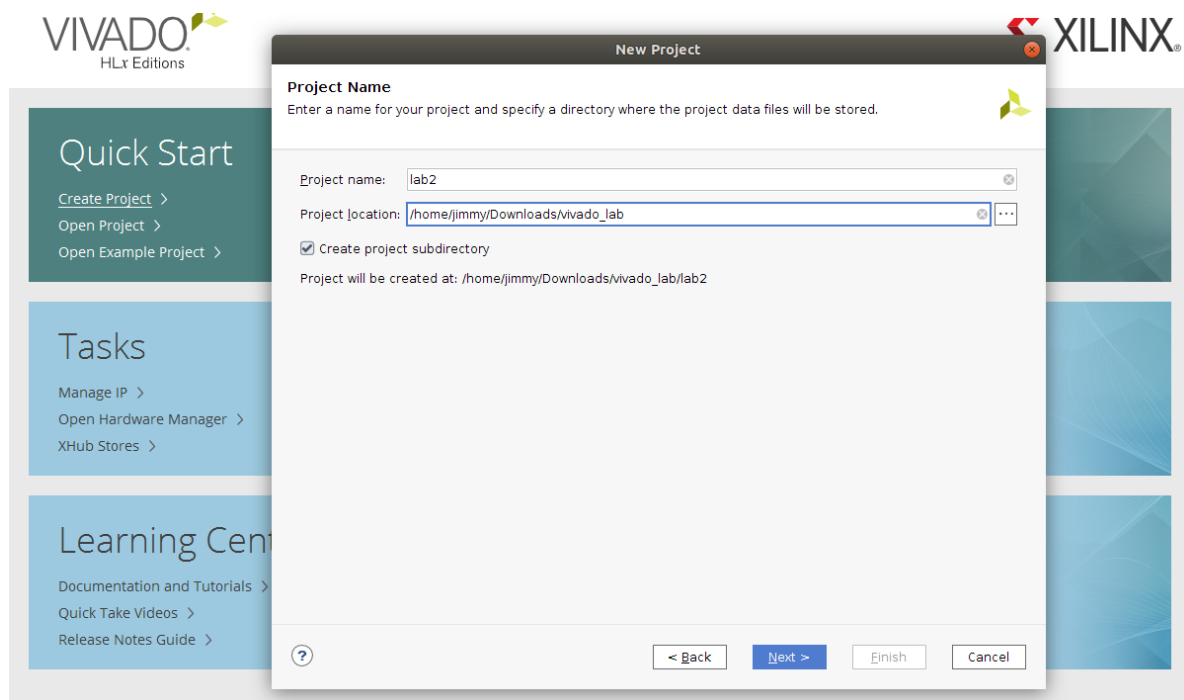
Shown in Figure A typical design flow consists of model(s) creating, user constraint creating file(s), Vivado project creating, the created models importing, assigning created constraint file(s), optionally running behavioral simulation, synthesizing the design, implementing the design, generating the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



