

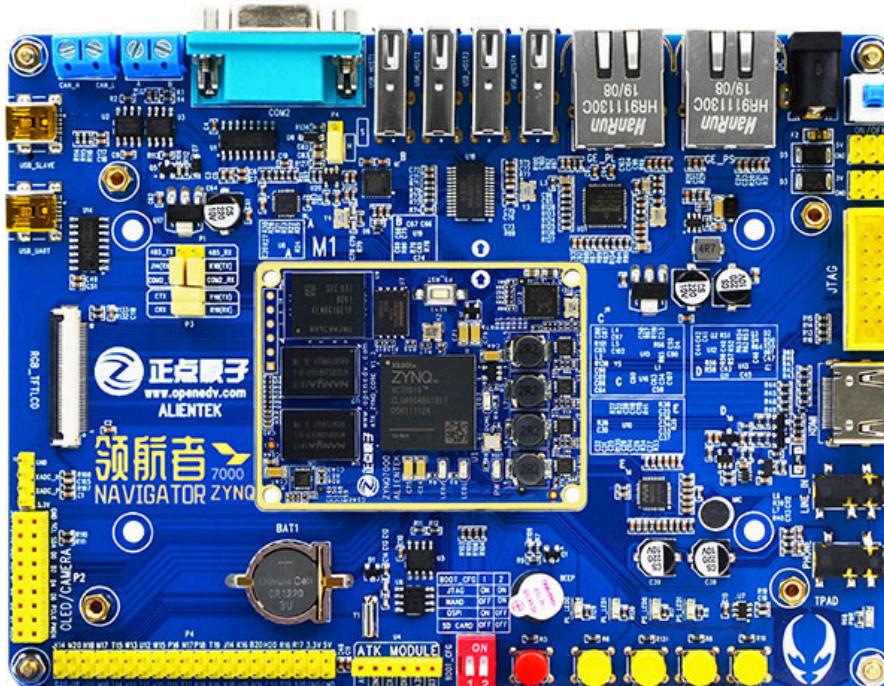
tinyTPU Project Artifacts

How to start

All the codes can be download from <https://github.com/Zongweihen/tinyTPU.git>

Hardware Design

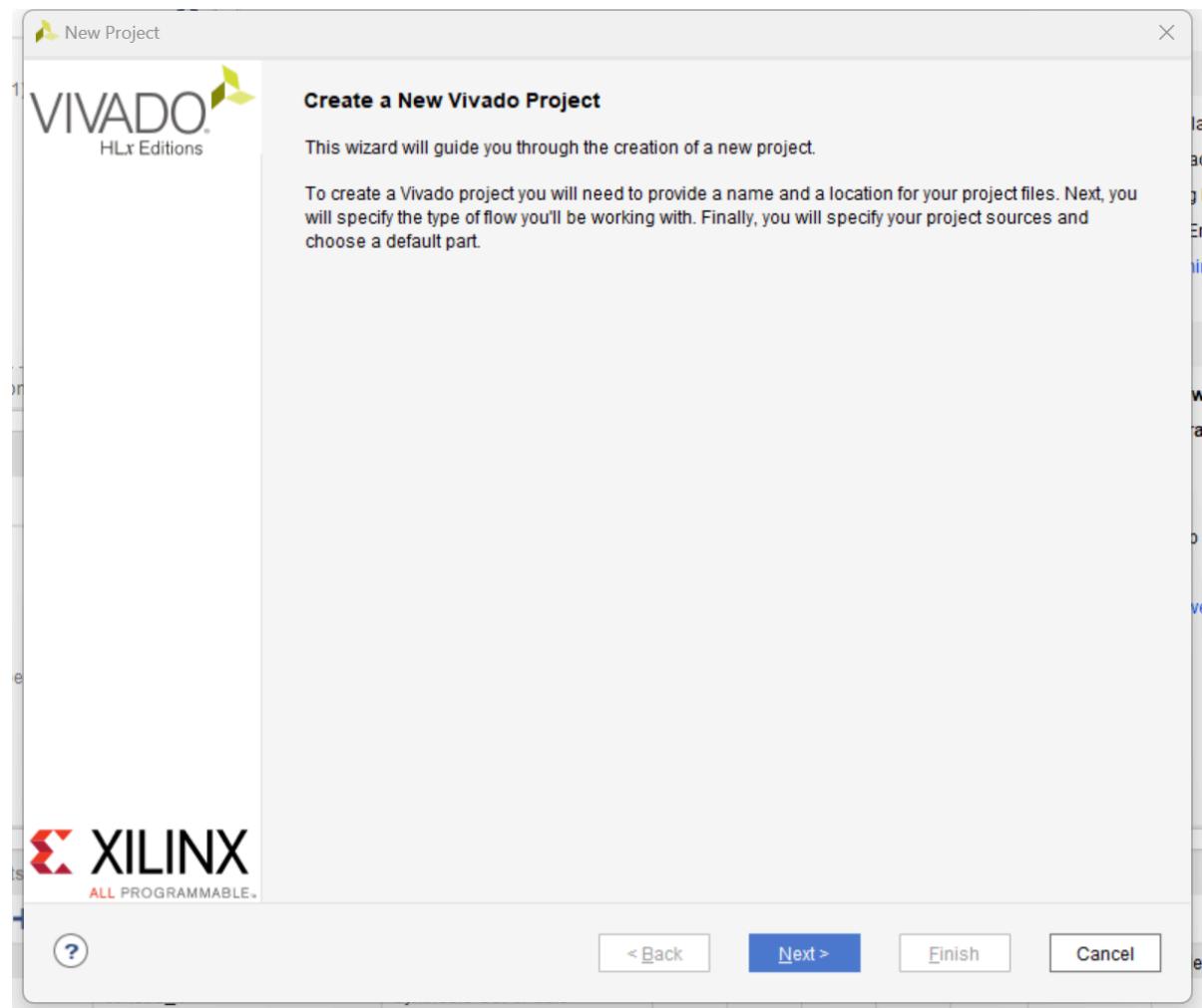
Hardware Platform Introduction



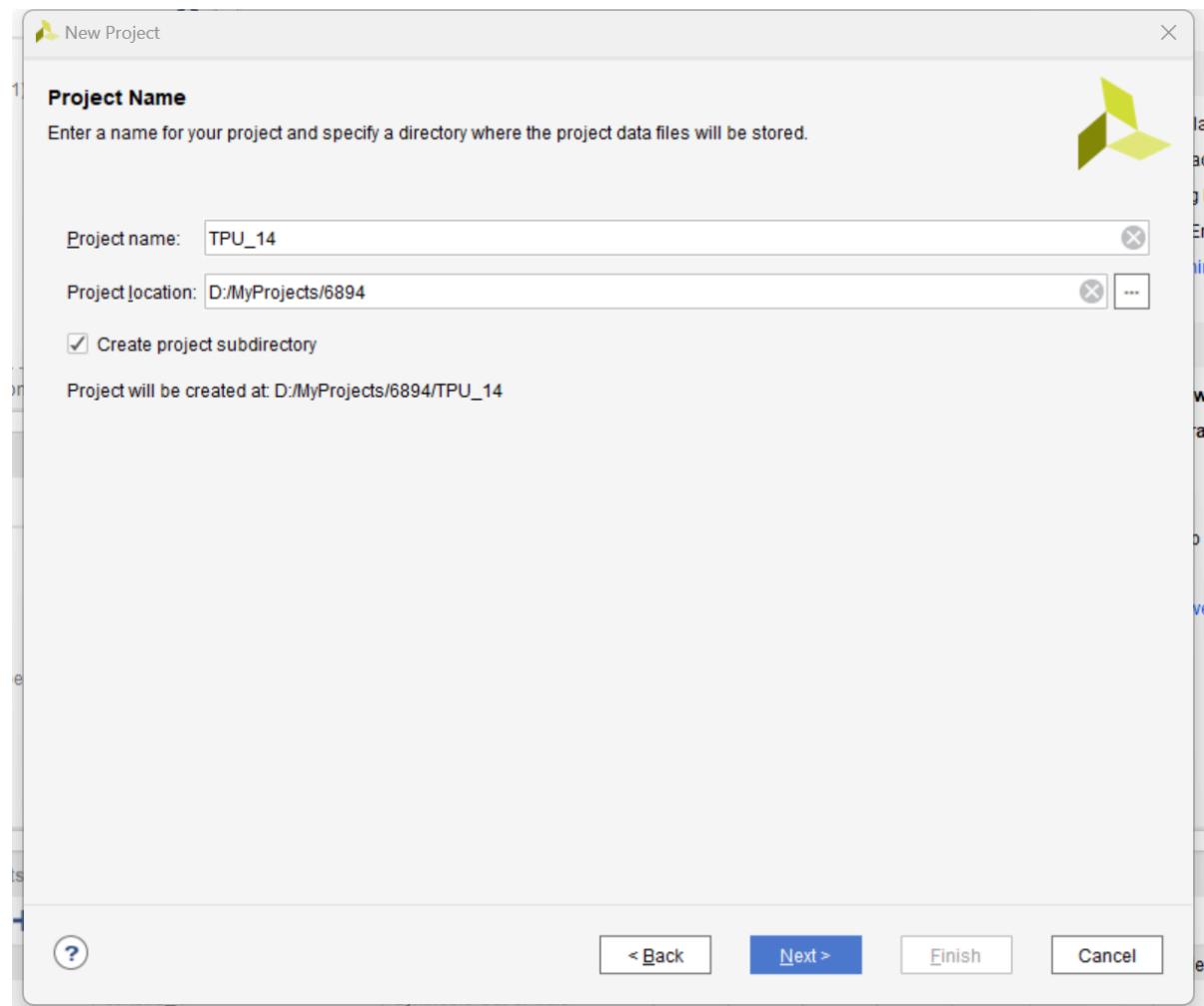
Hardware platform is Navigator ZYNQ7000 board from ALENTEX v1. Development platform is Vivado 2017.4. You can find this board in www.openedv.com.

Create a Project

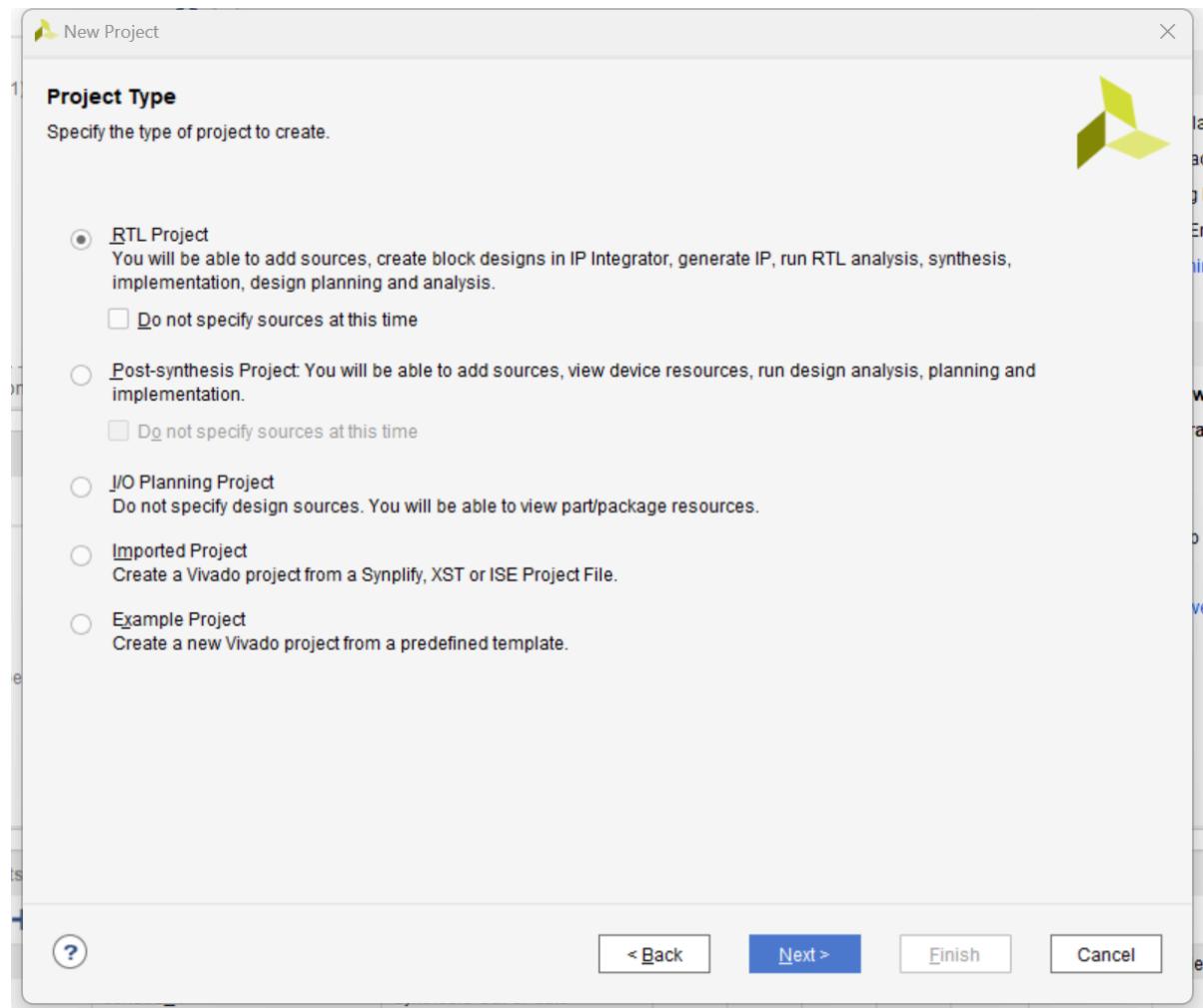
Click file -> new project or click create new project in main page.



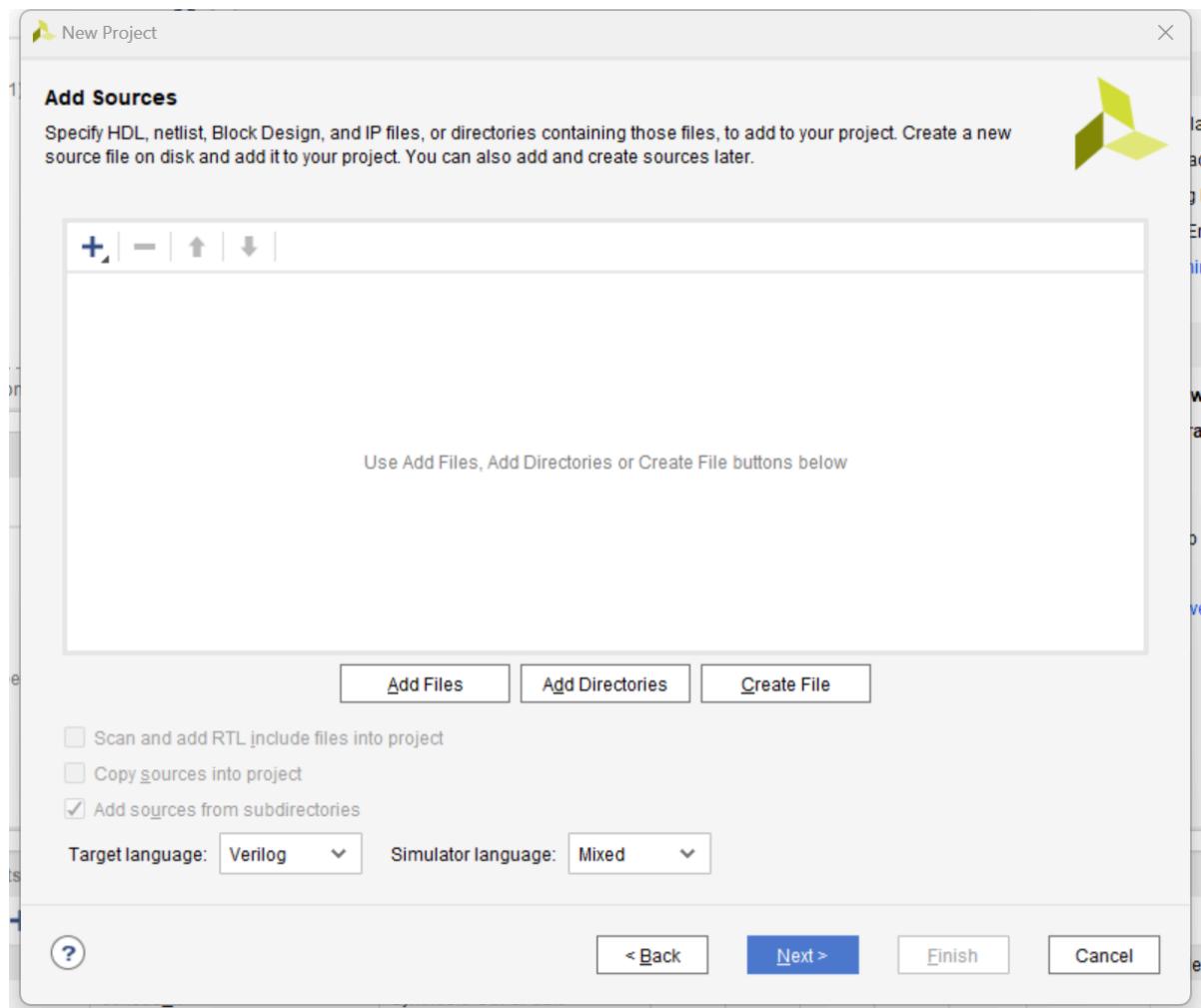
Click next.



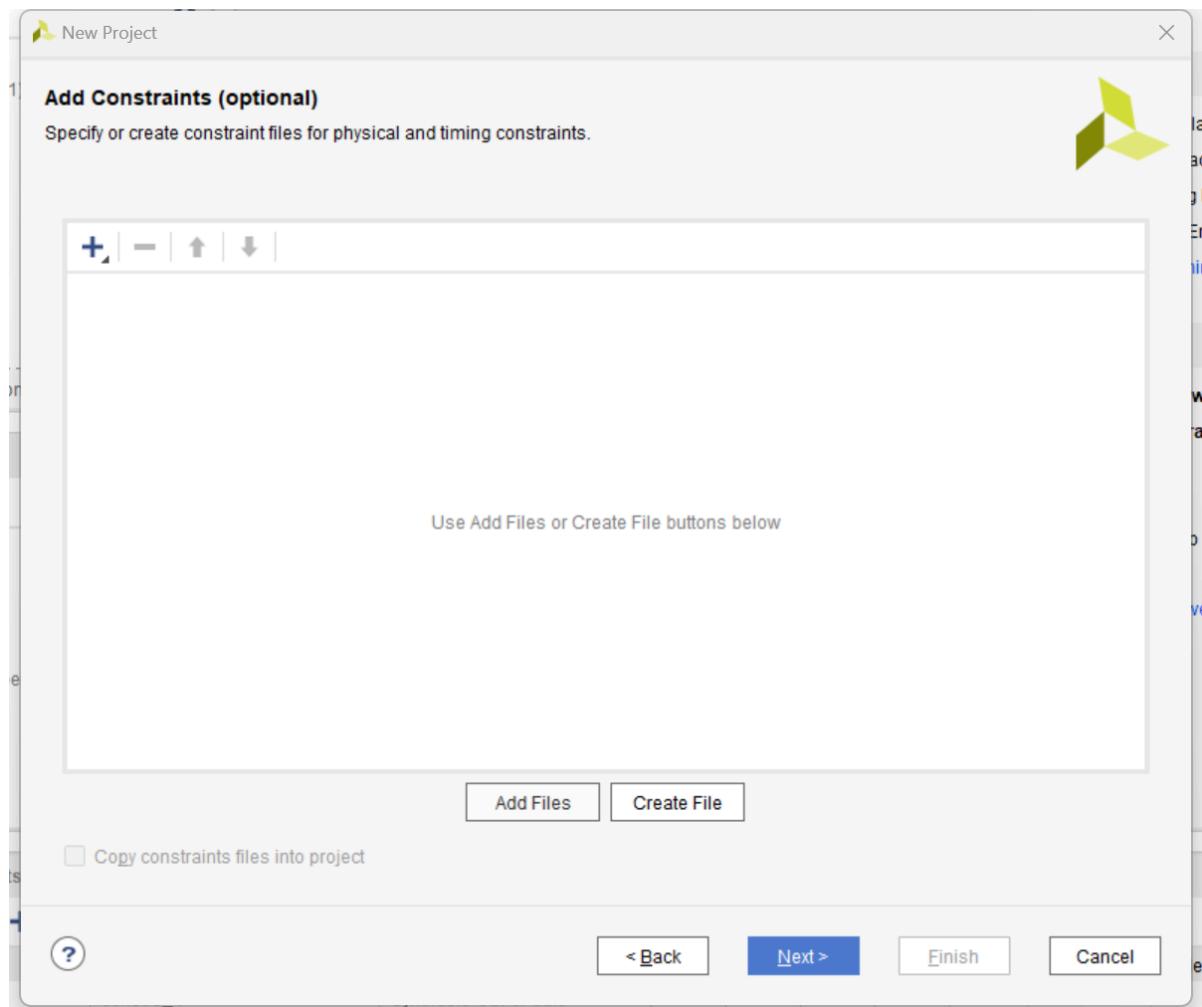
Rename your project as TPU_14 and press next.



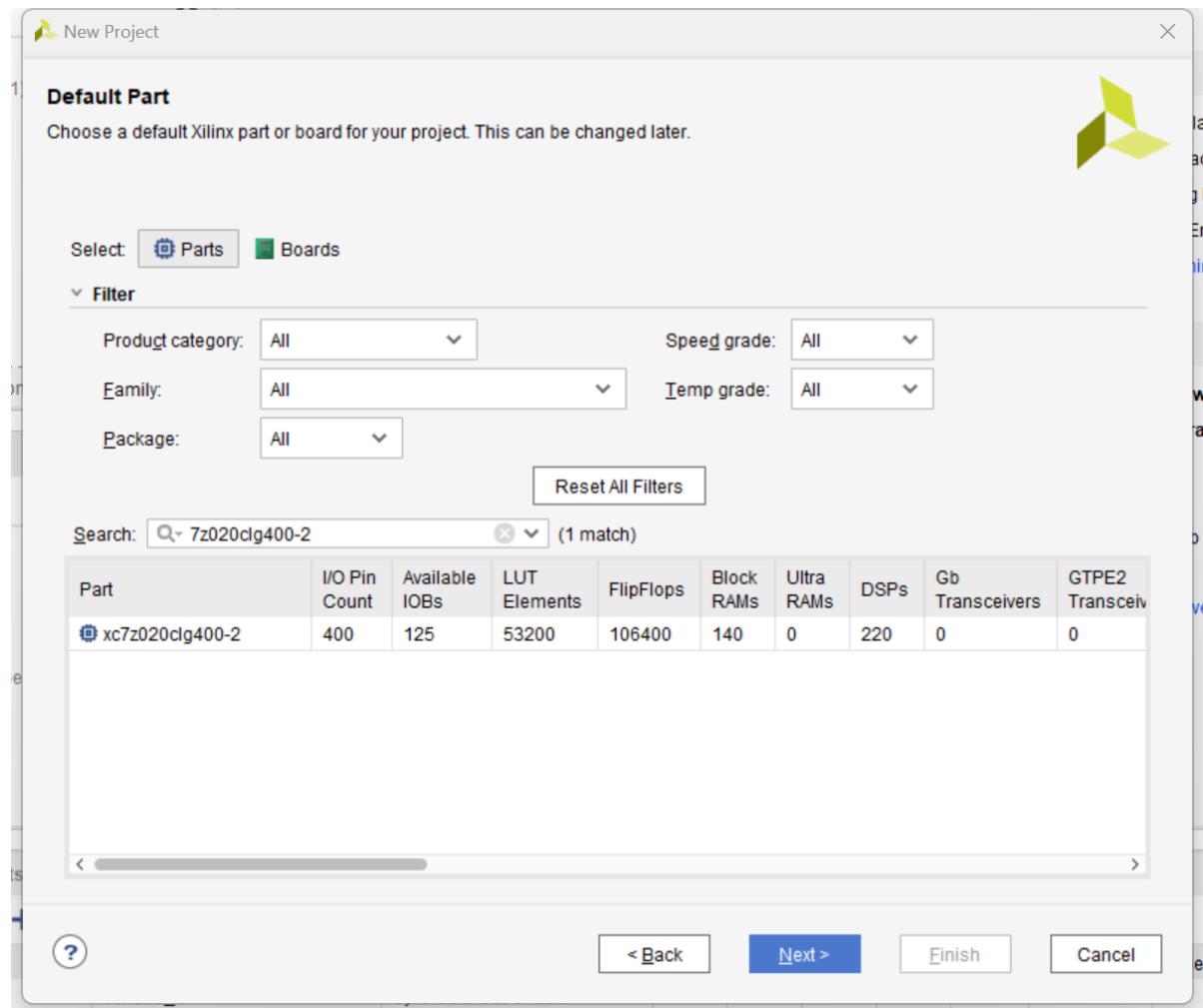
Choose RTL Project, and click next.



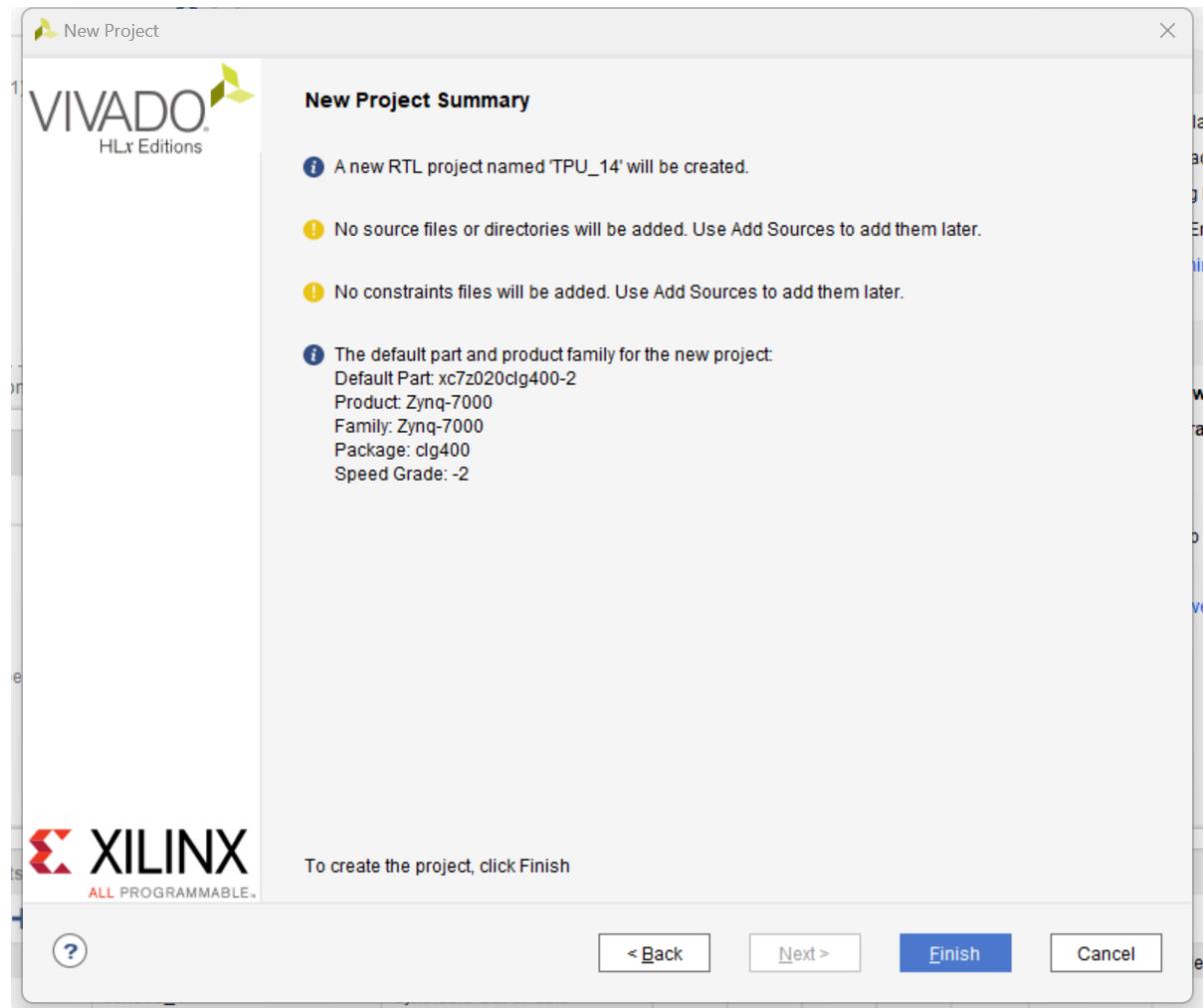
Click next.



Click next.



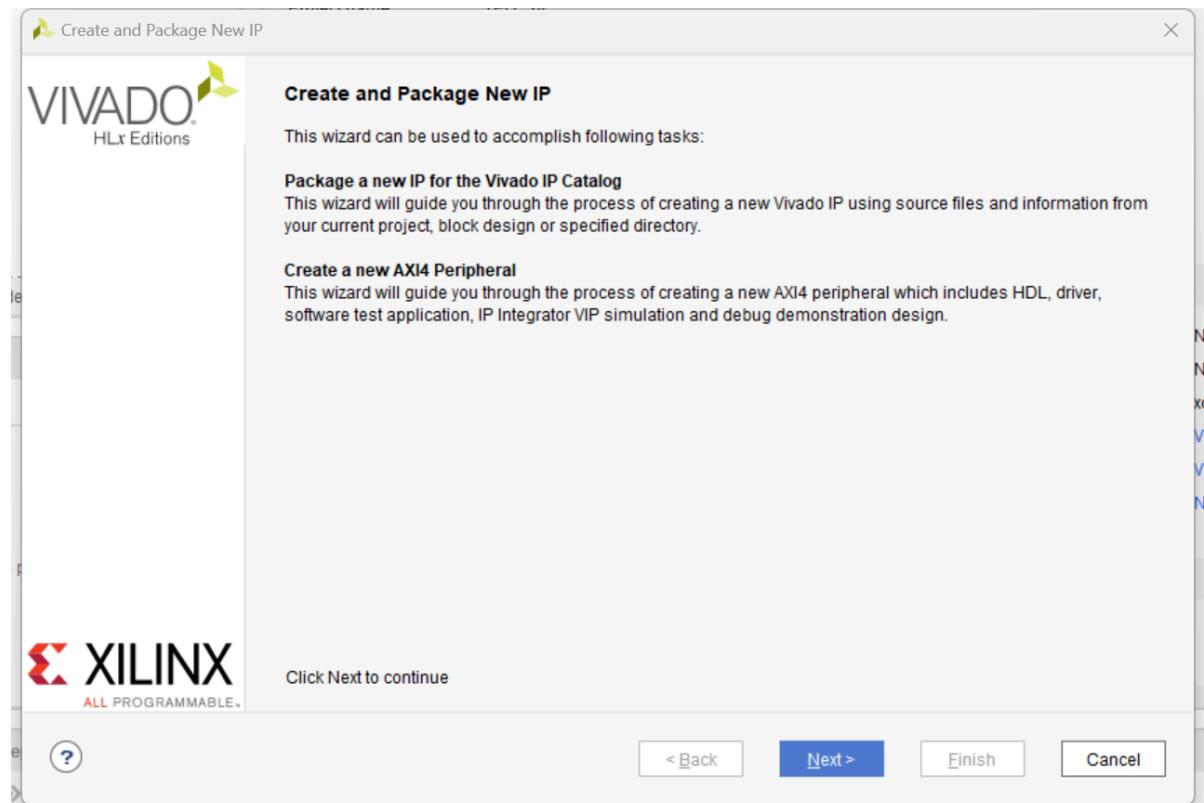
Choose the chip's Part name: xc7z020clg400-2, and choose the part and press next.

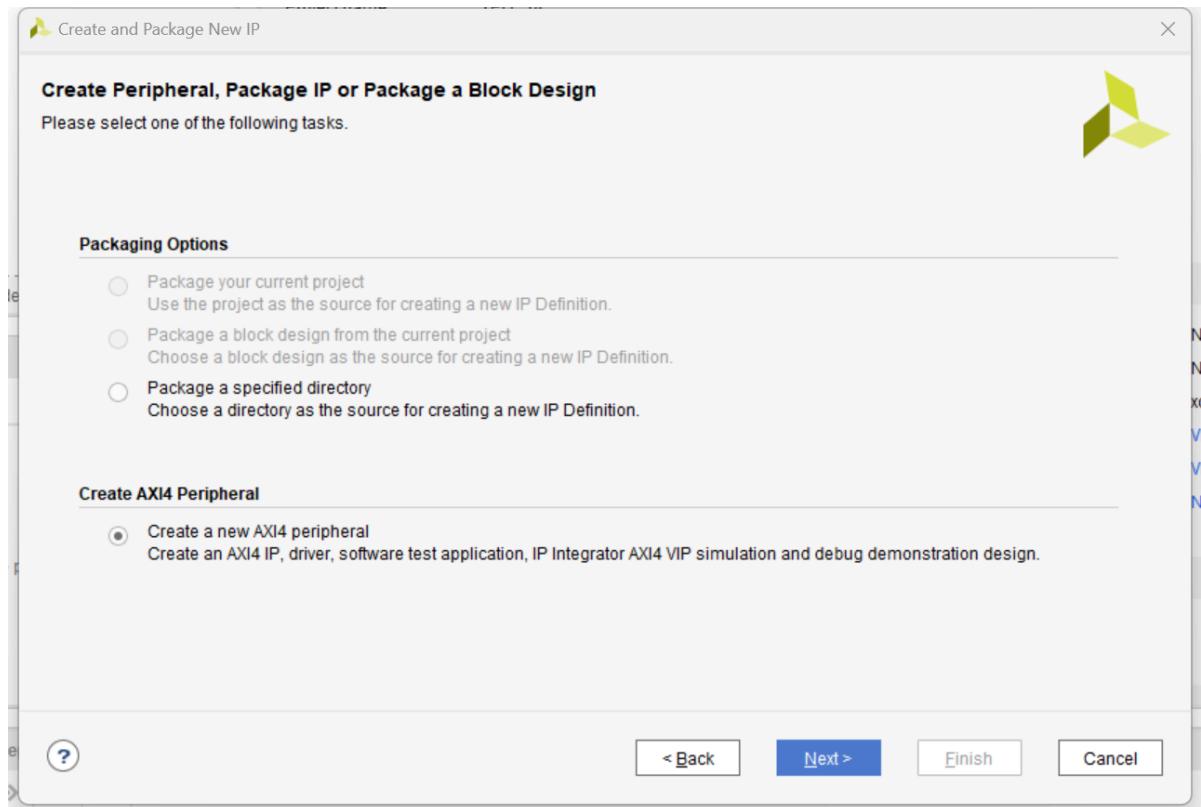


Click finish, and you will create one empty project.

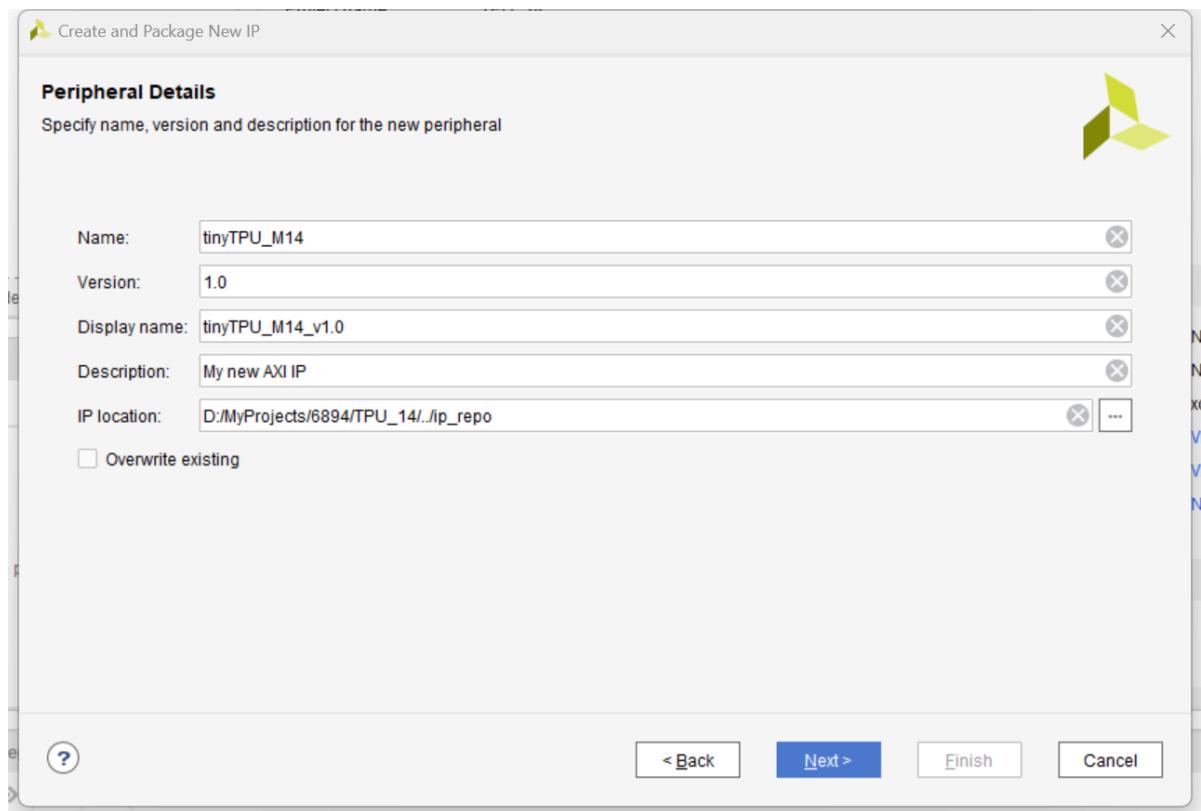
Create IP

Click on Tools -> Create and Package new IP and click Next.

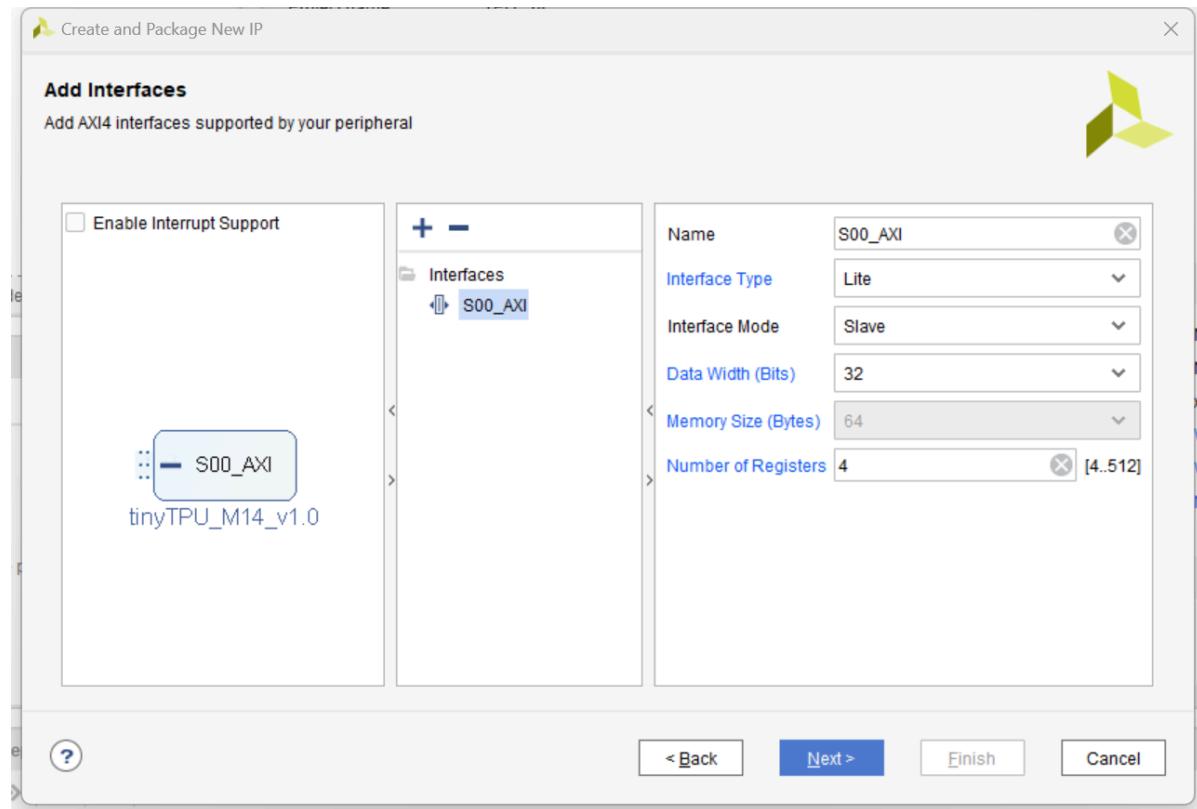




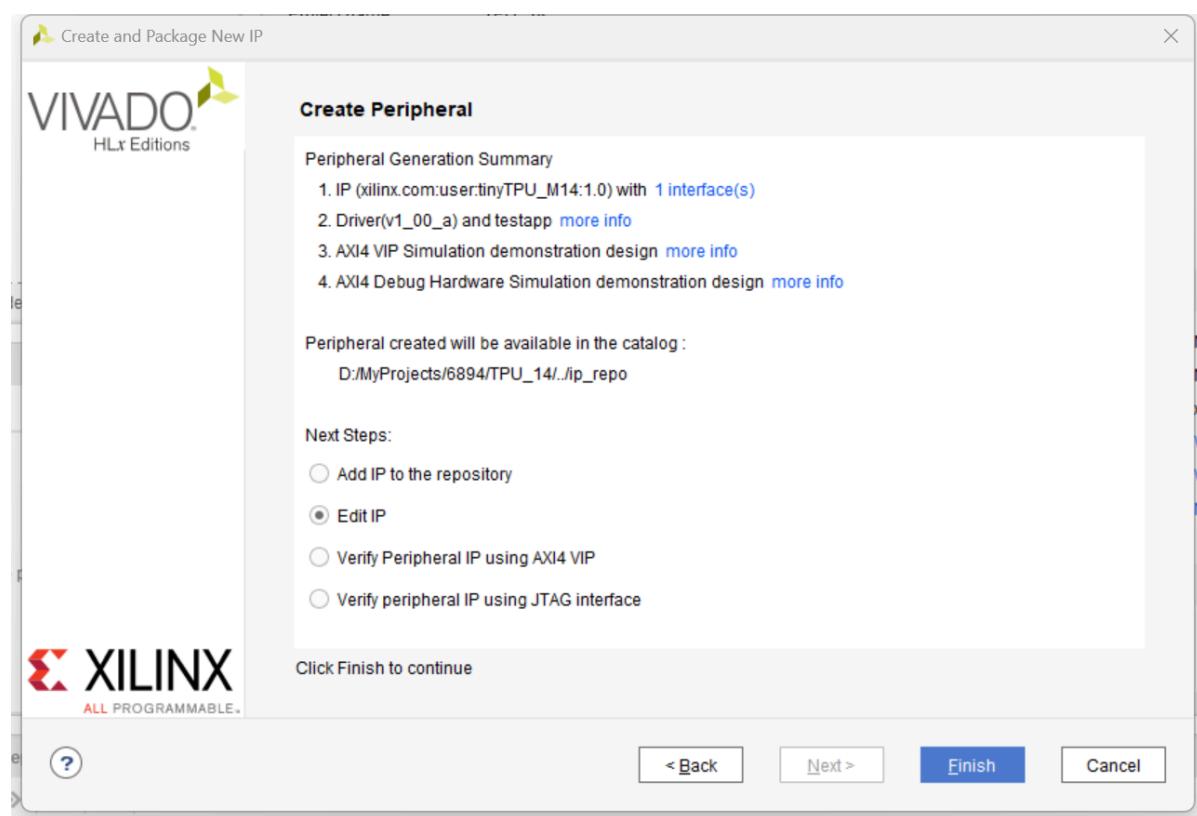
Choose Create a new AXI4 peripheral and click Next.