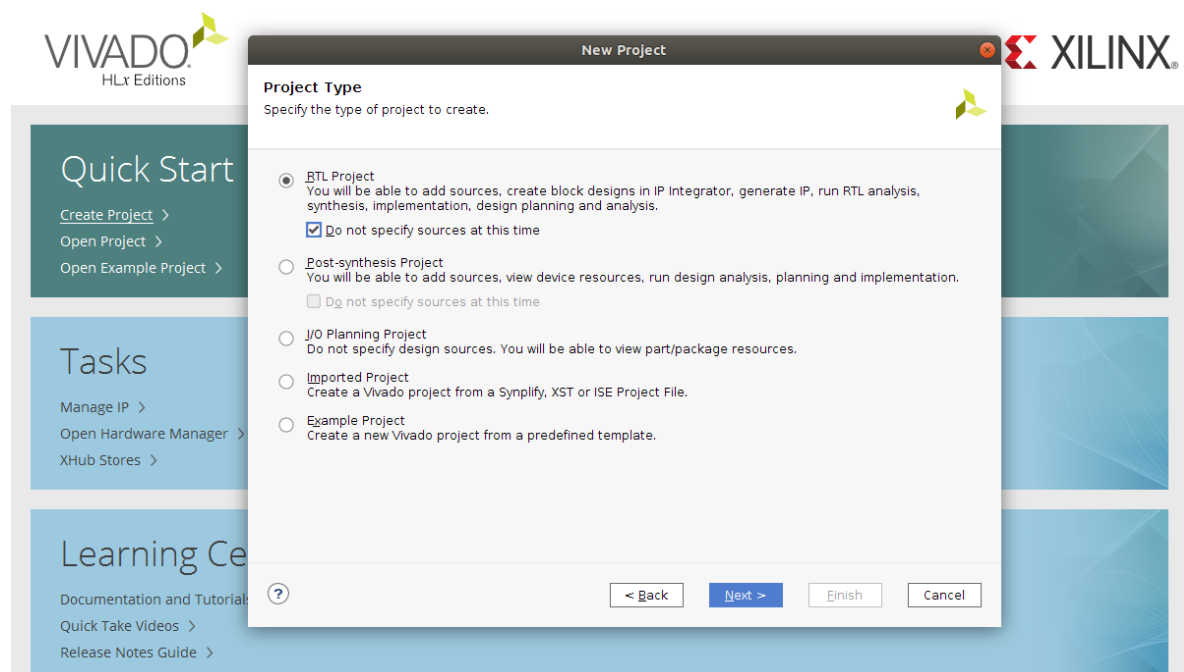
xczu29dr-fsvf1760-1L-i

Project creating

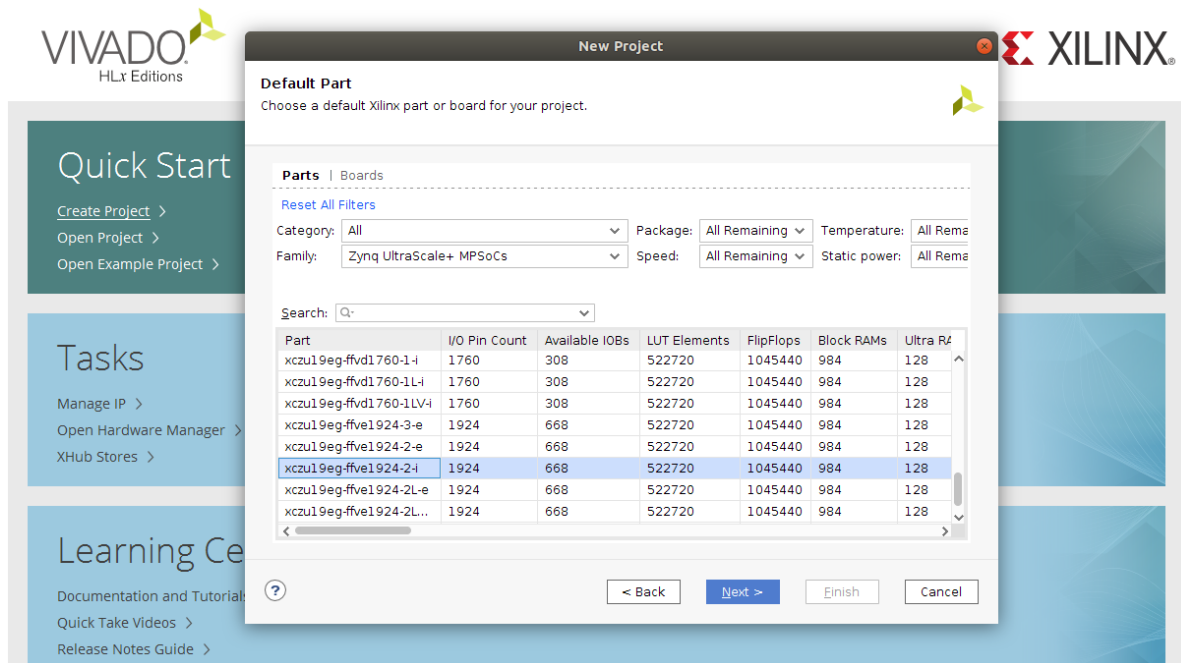
Figure to Figure is the new project, lab2's creating process. You can locate the project on the **vivado_lab** folder, which is provided and can be downloaded from the github.