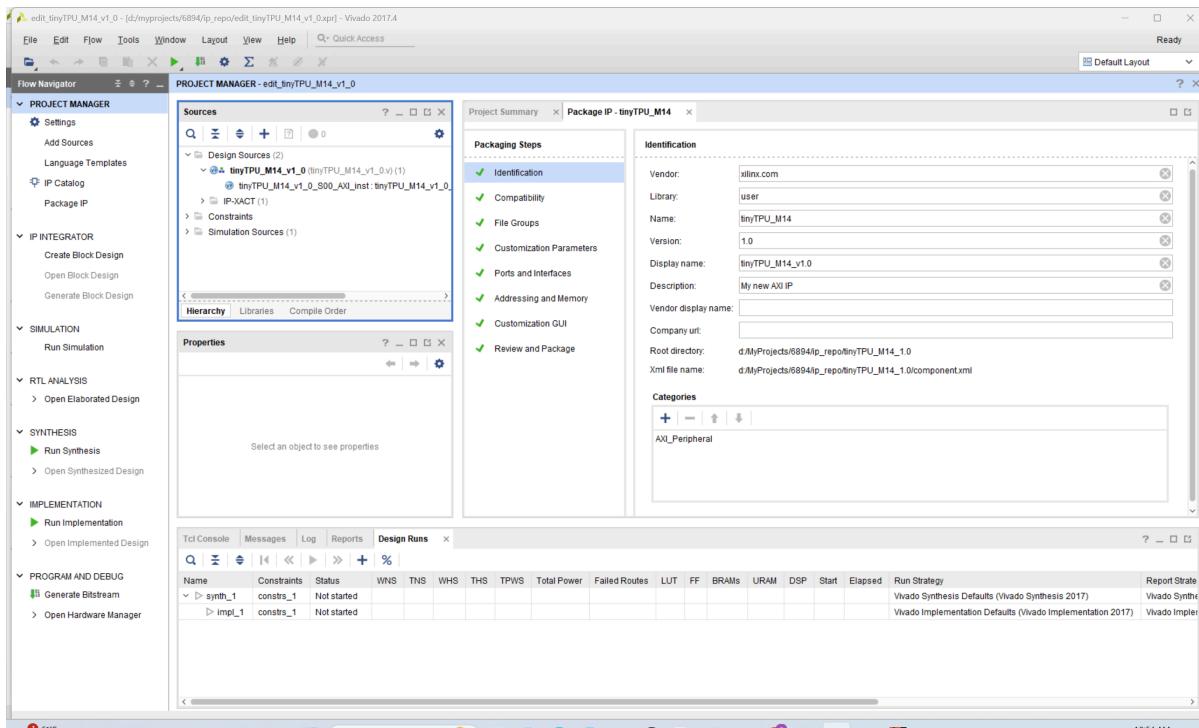
Rename our IP as tinyTPU_M14 and keep others default. Then click Next.



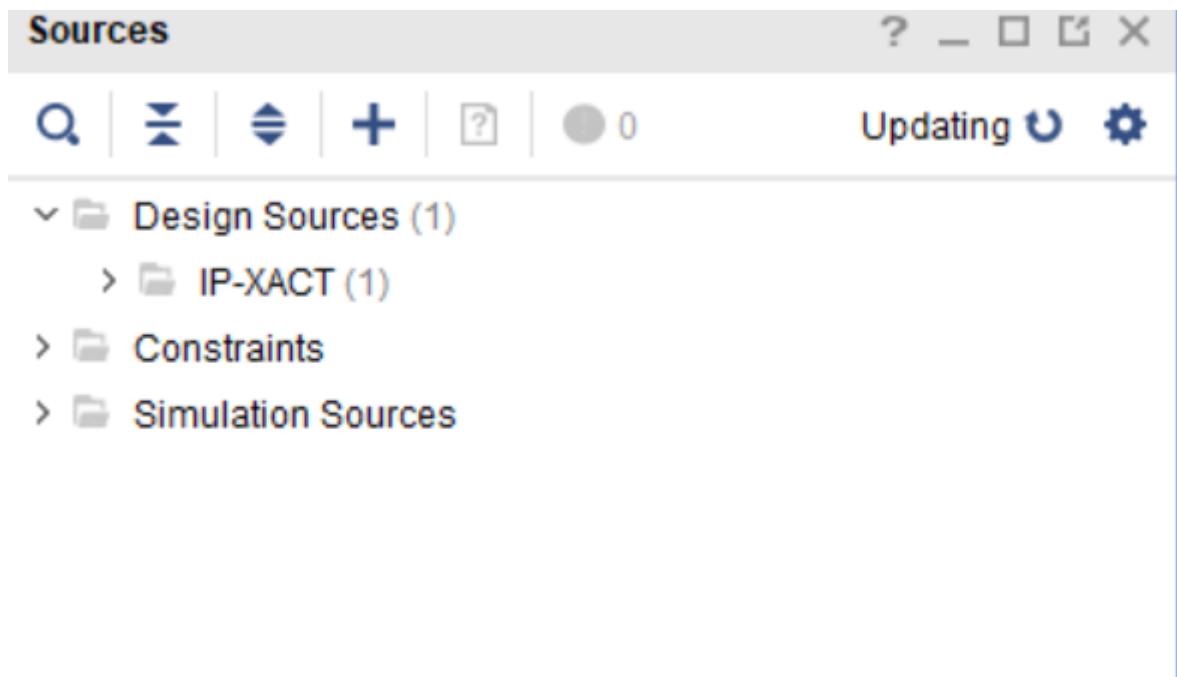
Keep these parameters as default and click Next.



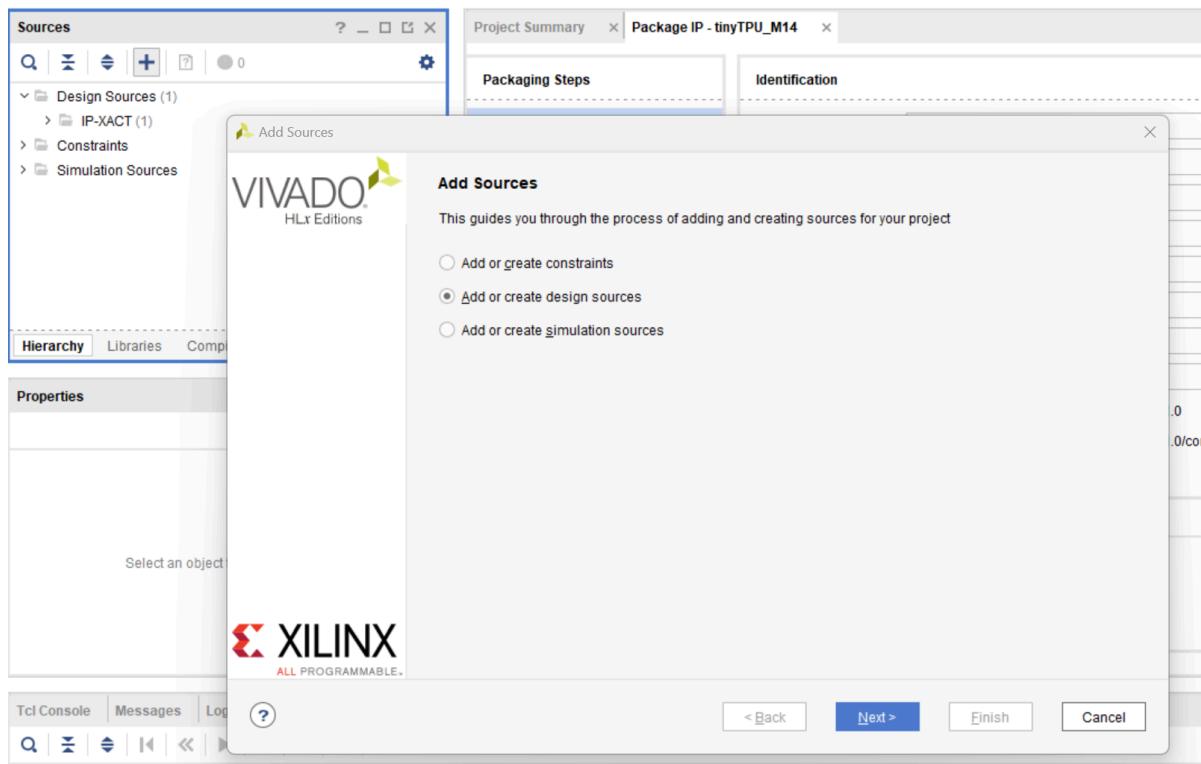
Choose Edit IP and click finish.



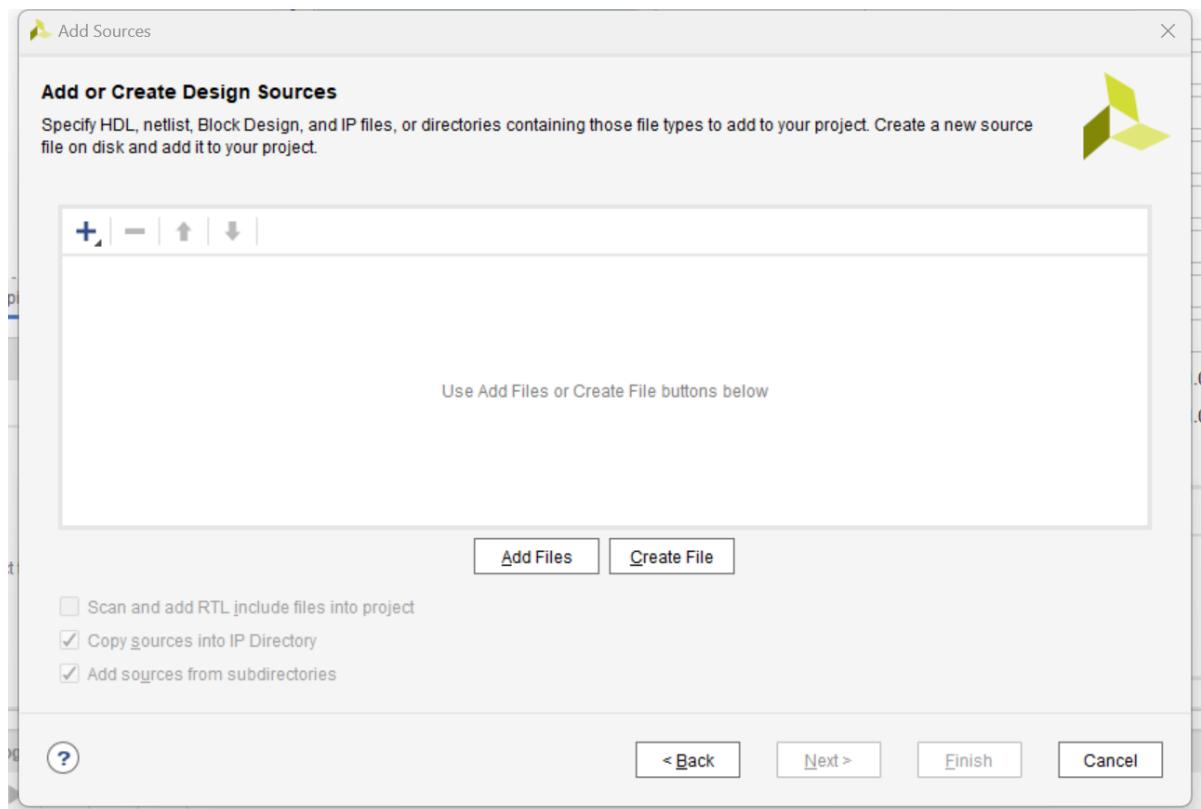
Then, you will find this window will be opened. There will be two generated verilog files, delete them.



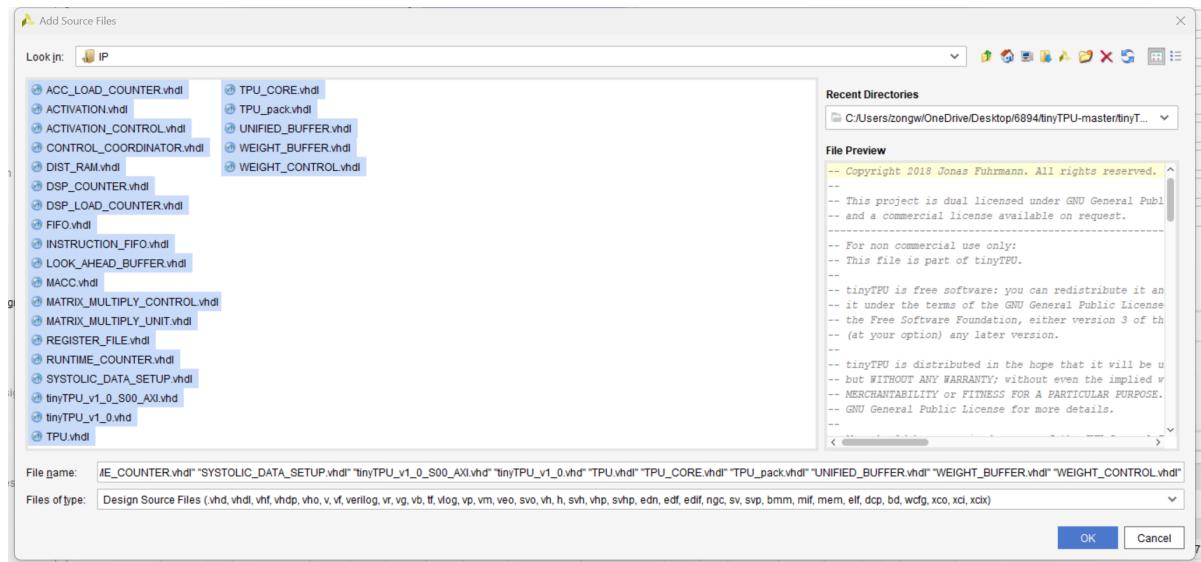
Just right click the files and choose delete and then you will see the empty Sources.



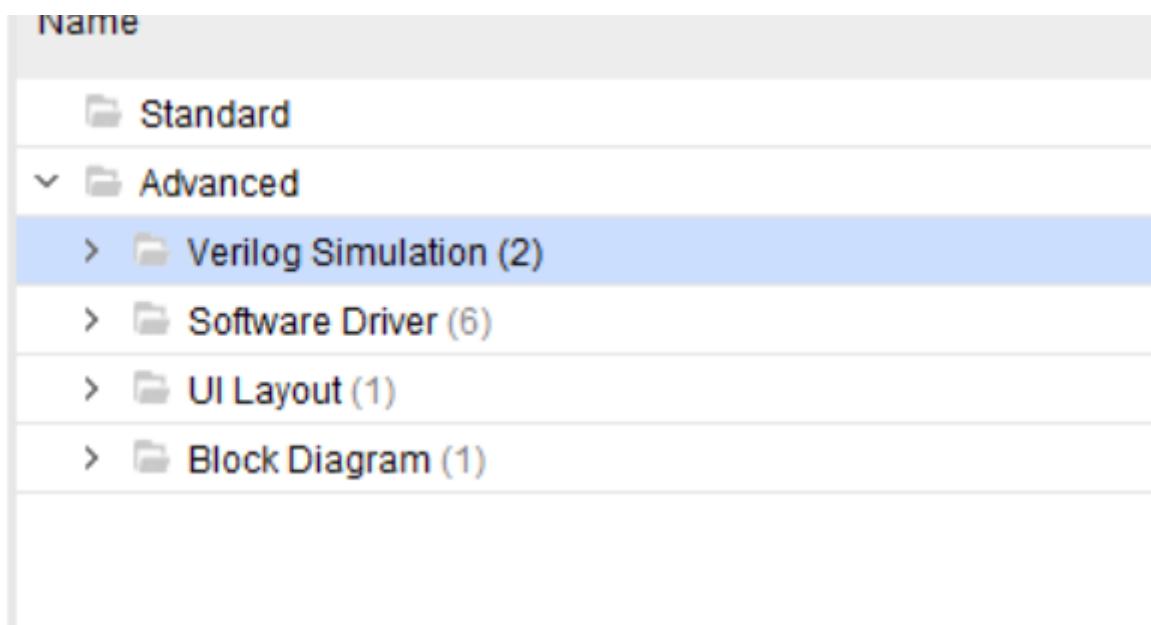
Then, click the "+" to add our VHDL IP source. Choose Add or create design sources, then click Next.



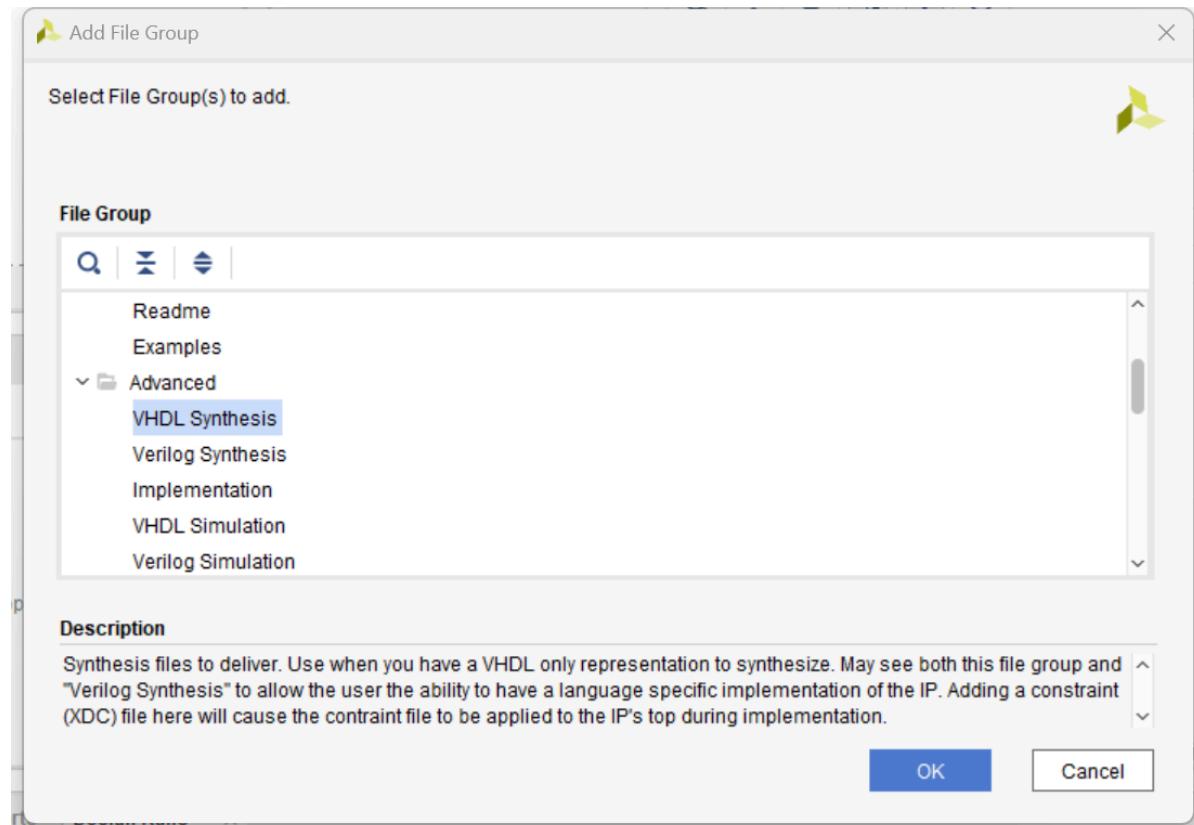
Click Add Files.



Select all vhdl files down loaded from github and click Ok.



Then you must remove Verilog SImulation group and Verilog Synthesis group.



Then add new VHDL Synthesis group and click OK.

Merge changes from File Groups Wizard						
Name	Library Name	Type	Is Include	Used In Constant	Model Name	
Standard			<input type="checkbox"/>	<input type="checkbox"/>		
Advanced			<input type="checkbox"/>	<input type="checkbox"/>		
Software Driver (6)			<input type="checkbox"/>	<input type="checkbox"/>		
UI Layout (1)			<input type="checkbox"/>	<input type="checkbox"/>		
Block Diagram (1)			<input type="checkbox"/>	<input type="checkbox"/>		
VHDL Synthesis (0)			<input type="checkbox"/>	<input type="checkbox"/>	tinyTPU_M14_v1_0	

Rename model name as tinyTPU_M14_v1_0.

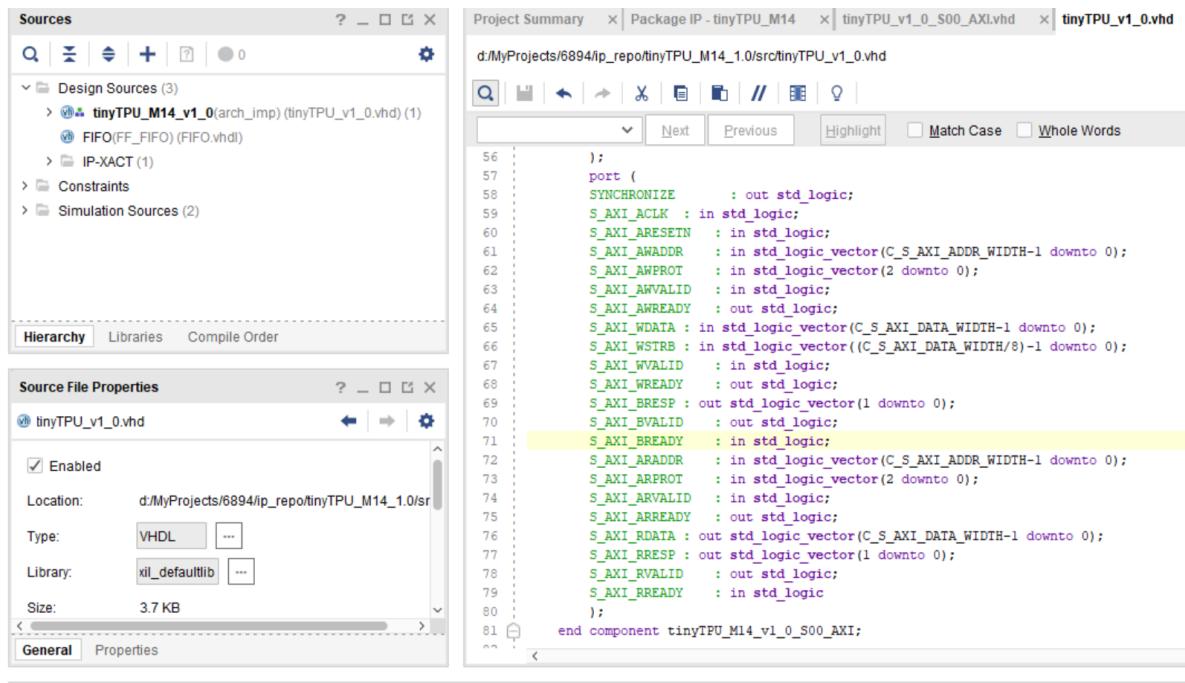
```

S_AXI_RVALID    : out std_logic;
-- Read ready. This signal indicates that t
-- accept the read data and response in
S_AXI_RREADY    : in std_logic
;
end tinyTPU_M14_v1_0_S00_AXI;

architecture arch_imp of tinyTPU_v1_0_S00_AXI is
-- tinyTPU logic

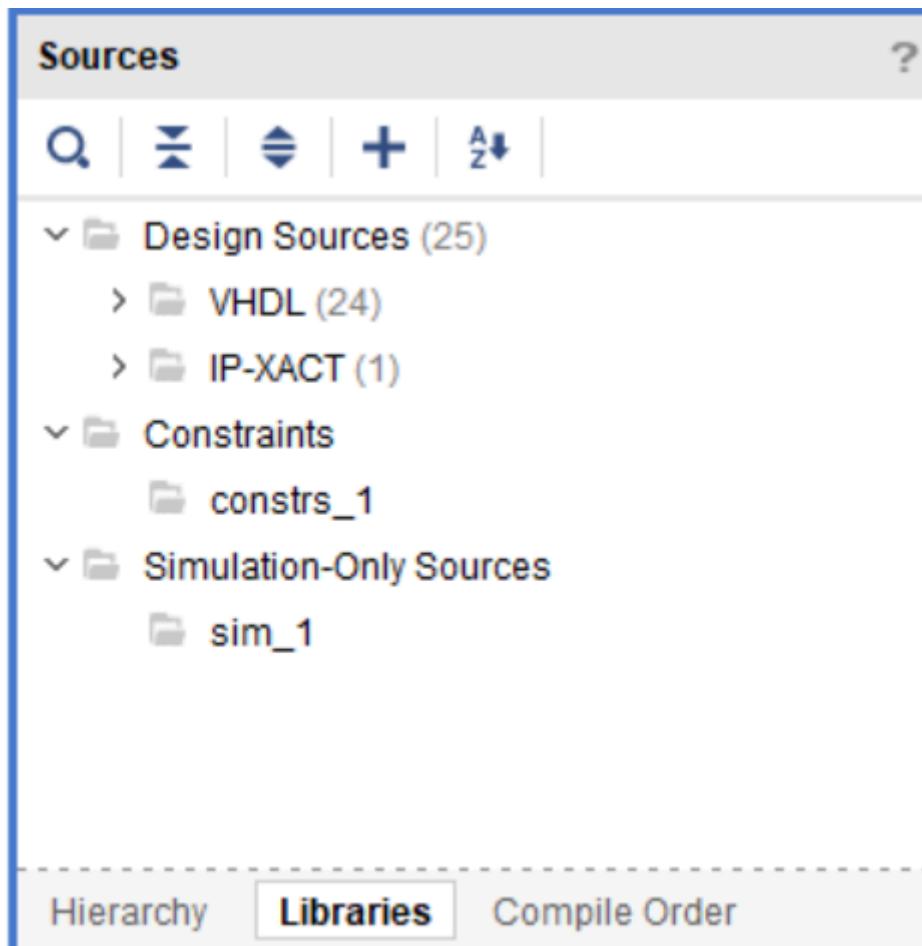
```

Add _M14 in all tinyTPU_V1_0 like this in picture.