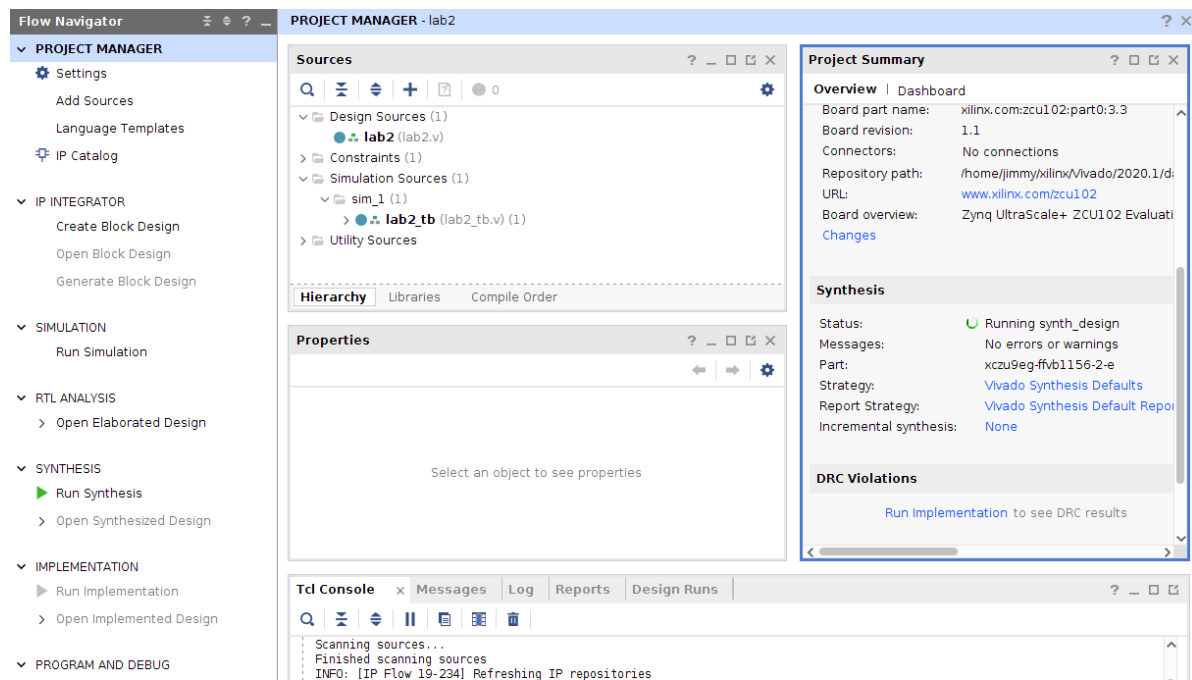
Choose RTL project as the project type.



After the type selection, decide the board part.