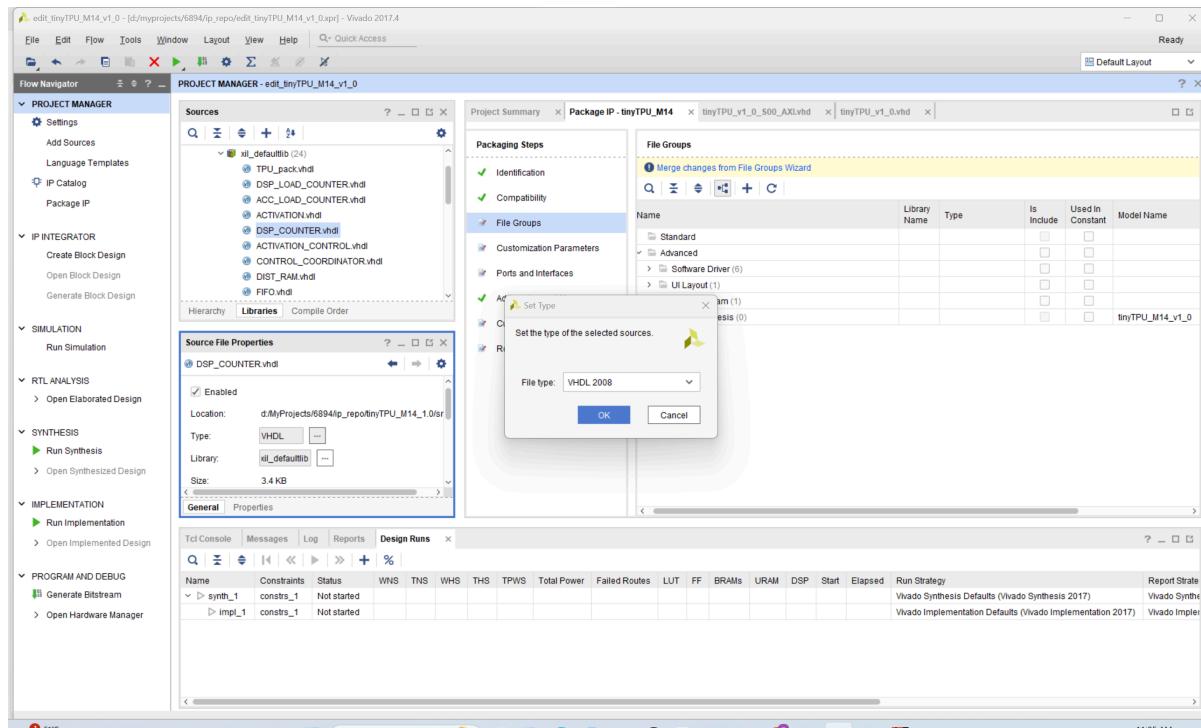


```
56     );
57     port (
58         SYNCHRONIZE      : out std_logic;
59         S_AXI_ACLK      : in std_logic;
60         S_AXI_ARESETN   : in std_logic;
61         S_AXI_AWADDR   : in std_logic_vector(C_S_AXI_ADDR_WIDTH-1 downto 0);
62         S_AXI_ARPROT   : in std_logic_vector(2 downto 0);
63         S_AXI_ARVALID  : in std_logic;
64         S_AXI_ARREADY  : out std_logic;
65         S_AXI_WDATA    : in std_logic_vector(C_S_AXI_DATA_WIDTH-1 downto 0);
66         S_AXI_WSTRB   : in std_logic_vector((C_S_AXI_DATA_WIDTH/8)-1 downto 0);
67         S_AXI_WVALID  : in std_logic;
68         S_AXI_WREADY  : out std_logic;
69         S_AXI_BRESP    : out std_logic_vector(1 downto 0);
70         S_AXI_BVALID  : out std_logic;
71         S_AXI_BREADY  : in std_logic;
72         S_AXI_ARADDR  : in std_logic_vector(C_S_AXI_ADDR_WIDTH-1 downto 0);
73         S_AXI_ARPROT  : in std_logic_vector(2 downto 0);
74         S_AXI_ARVALID  : in std_logic;
75         S_AXI_ARREADY  : out std_logic;
76         S_AXI_RDATA    : out std_logic_vector(C_S_AXI_DATA_WIDTH-1 downto 0);
77         S_AXI_RRESP    : out std_logic_vector(1 downto 0);
78         S_AXI_RVALID  : out std_logic;
79         S_AXI_RREADY  : in std_logic
80     );
81 end component tinyTPU_M14_v1_0_S00_AXI;
```

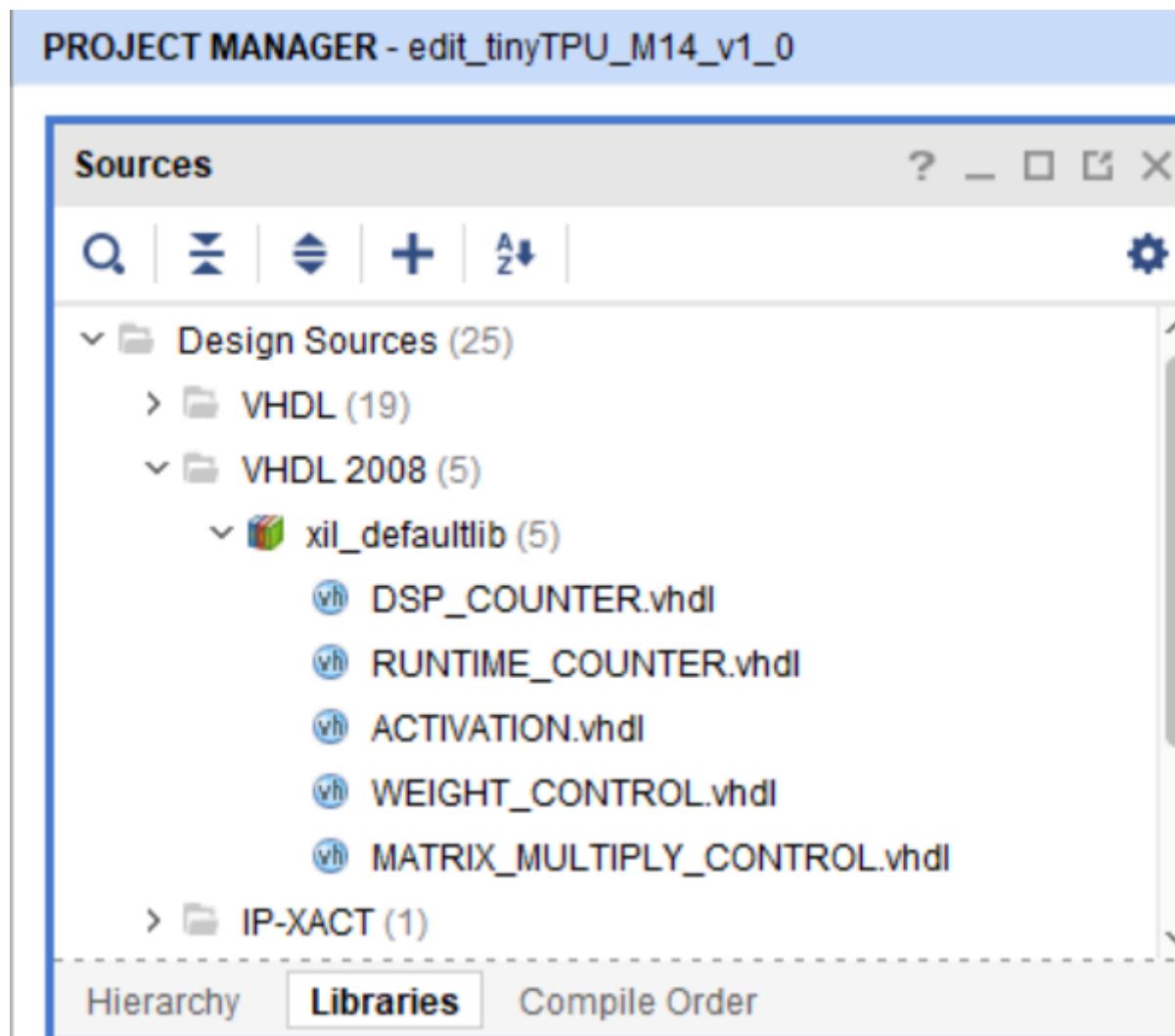
Two AXI files should be changed, then there will be no errors and Design Sources will add _M14 in the name.



You can double check in Libraries and you can see VHDL(24).



Then change five files type name from VHDL to VHDL2008. Includes **DSP_COUNTER**, **RUNTIME_COUNTER**, **ACTIVATION**, **WEIGHT_CONTROL** and **MATRIX_MULTIPLY_CONTROL**.



After change, you can see there will be five VHDL 2008 type files and 19 VHDL files.

Then we can merge File Groups.

There will be 5 warnings, no worries.

Packaging Steps

- ✓ Identification
- ✓ Compatibility
- 🟡 File Groups
- ✗ Customization Parameters
- ✗ Ports and Interfaces
- ✓ Addressing and Memory
- ✗ Customization GUI
- ✗ Review and Package

Customization Parameters

⚠ Merge changes from Customization Parameters Wizard

Name	Description	Display Name	Value	Value Bit String Length	Value Format
Customization Parameters					
C_S00_AXI_DATA_WIDTH	Width of S_AXI data bus	C S00 AXI DATA WIDTH	32	0	long
C_S00_AXI_ADDR_WIDTH	Width of S_AXI address bus	C S00 AXI ADDR WIDTH	4	0	long
C_S00_AXI_BASEADDR		C S00 AXI BASEADDR	0xFFFFFFFF	32	bitString
C_S00_AXI_HIGHADDR		C S00 AXI HIGHADDR	0x00000000	32	bitString

Merge Parameters and the rest will be merged automatically.

Project Summary x **Package IP - tinyTPU_M14** x **tinyTPU_v1_0_S00_AXI.vhd** x **tinyTPU_v1_0.vhd** x

Packaging Steps

- ✓ Identification
- ✓ Compatibility
- 🟡 File Groups
- ✓ Customization Parameters
- ✓ Ports and Interfaces
- ✓ Addressing and Memory
- ✓ Customization GUI

Review and Package

⚠ 5 warnings 2 info messages

Summary

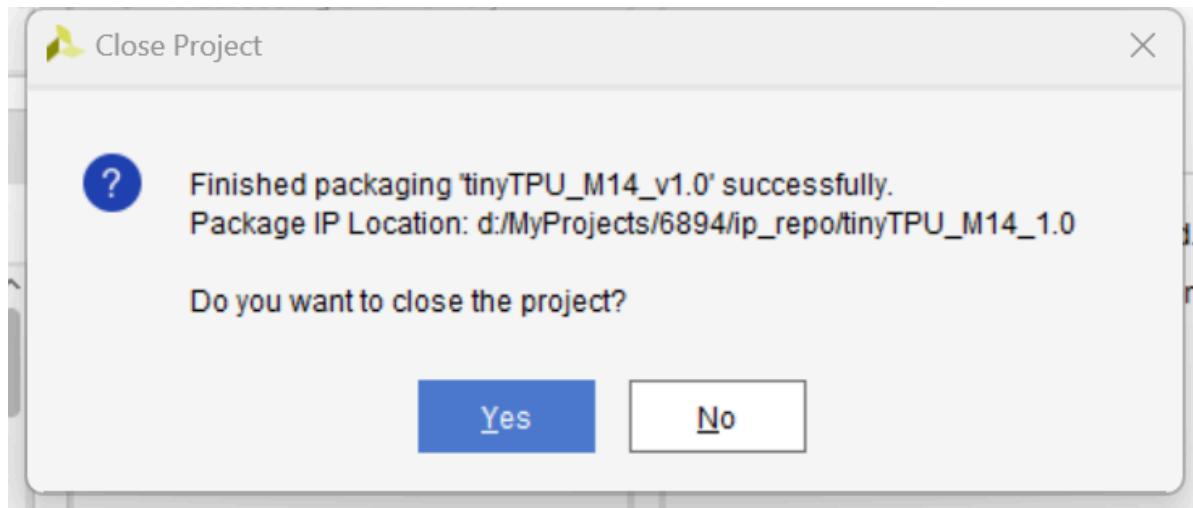
Display name: tinyTPU_M14_v1.0
 Description: My new AXI IP
 Root directory: d:/MyProjects/6894/ip_repo/tinyTPU_M14_1.0

After Packaging

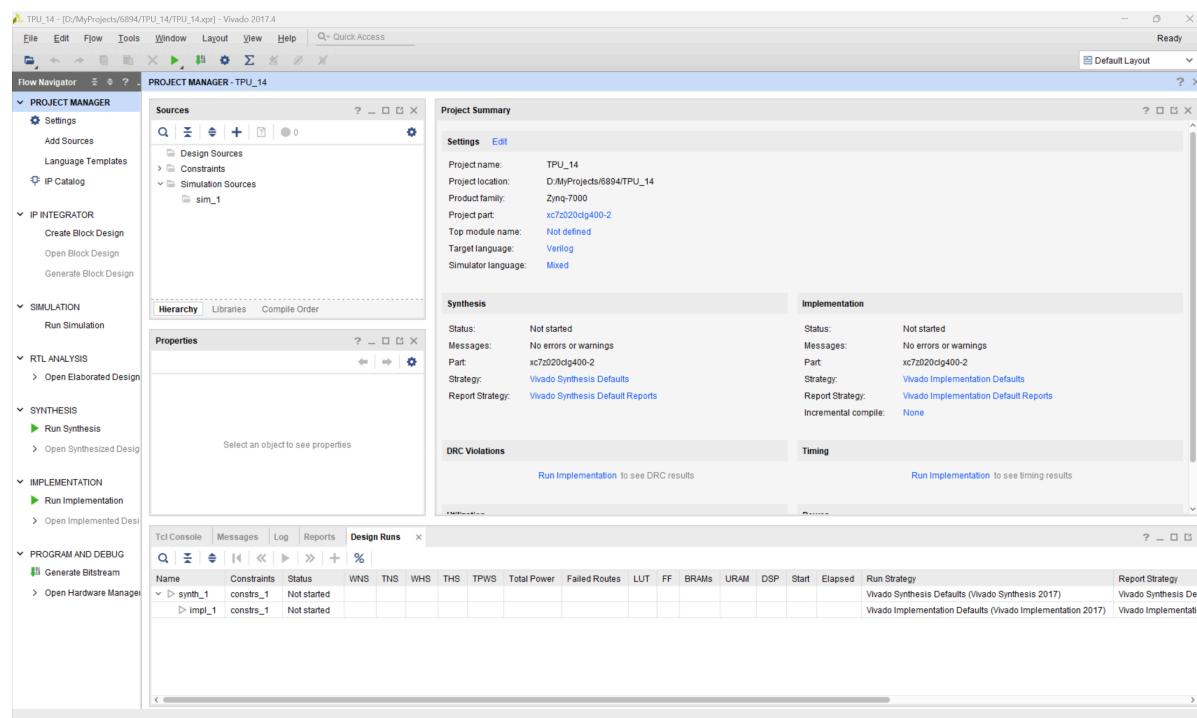
An archive will not be generated. Use the settings link below to change your preference
 Project will be removed after completion
[Edit packaging settings](#)

Re-Package IP

Click Re-Package IP.

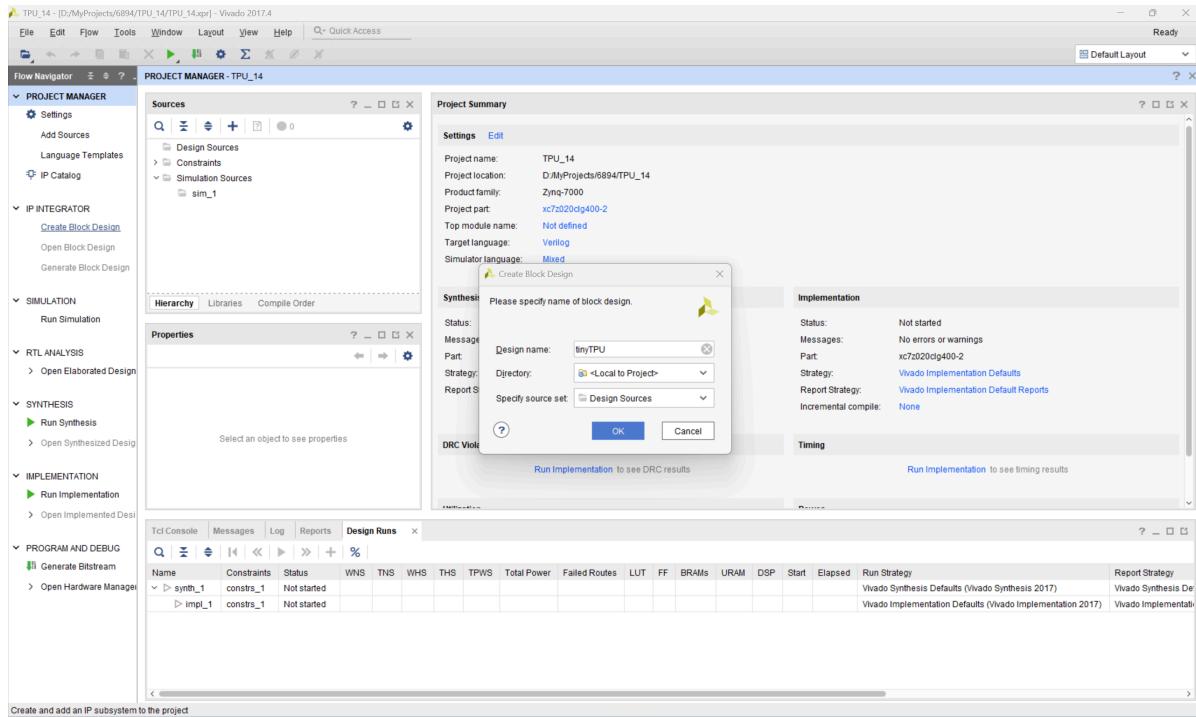


Just click Yes.

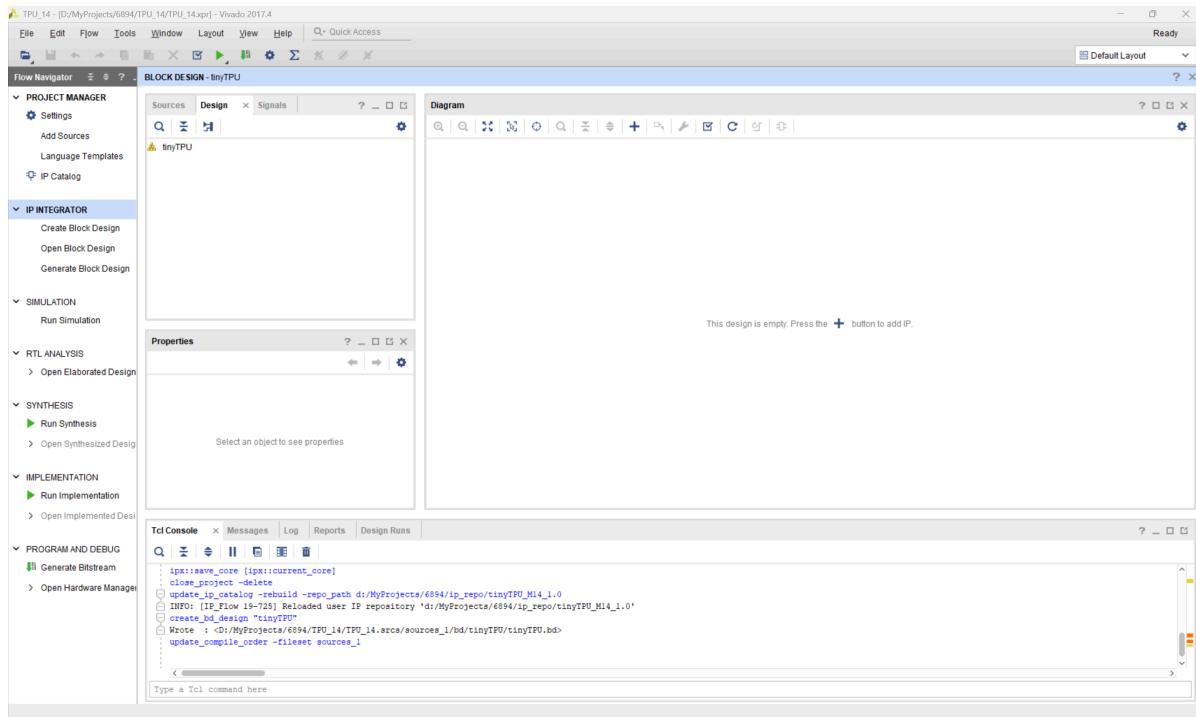


Then you will come back to empty project you created as TPU_14.

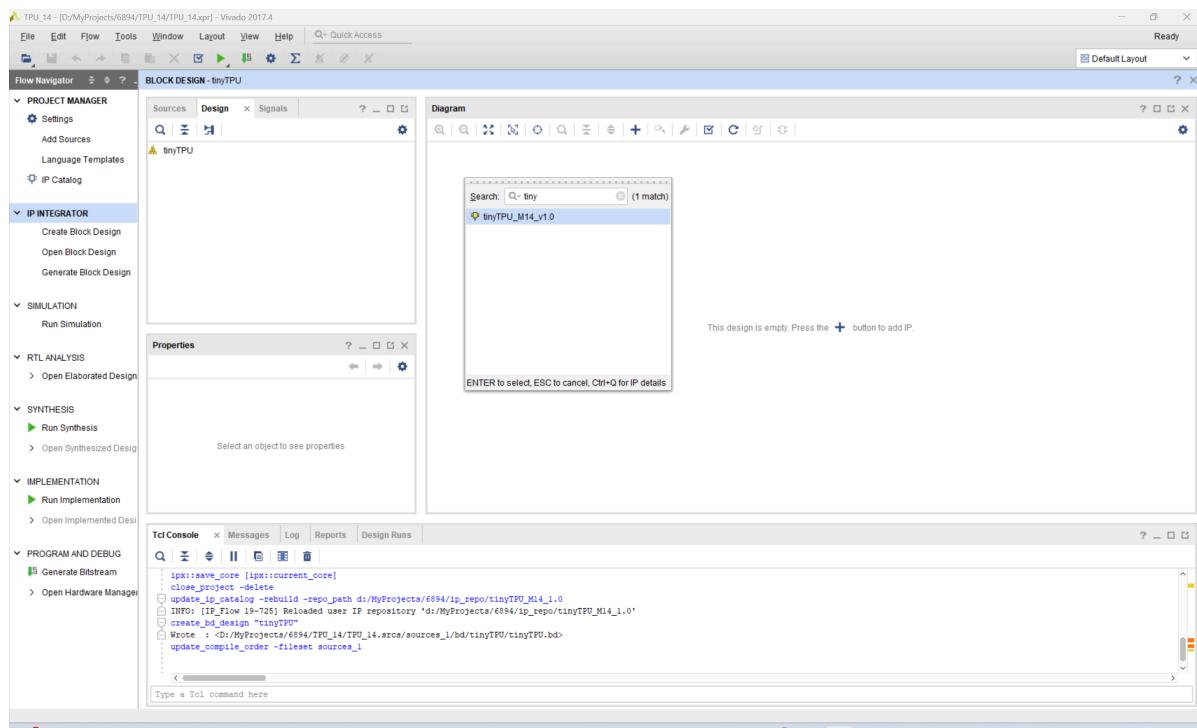
Create Block Design



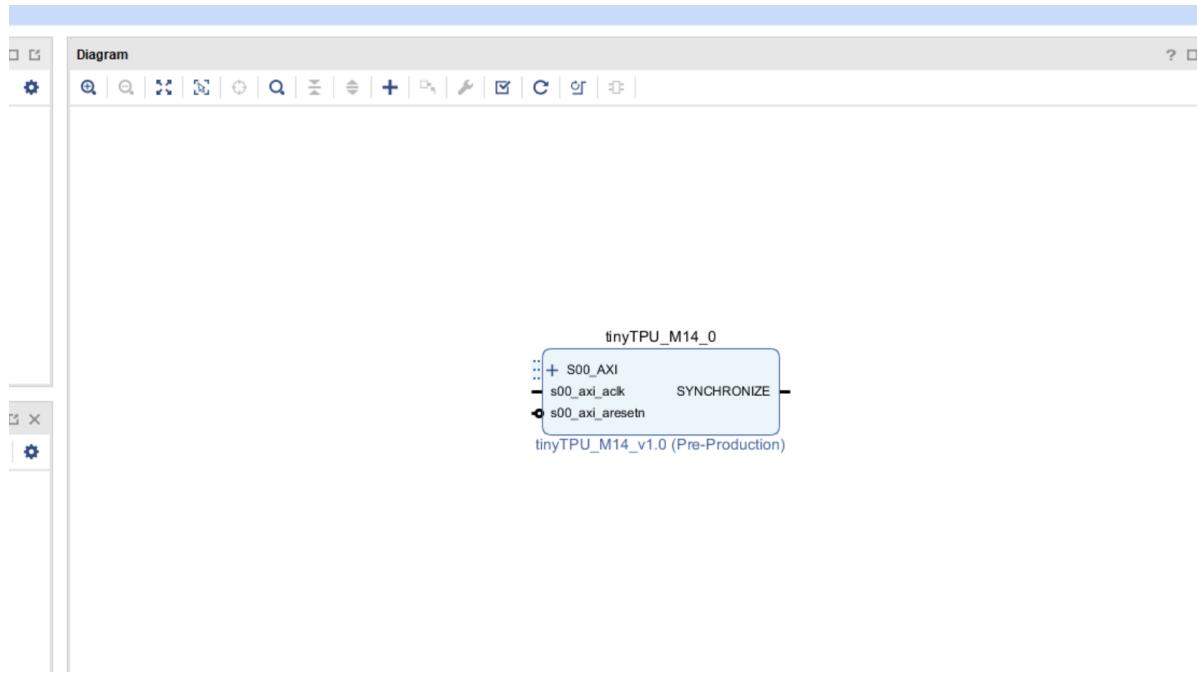
Press left list Create Block Design, then there will be a window to rename your Design name as tinyTPU.



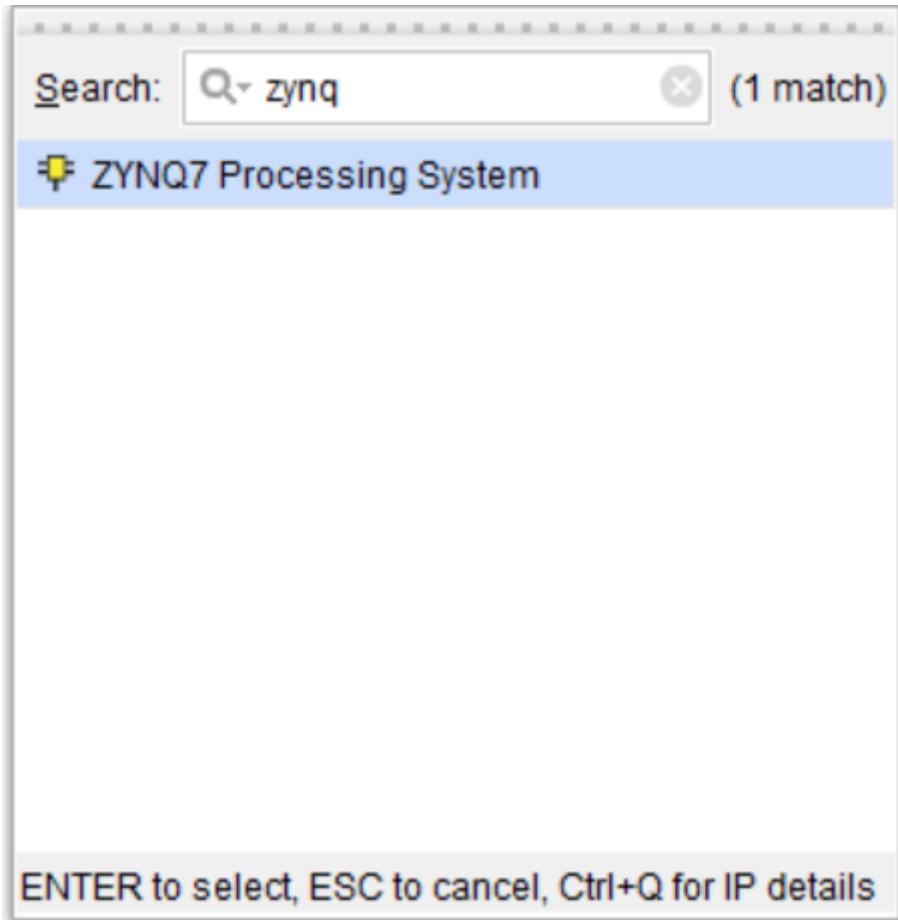
There will be an empty block.



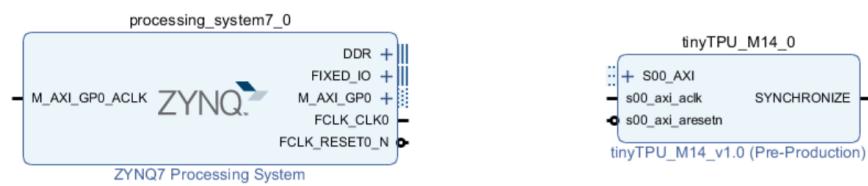
Click "+" to create IP block of tinyTPU. Search tinyTPU_M14_v1.0 and click the IP.



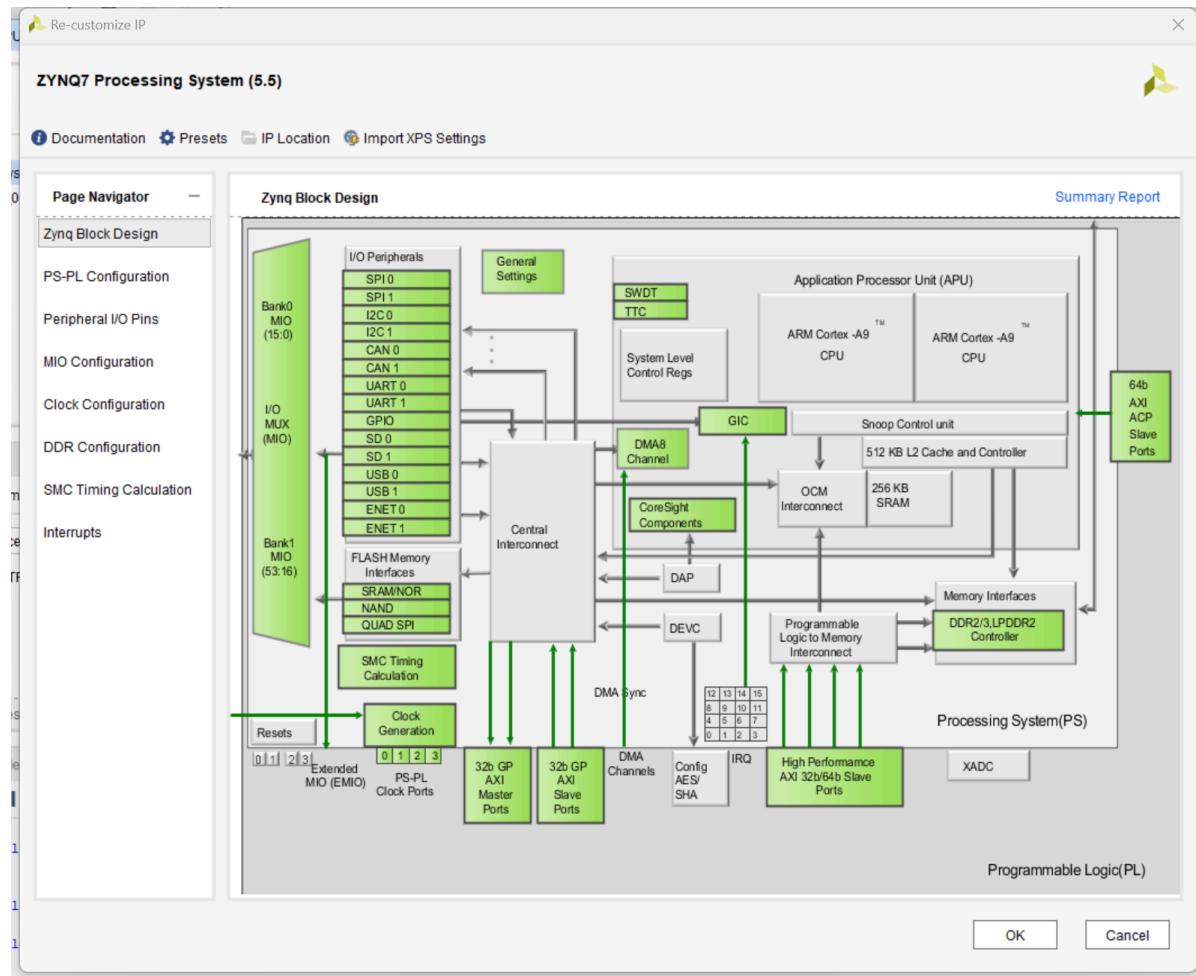
Here it will be placed on the design.



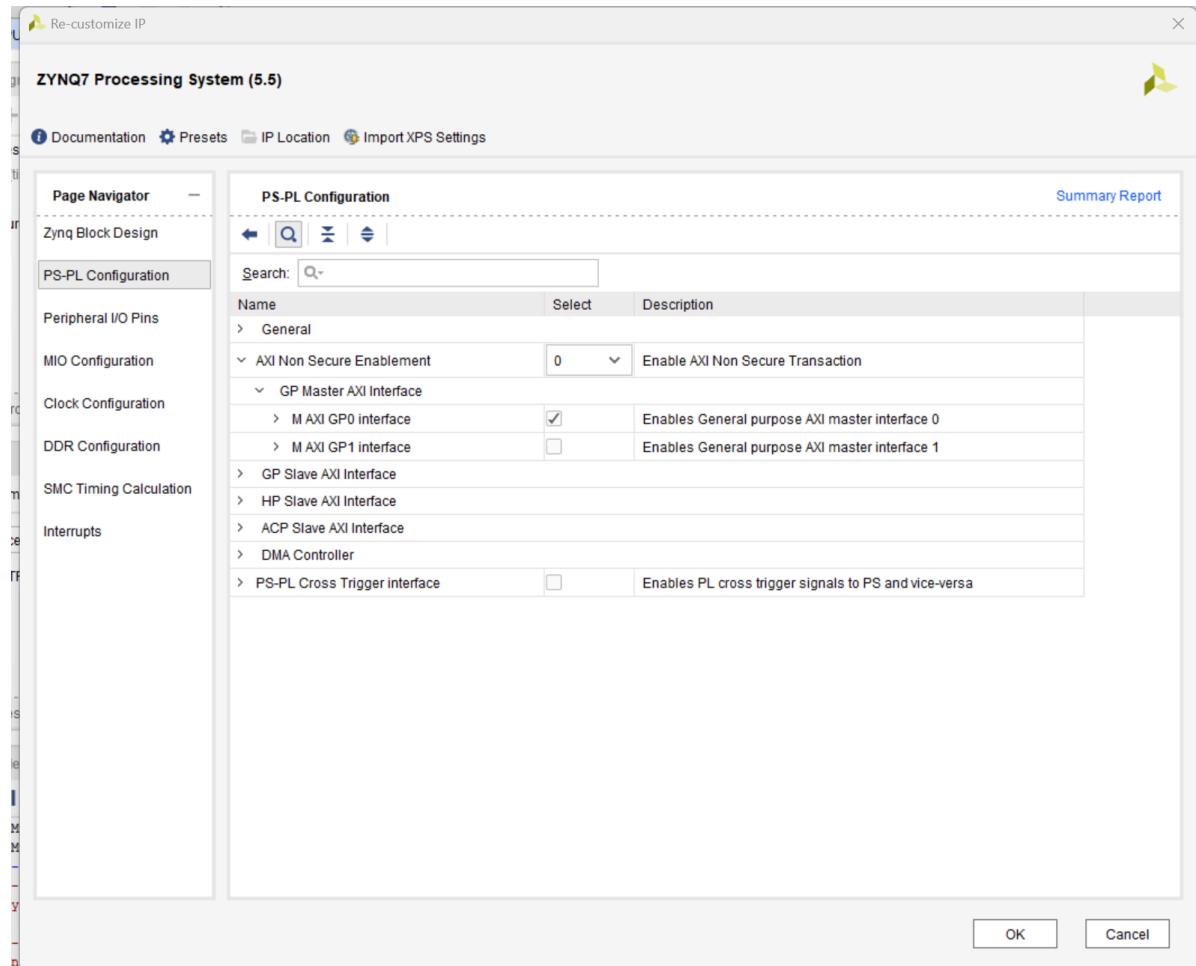
Similarly, search ZYNQ to find the process core.



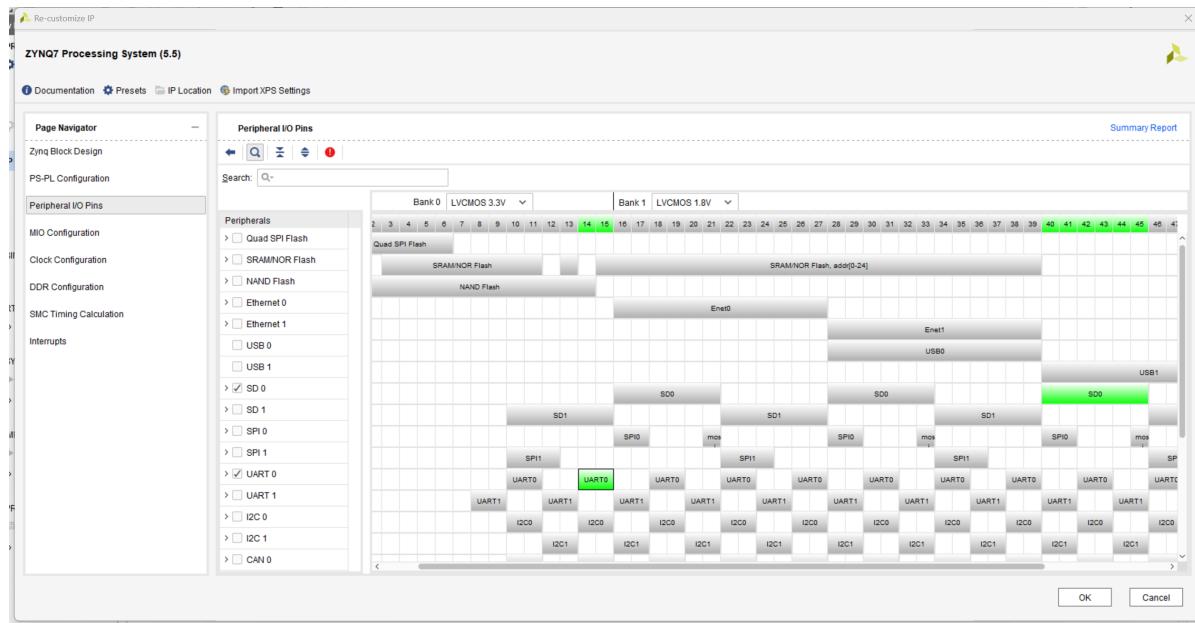
Place it here and double click the ZYNQ.



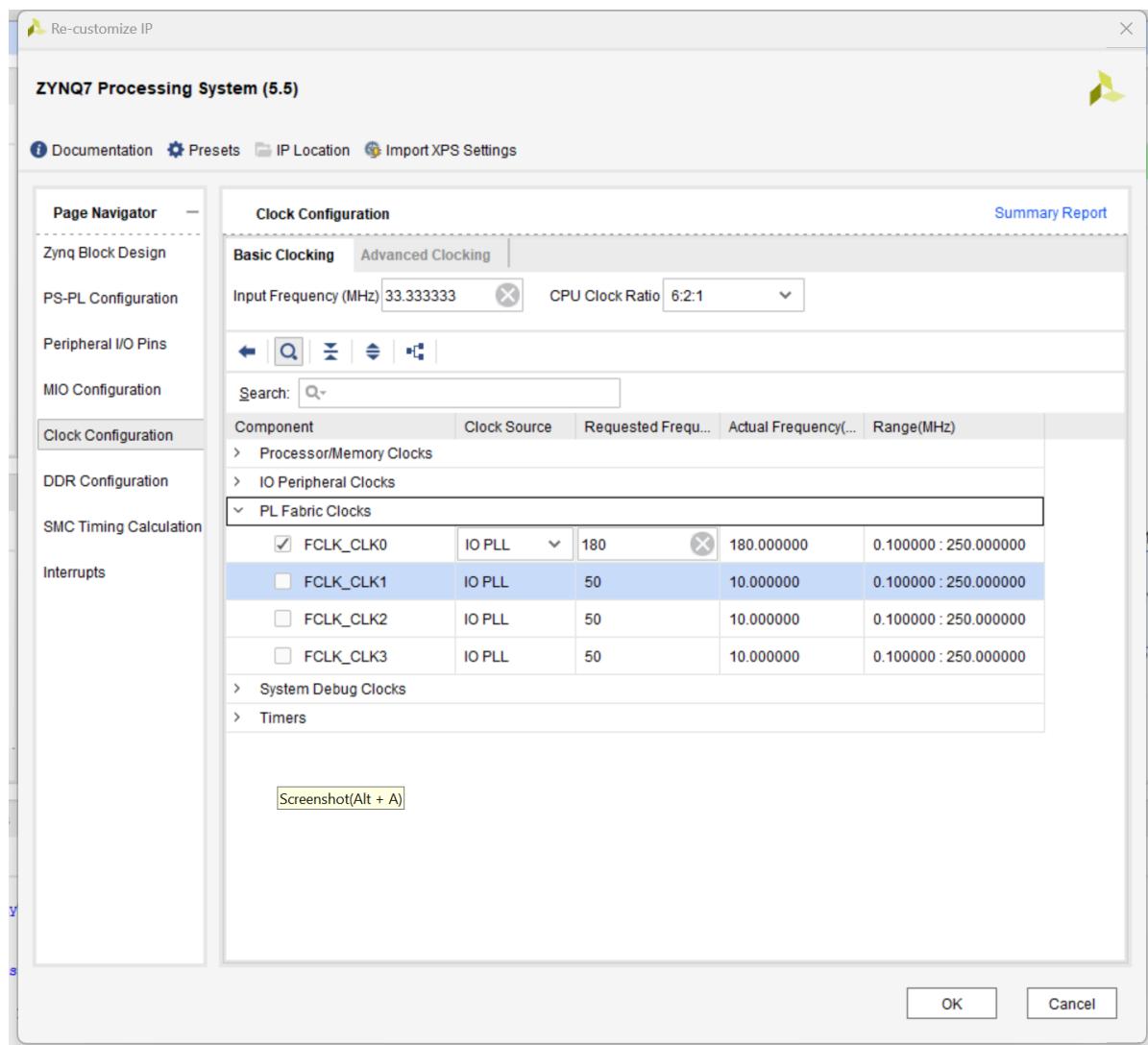
This window will be opened.