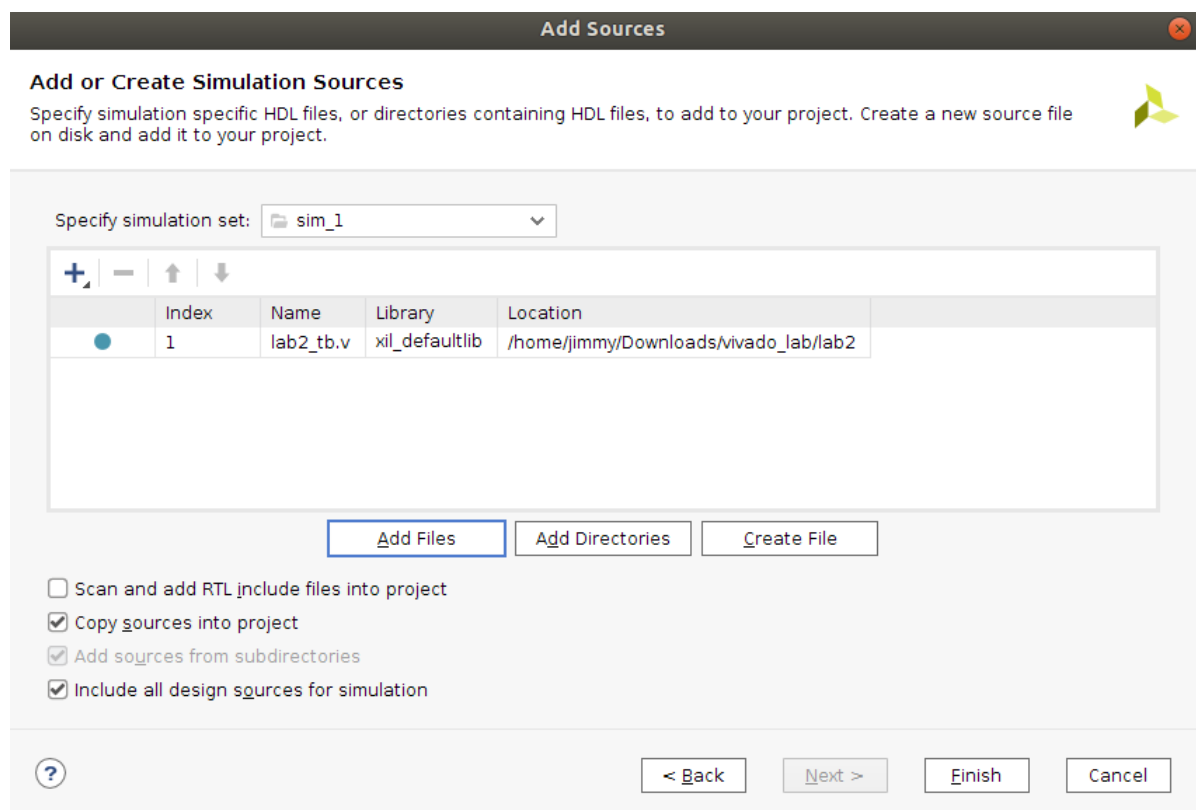


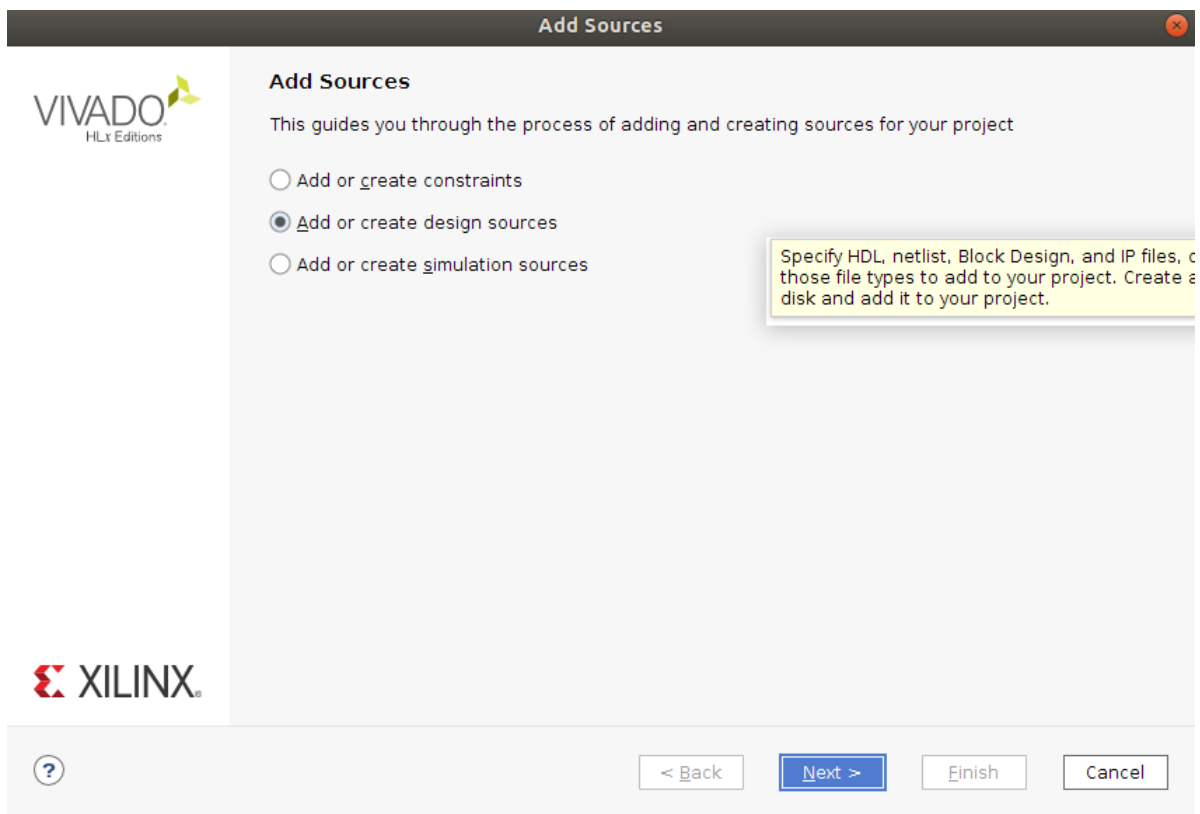
We can see the layout of the project platform and the sources hierarchy.