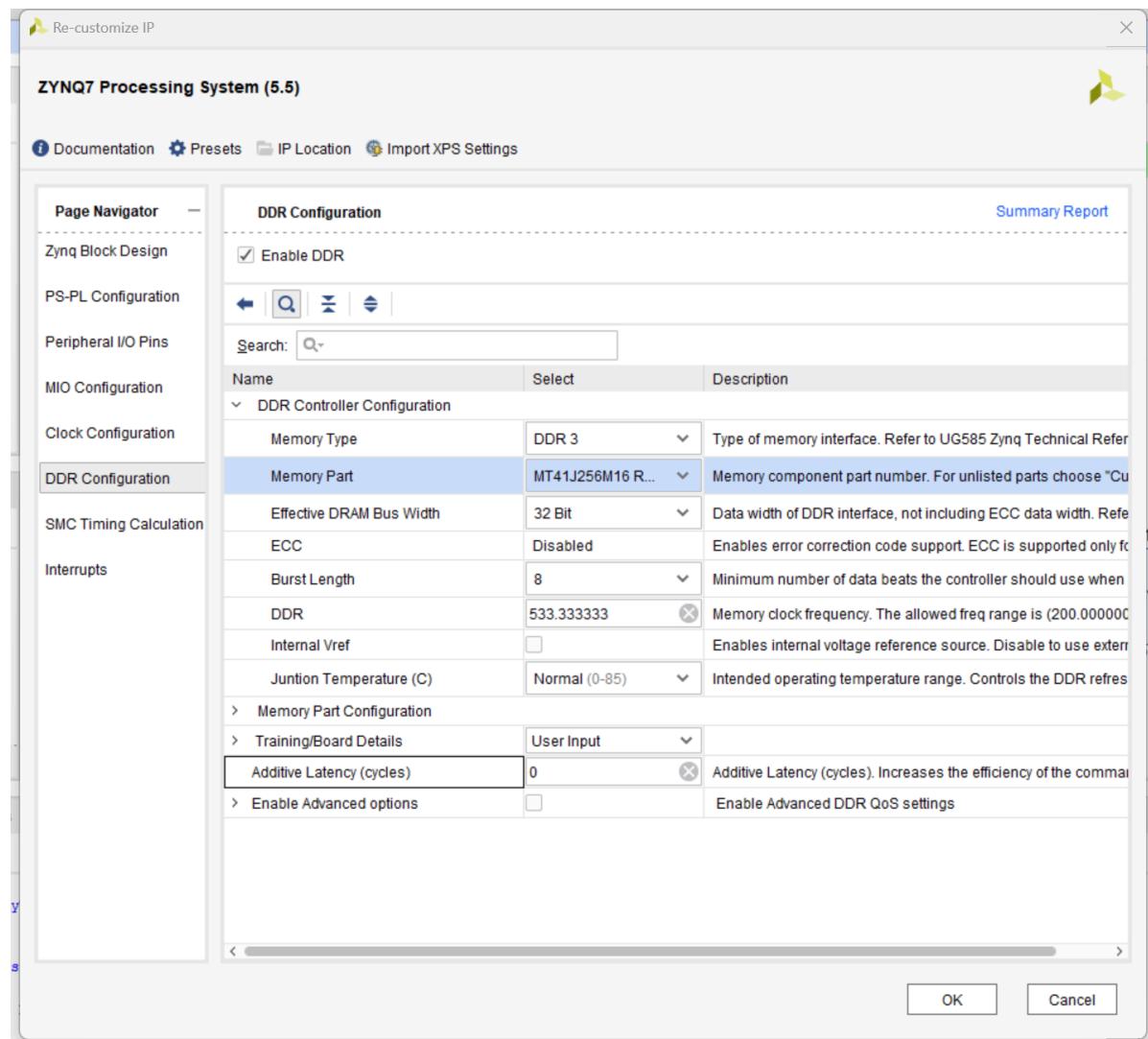
Enable M AXI GP0 interface.



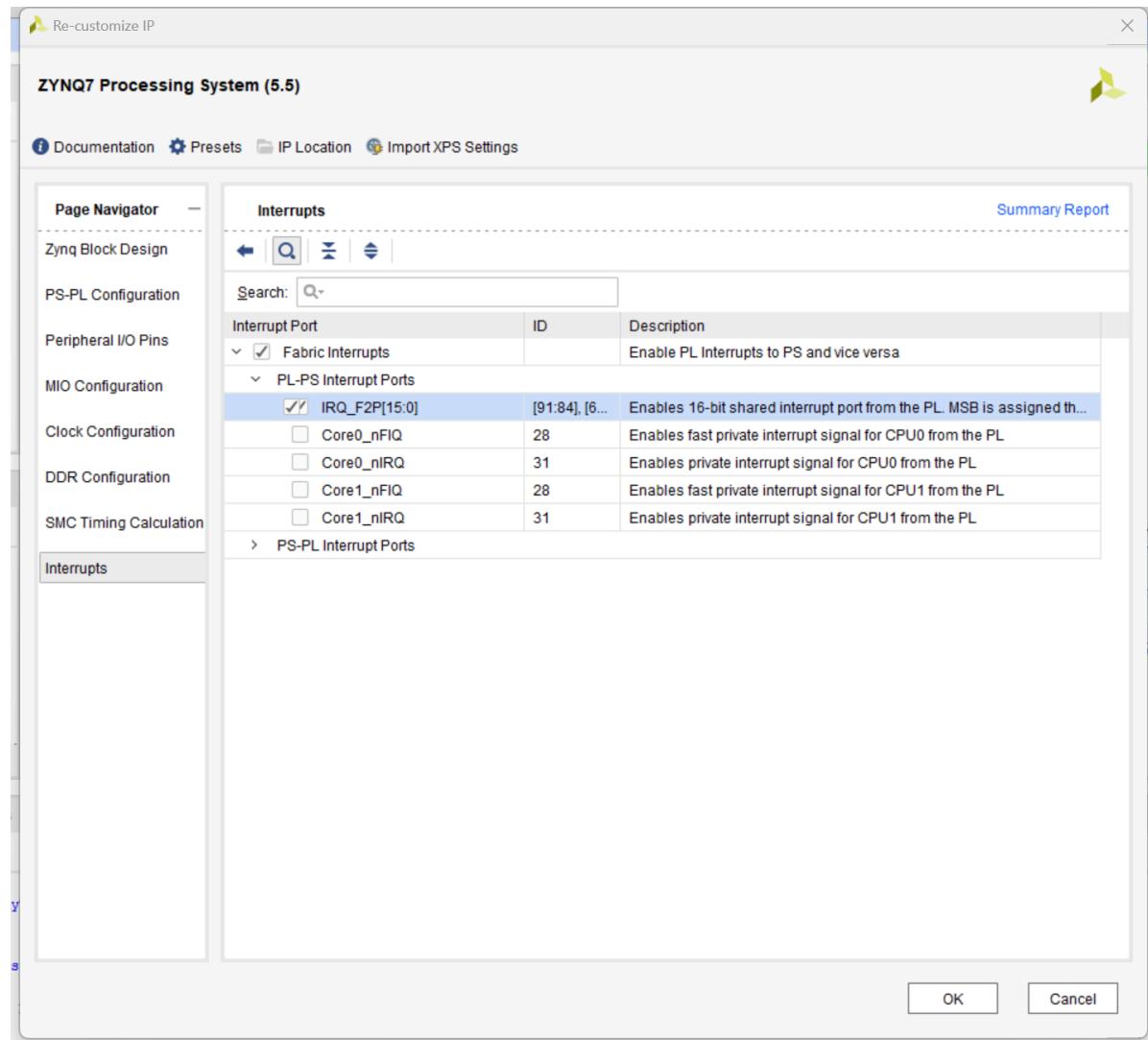
Enable SD0 with pin40-pin45 and UART0 with pin14 pin15. Select Bank1 LVC MOS1.8V.



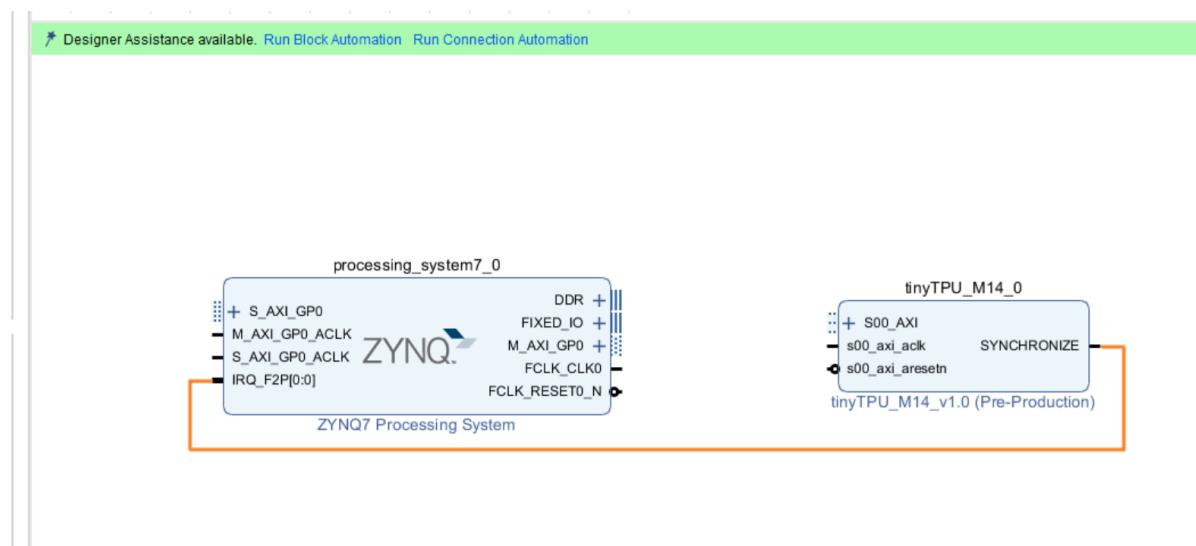
Choose FCLK_CLK0 IO PLL with 180MHz.



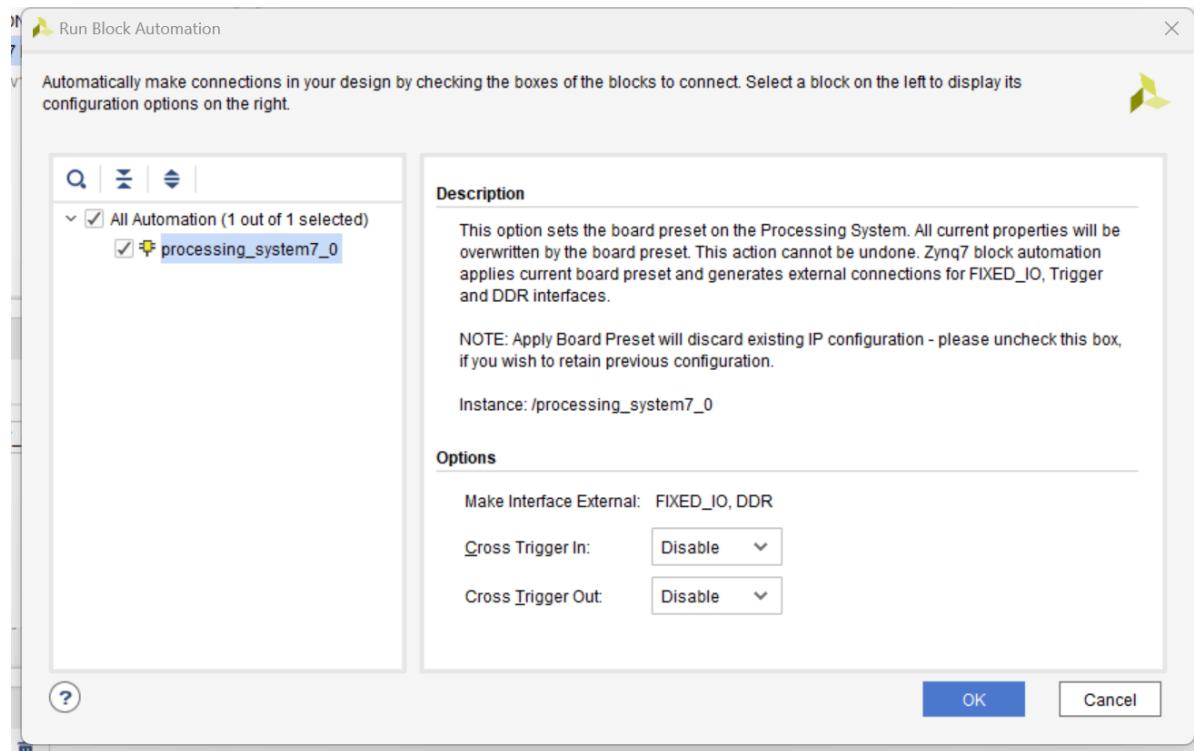
Select Memory Part as MT41J256M16 RE-125.



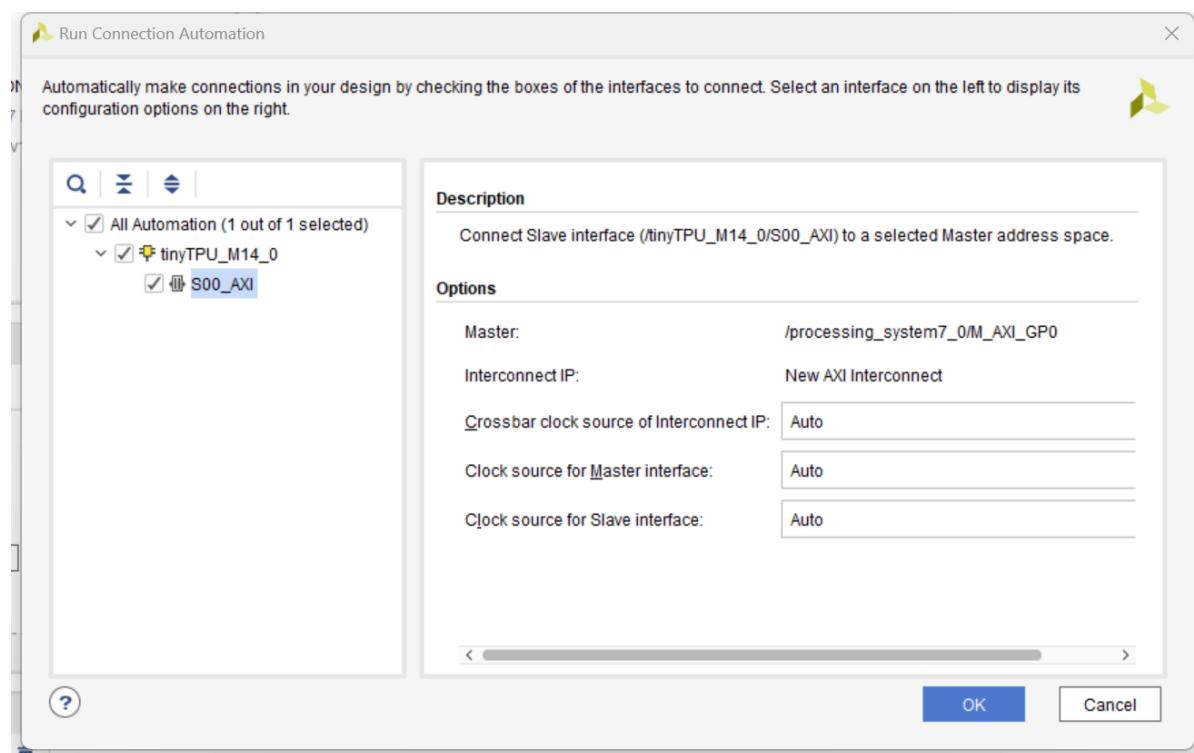
Enable interrupts.



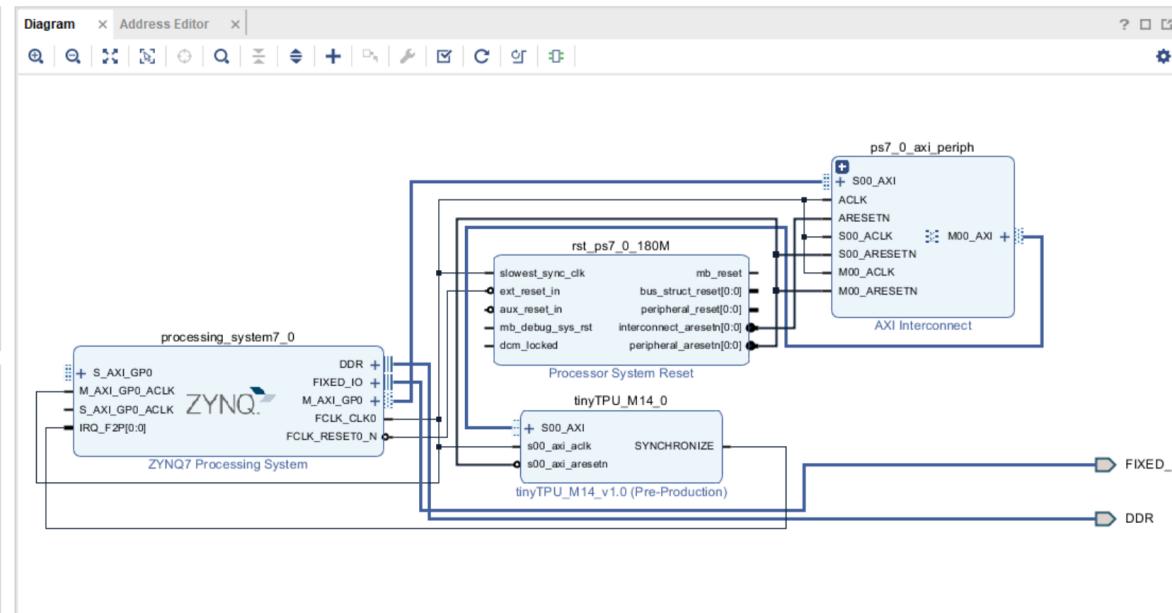
Connect IRQ with SYNCHRONIZE.



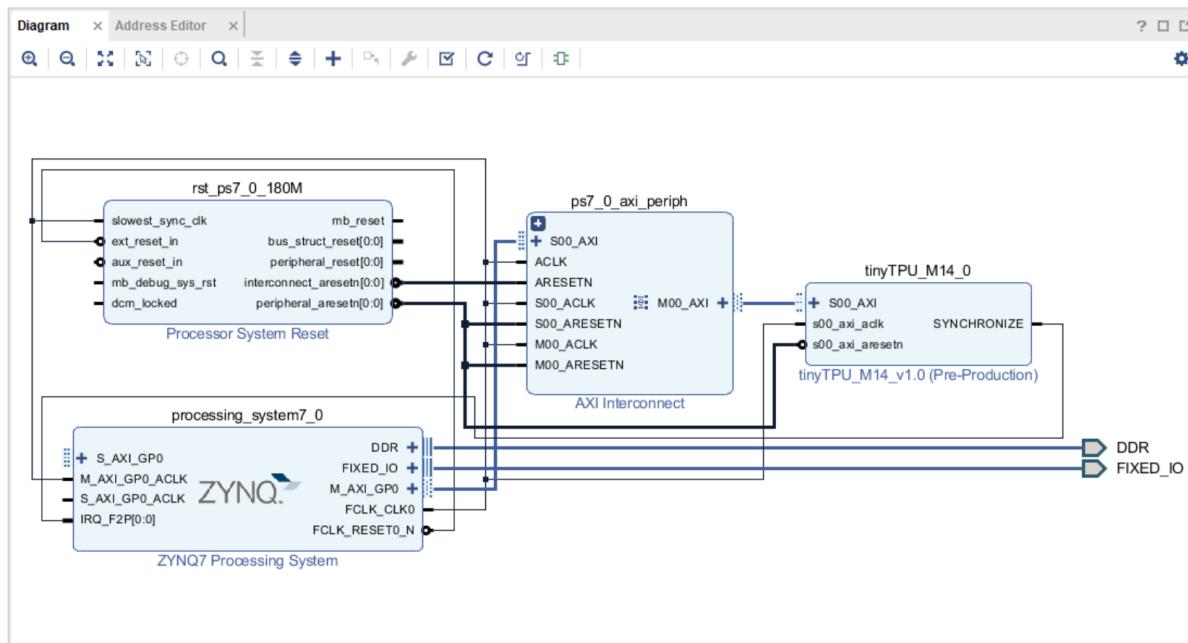
Click Run Block Automation and click OK.



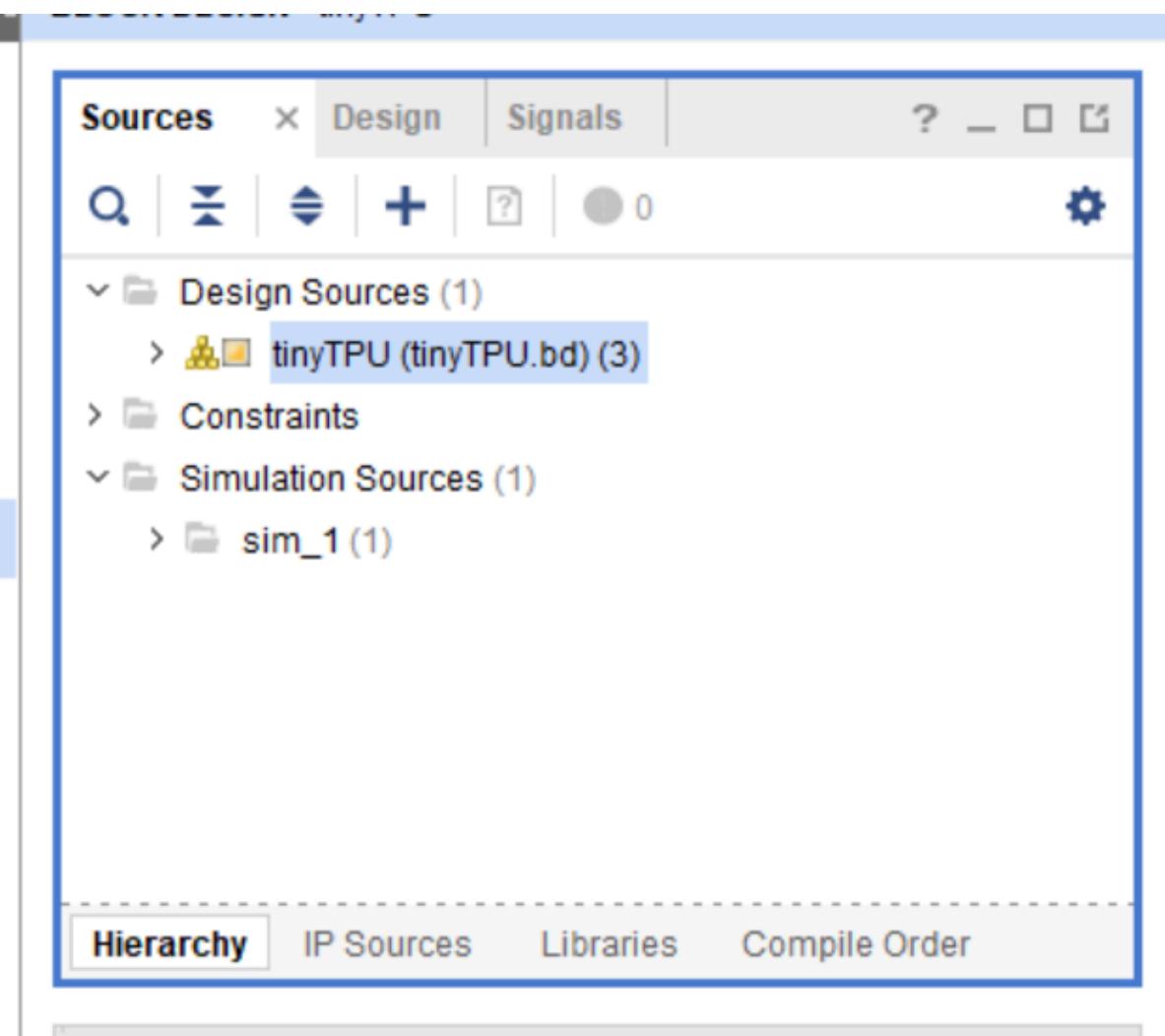
Click Run Connection Automation and click OK.



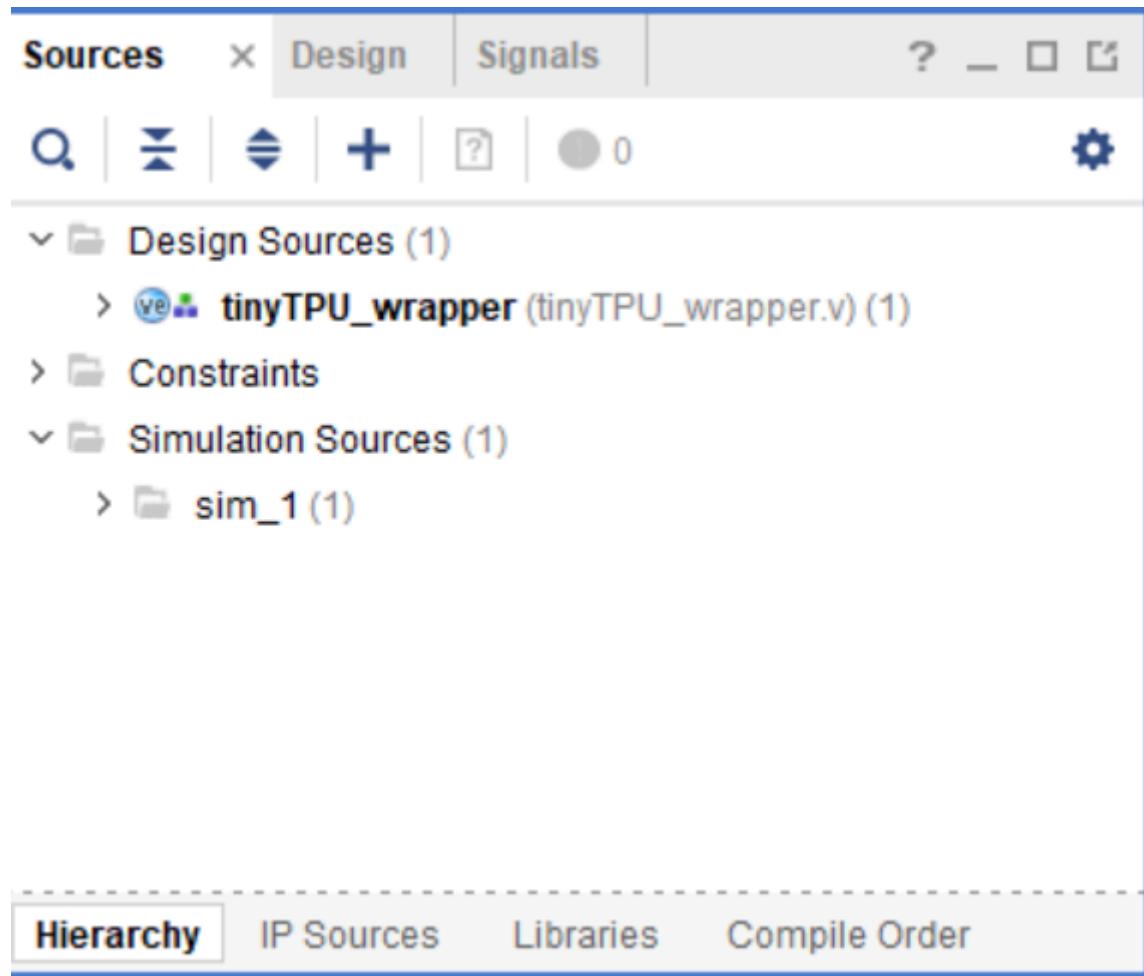
Here will be the Block Design.



You can click the refresh circle to reorder the blocks.



Right click this yellow square and choose *Create HDL Wrapper* to generate programmable top level file.

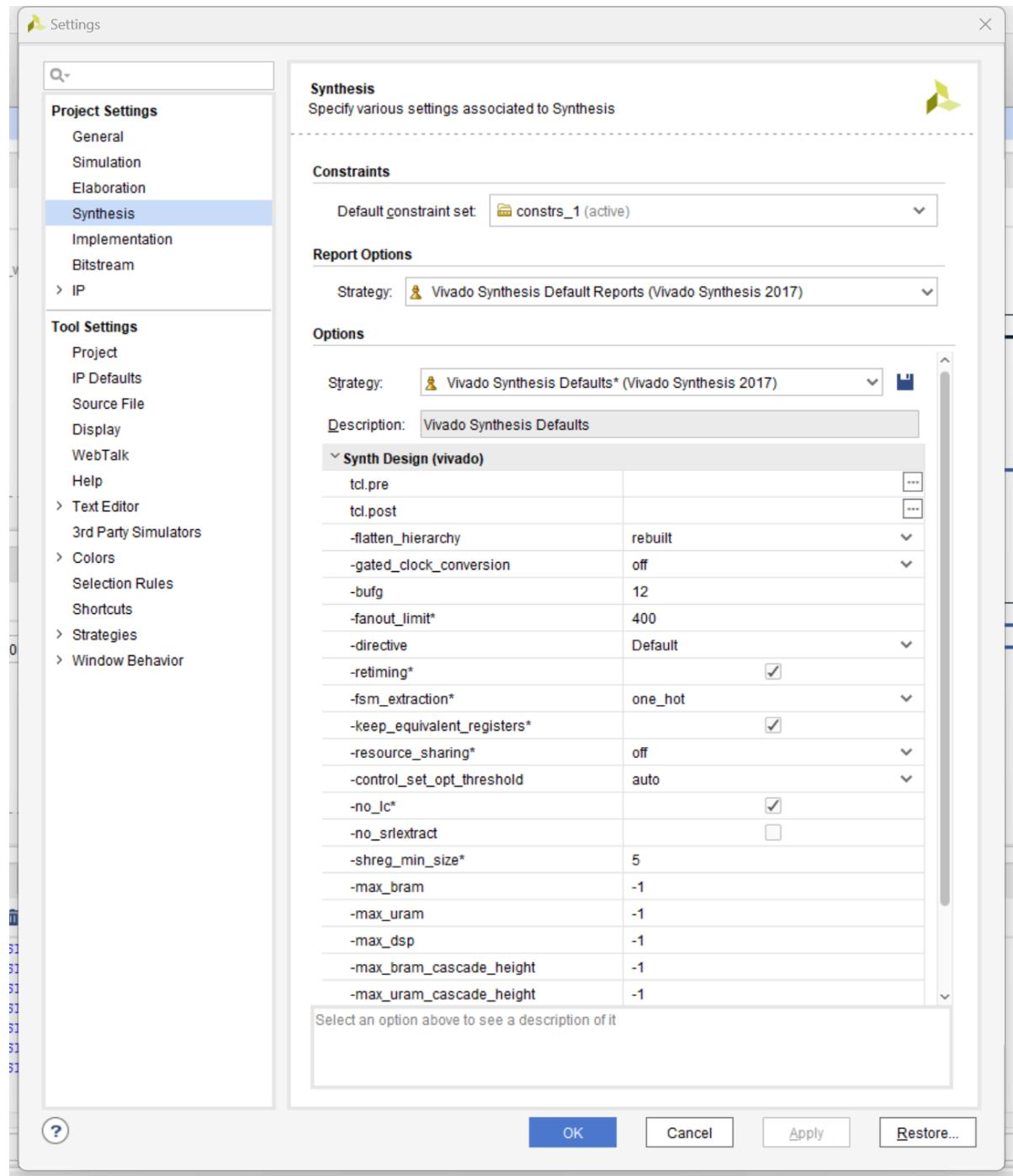


It will be like this.

Synthesis

Click Tools -> Setting and click Synthesis. Change the settings shown below.

You can run synthesis now.



Implement

Similarly click Implementation and change the settings like below.

Settings

Project Settings

- General
- Simulation
- Elaboration
- Synthesis
- Implementation**
- Bitstream
- IP

Tool Settings

- Project
- IP Defaults
- Source File
- Display
- WebTalk
- Help
- Text Editor
- 3rd Party Simulators
- Colors
- Selection Rules
- Shortcuts
- Strategies
- Window Behavior

Implementation
Specify various settings associated to Implementation

Constraints

Default constraint set: **constrs_1 (active)**

Report Options

Strategy: **Vivado Implementation Default Reports (Vivado Implementation 2017)**

Options

Incremental compile:

Strategy: **Vivado Implementation Defaults* (Vivado Implementa...)**

Description: Default settings for Implementation.

Opt Design (opt_design)

is_enabled	<input checked="" type="checkbox"/>
tcl.pre	<input type="button" value="..."/>
tcl.post	<input type="button" value="..."/>
-verbose	<input type="checkbox"/>
-directive	ExploreWithRemap
More Options	

Power Opt Design (power_opt_design)

is_enabled	<input type="checkbox"/>
tcl.pre	<input type="button" value="..."/>
tcl.post	<input type="button" value="..."/>
More Options	

Place Design (place_design)

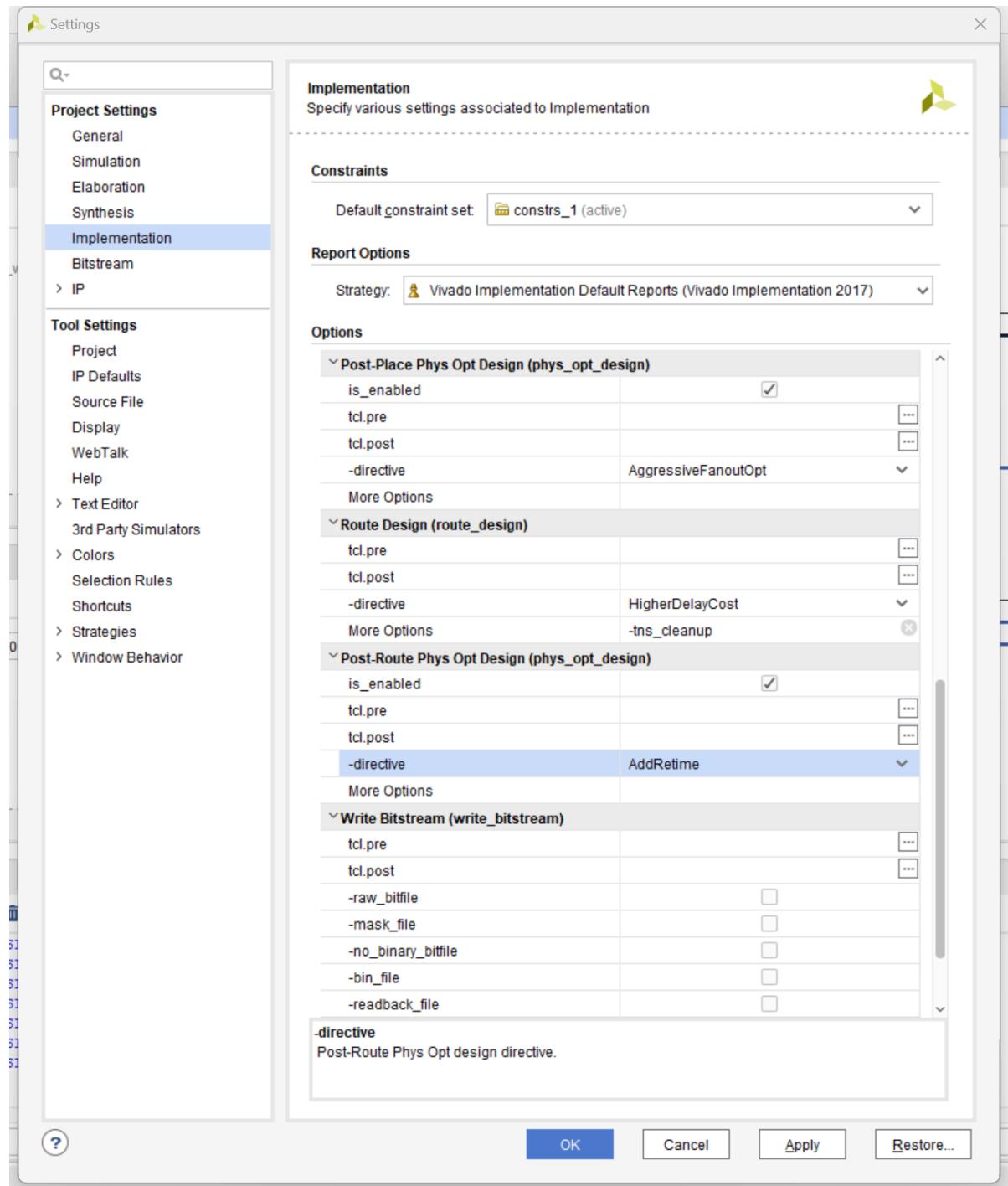
tcl.pre	<input type="button" value="..."/>
tcl.post	<input type="button" value="..."/>
-directive	ExtraNetDelay_high
More Options	

Post-Place Power Opt Design (power_opt_design)

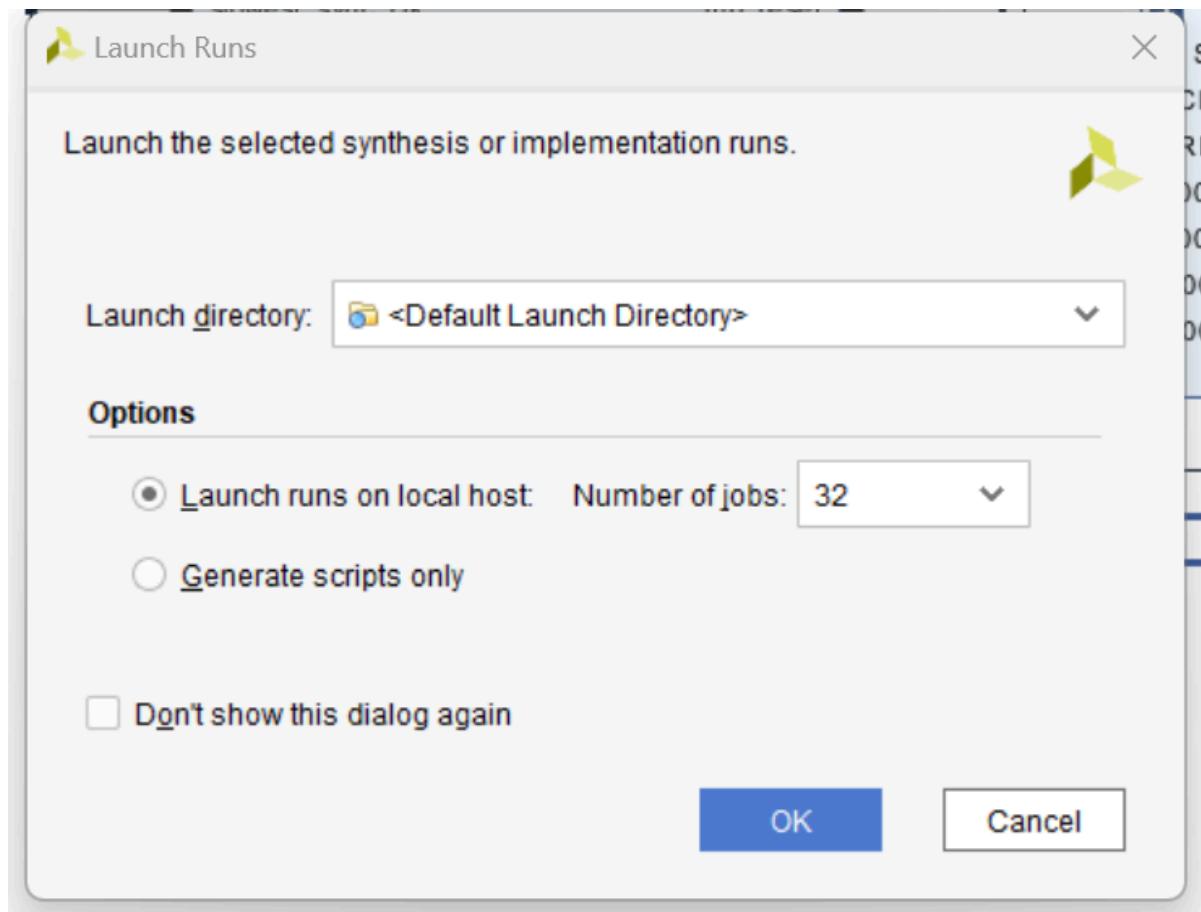
is_enabled	<input type="checkbox"/>
-directive	Place design directive.

Buttons

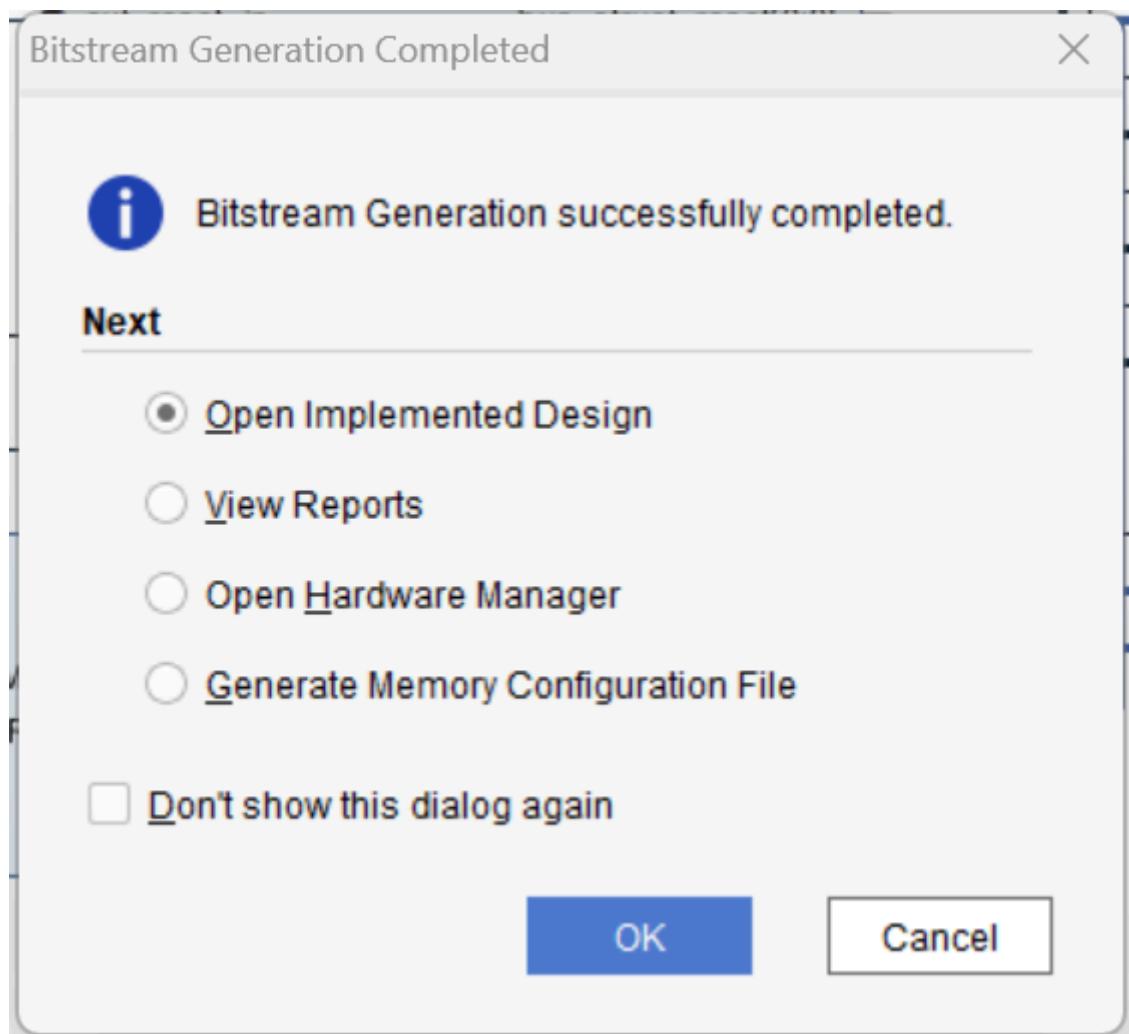
OK Cancel Apply Restore...



Then you can Run Implementation. After implementation finishes, you can run Generate BitStream.



You can choose Number of Jobs as much as possible.