Source Adding





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 simulation. 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.

```

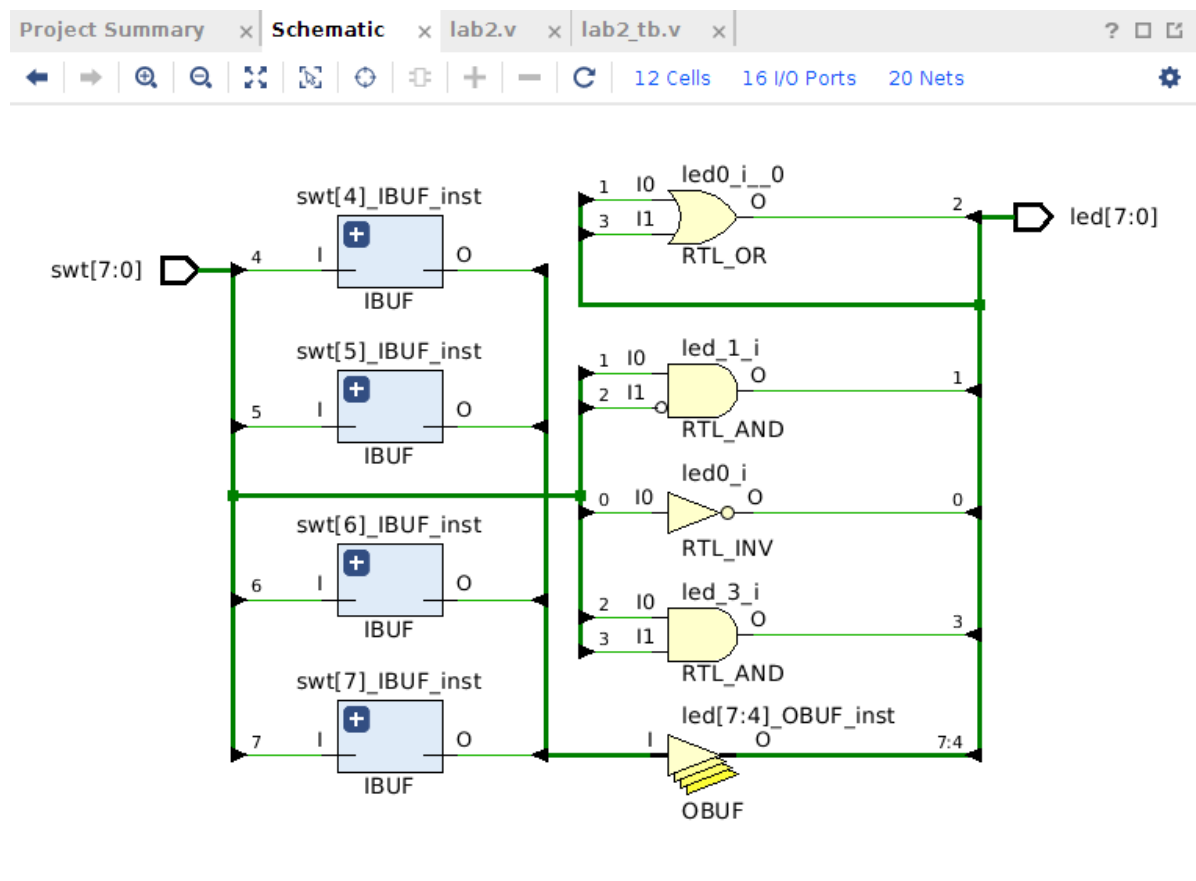
input [7:0] swt,
output [7:0] led

```

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 .

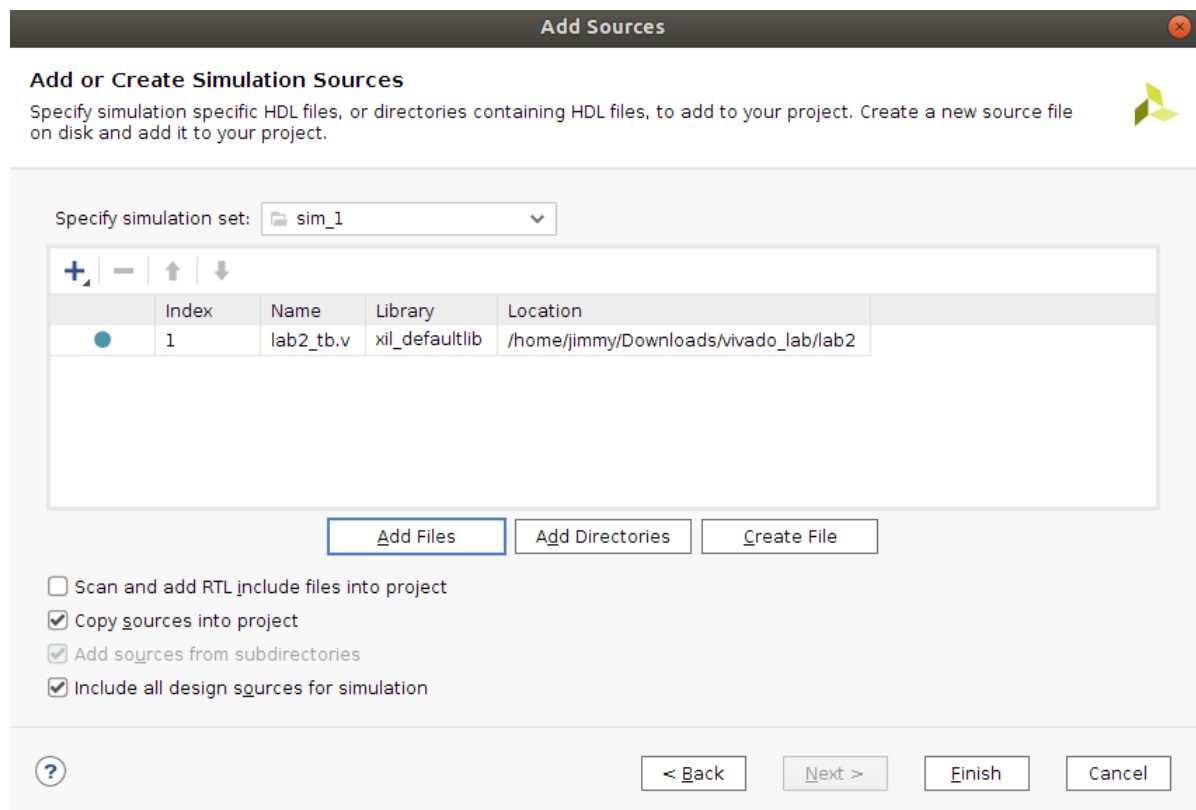
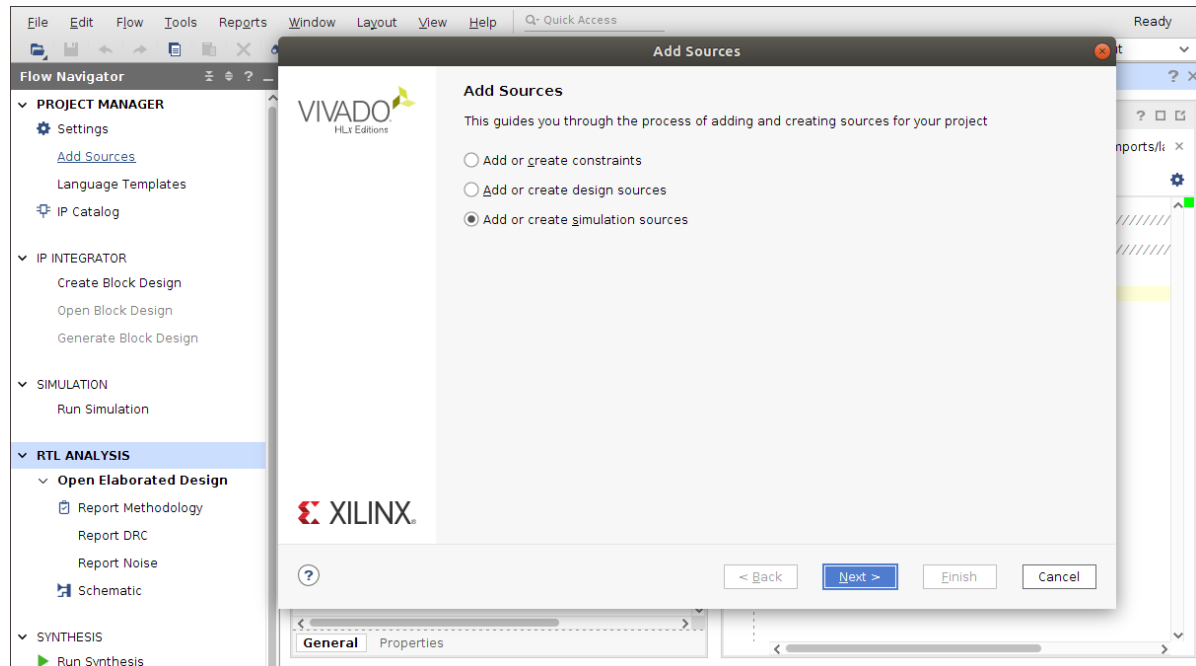


To validate the performance of the module, a testbench script file is needed.

2.2 Testbench

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.



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
    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(swatches));
```

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 swatches=i;
        #10 e_led = expected_led(swatches);
        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
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 console**.

Simulation

Lab 3 Convolutional Layer

3.1 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. For example, in synthesis imaging, the measured dirty map is a convolution of the "true" CLEAN map with the dirty beam (the [Fourier transform](#) of the sampling distribution).

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

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

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

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

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). The feature can later be used for further deep learning training or traditional computer vision object recognition and tracking.

edge1

In image process, Features which distinguish them from the surrounding display the color intensity change.

3.2 Task(U)

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

$$M_{11} = A_{11} * B_{11}, M_{12} = A_{12} * B_{12}, M_{13} = A_{13} * B_{13};$$

$$M_{21} = A_{21} * B_{21}, M_{22} = A_{22} * B_{22}, M_{23} = A_{23} * B_{23};$$

$$M_{31} = A_{31} * B_{31}, M_{32} = A_{32} * B_{32}, M_{33} = A_{33} * B_{33};$$

2. Summation of the multiplication

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

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 continuous convolution between matrixes. 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.

MAC

Multiplex

Add

Matrix