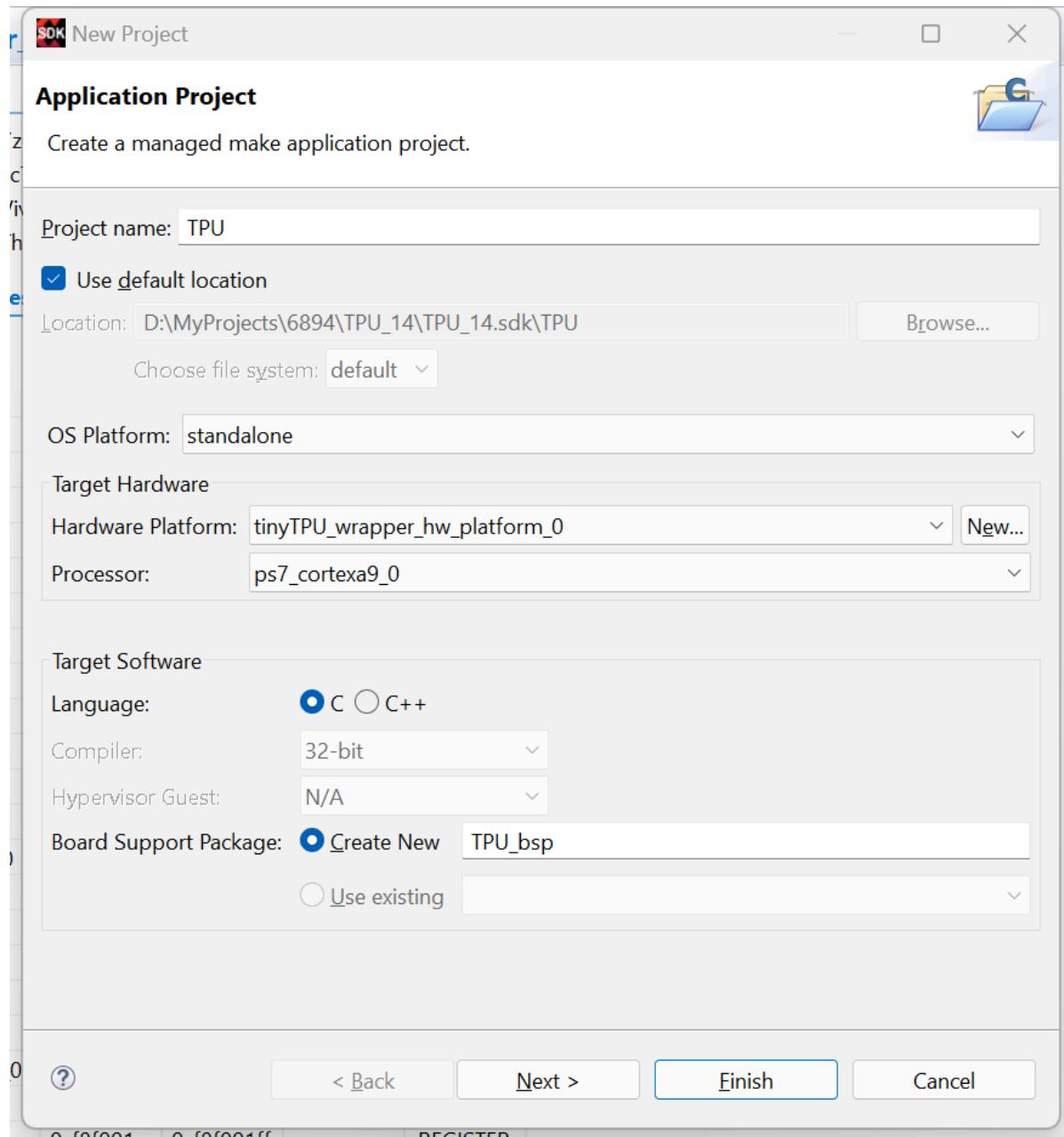
Here you can see you finished Bitstream Generation.

Congratulations, you have finished hardware design part.

Software Design

Add Sample Project in Xilinx SDK

Go to File -> Export Hardware and click Include Bitstream: Press OK. Then press File -> Launch SDK and the Xilinx SDK will be started (After v2019.3 there will be no launch SDK in Vivado). Create an application project and click next.



Templates



Create one of the available templates to generate a fully-functioning application project.

Available Templates:

- Dhrystone
- Empty Application
- Hello World**
- IwIP Echo Server
- Memory Tests
- OpenAMP echo-test
- OpenAMP matrix multiplication Demo
- OpenAMP RPC Demo
- Peripheral Tests
- RSA Authentication App
- Zynq DRAM tests
- Zynq FSBL

Let's say 'Hello World' in C.



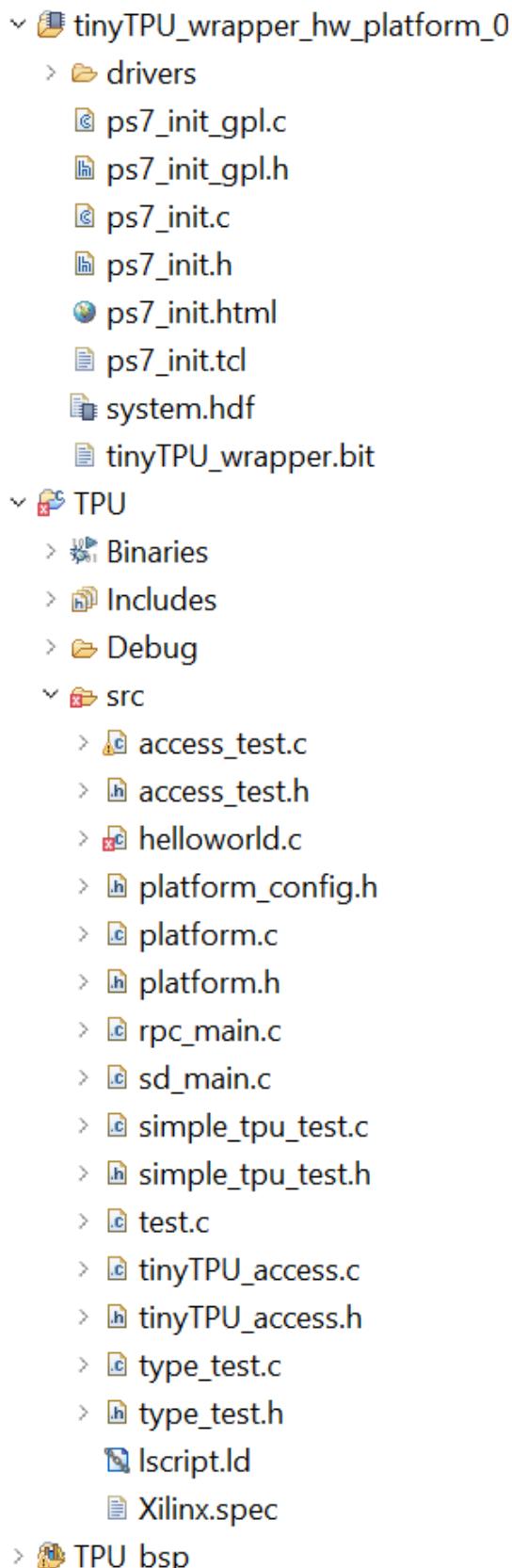
[< Back](#)

[Next >](#)

[Finish](#)

[Cancel](#)

Choose Hello World and click finish. BSP will be auto generated.



Add all .c and .h files in EB_C/src and overwrite exist files.

```

// Copyright 2018 Jonas Fuhrmann. All rights reserved.

#include <stdio.h>
#include "platform.h"
#include "xil_printf.h"
#include "xil_io.h"
#include "tinyTPU_M14.h"
#include "xil_exception.h"
#include "xscugic.h"
#include "xparameters.h"
#include "xparameters_ps.h"

#define TPU_BASE           XPAR_TINYTPU_0_S00_AXI_BASEADDR
#define TPU_WEIGHT_BUFFER_BASE (TPU_BASE)
#define TPU_UNIFIED_BUFFER_BASE (TPU_BASE + 0x80000)
#define TPU_INSTRUCTION_BASE (TPU_BASE + 0x90000)

#define TPU_LOWER_WORD_OFFSET 0x4
#define TPU_MIDDLE_WORD_OFFSET 0x8
#define TPU_UPPER_WORD_OFFSET 0xC

#define TPU_DEVICE_ID      XPAR_TINYTPU_0_DEVICE_ID
#define INTC_TPU_SYNCHRONIZE_ID XPS_FPGA2_INT_ID

static XScuGic INTCInst;

#ifndef STANDARD
int setup_interrupt(void);
void synchronize_isr(void *vp);

volatile char synchronizeHappened;

int main() {
    init_platform();

    synchronizeHappened = FALSE;
    if(setup_interrupt() != XST_SUCCESS) {
        printf("Error initializing interrupts.");
    }

    // Nearly identity matrix
    ...
}

```

Replace "tinyTPU.h" as "tinyTPU_M14.h".

Right click BSP and select Board Support Package Settings and check the xilffs library.

Board Support Package Settings

Board Support Package Settings

Control various settings of your Board Support Package.

Overview

- standalone
- xilffs
- drivers
- ps7_cortexa9_0

TPU_bsp

OS Type: **standalone** OS Version: **6.5**

Standalone is a simple, low-level software layer. It provides access to basic processor features such as caches, interrupts and exceptions as well as the basic features of a hosted environment, such as standard input and output, profiling, abort and exit.

Target Hardware

Hardware Specification: D:\MyProjects\6894\TPU_14\TPU_14.sdk\tinyTPU_wrapper_hw_platform_1

Processor: ps7_cortexa9_0

Supported Libraries

Check the box next to the libraries you want included in your Board Support Package. You can configure the library in the navigator on the left.

	Name	Version	Description
<input type="checkbox"/>	libmetal	1.3	Libmetal Library
<input type="checkbox"/>	lwip141	2.0	lwIP TCP/IP Stack library: lwIP v1.4.1
<input type="checkbox"/>	openamp	1.4	OpenAmp Library
<input checked="" type="checkbox"/>	xilffs	3.7	Generic Fat File System Library
<input type="checkbox"/>	xilflash	4.4	Xilinx Flash library for Intel/AMD C...
<input type="checkbox"/>	xilisf	5.9	Xilinx In-system and Serial Flash Li...
<input type="checkbox"/>	xilmfs	2.3	Xilinx Memory File System
<input type="checkbox"/>	xilpm	2.2	Power Management API Library fo...
<input type="checkbox"/>	xilrsa	1.4	Xilinx RSA Library
<input type="checkbox"/>	xilskey	6.3	Xilinx Secure Key Library

OK Cancel

Click standalone, change stdin value as picture and same as stdout.

SDK Board Support Package Settings

Board Support Package Settings

Control various settings of your Board Support Package.

Configuration for OS: standalone

Name	Value	Default	Type	Description
hypervisor_guest	false	false	boolean	Enable hypervisor guest
stdin	ps7_coresight_c...	none	peripheral	stdin peripheral
stdout	ps7_coresight_c...	none	peripheral	stdout peripheral
zynqmp_fsbl_bsp	false	false	boolean	Disable or Enable Optimal Boot Sequence
microblaze_exceptions	false	false	boolean	Enable MicroBlaze Exceptions
enable_sw_intrusive_pr	false	false	boolean	Enable S/W Intrusive Peripheral

OK Cancel

Click xilffs and change use_strfunc Value as 2.

SDK Board Support Package Settings

Board Support Package Settings

Control various settings of your Board Support Package.

Overview

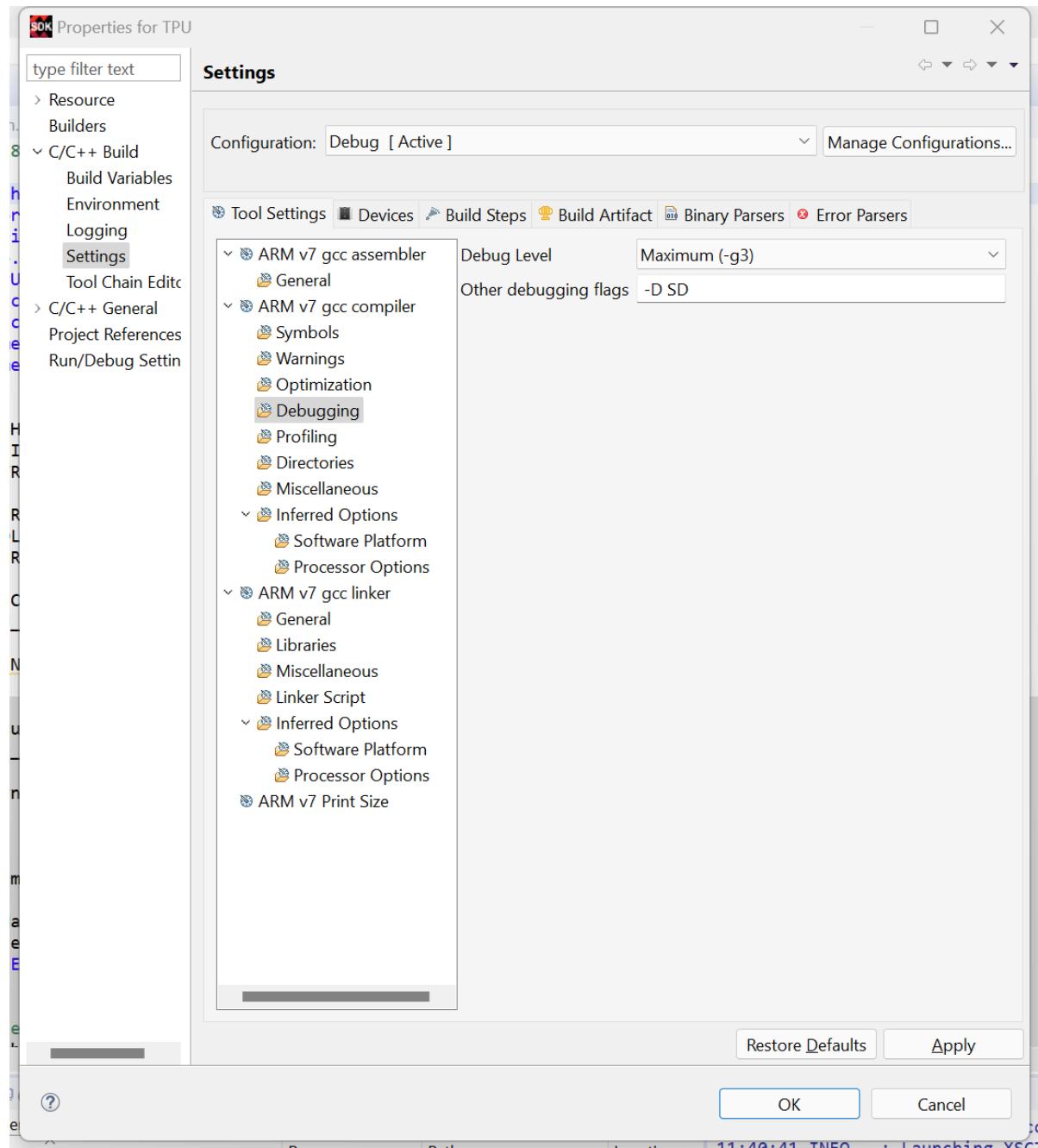
- standalone
 - xilffs
- drivers
 - ps7_cortexa9_0

Configuration for library: **xilffs**

Name	Value	Default	Type	Description
enable_multi_partition	false	false	boolean	0:Single partition, 1:Er
fs_interface	1	1	integer	Enables file system wi
num_logical_vol	2	2	integer	Number of volumes (
read_only	false	false	boolean	Enables the file syste
set_fs_rpath	0	0	integer	Configures relative pa
use_lfn	false	false	boolean	Enables the Long File
use_mkfs	true	true	boolean	Disable(0) or Enable(1
use_strfunc	2	0	integer	Enables the string fun
word_access	true	true	boolean	Enables word access t

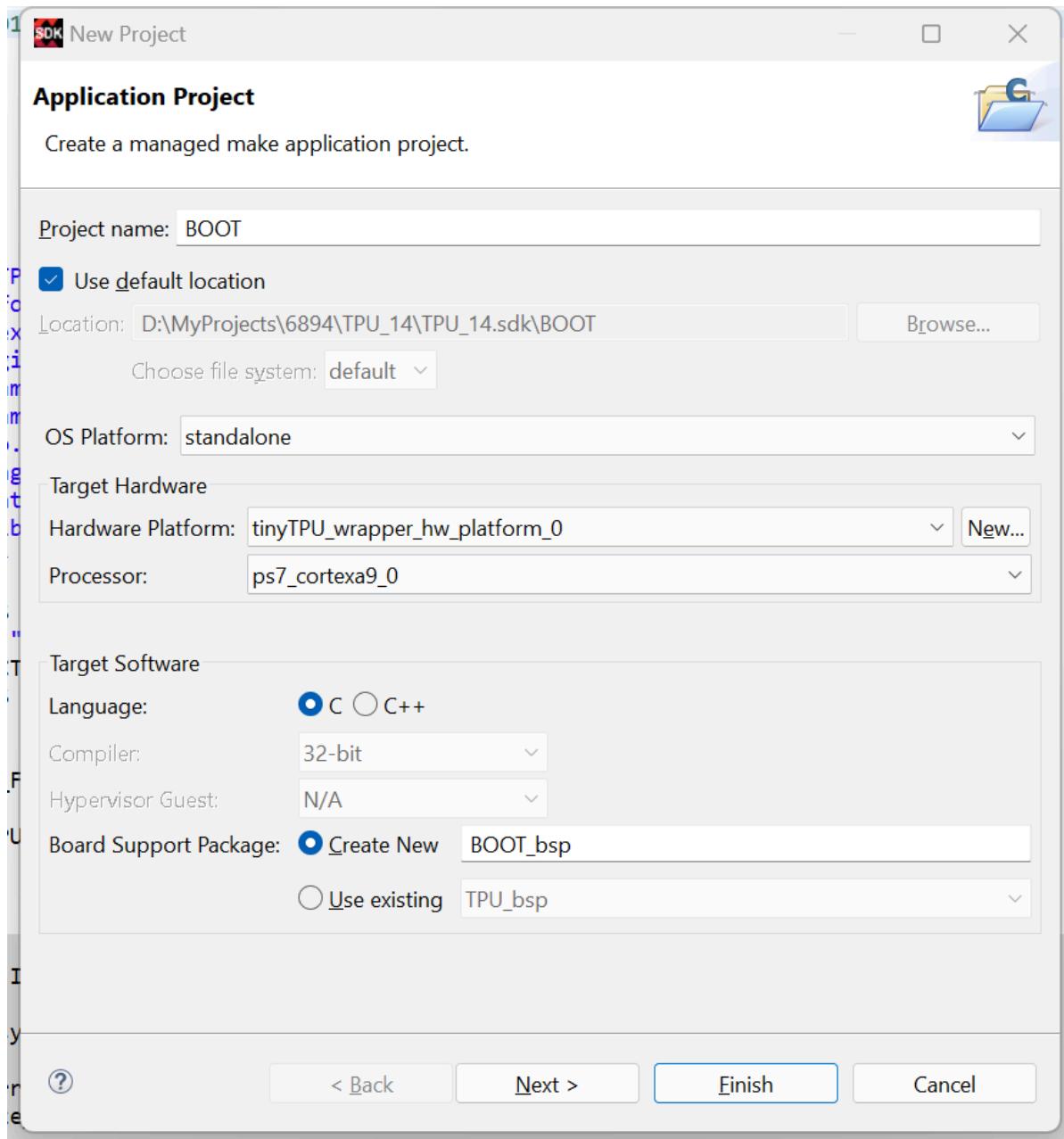
OK Cancel

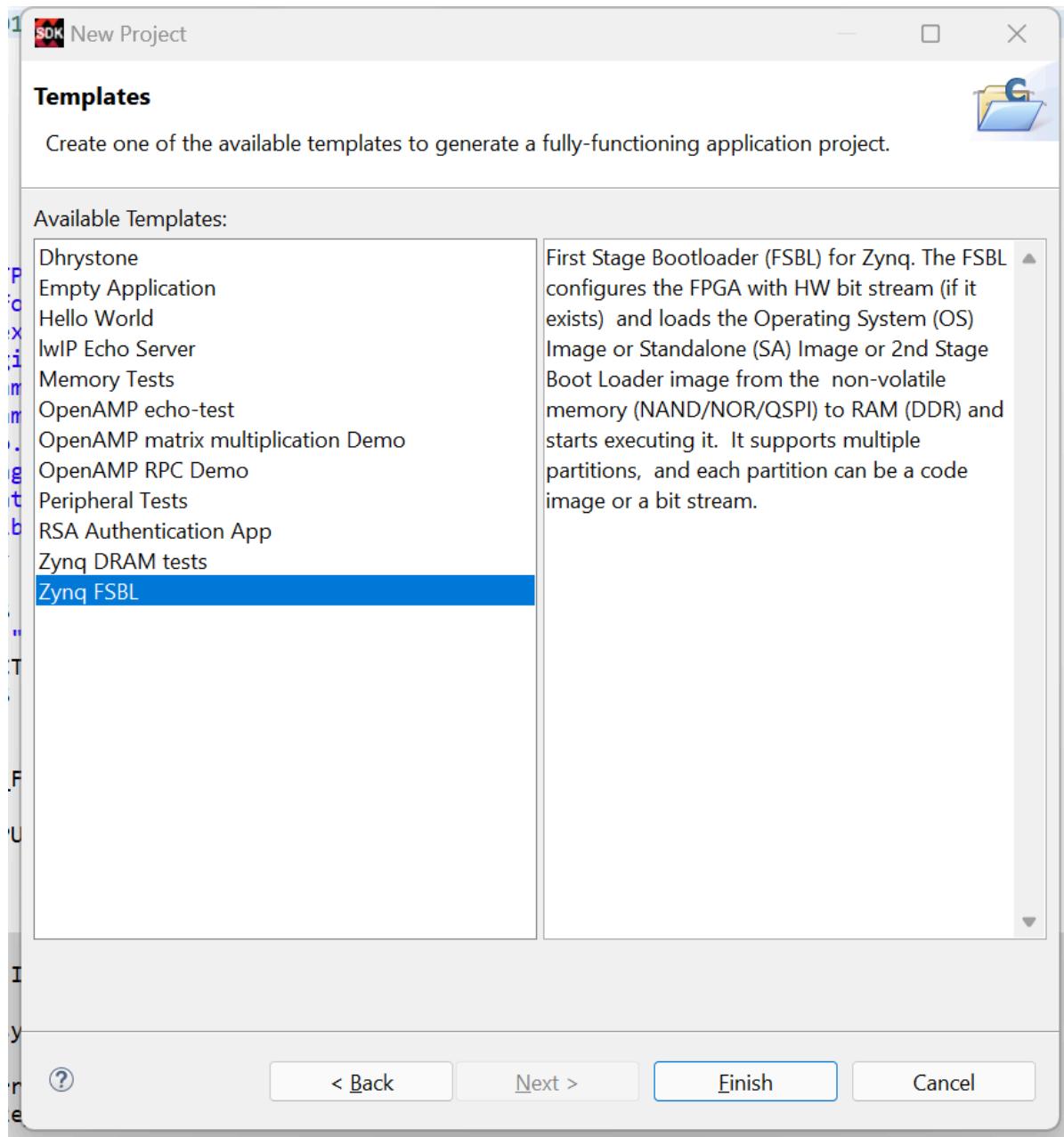
Now right click on the project and choose C/C++ Build Settings. Add the shown line to Debugging.



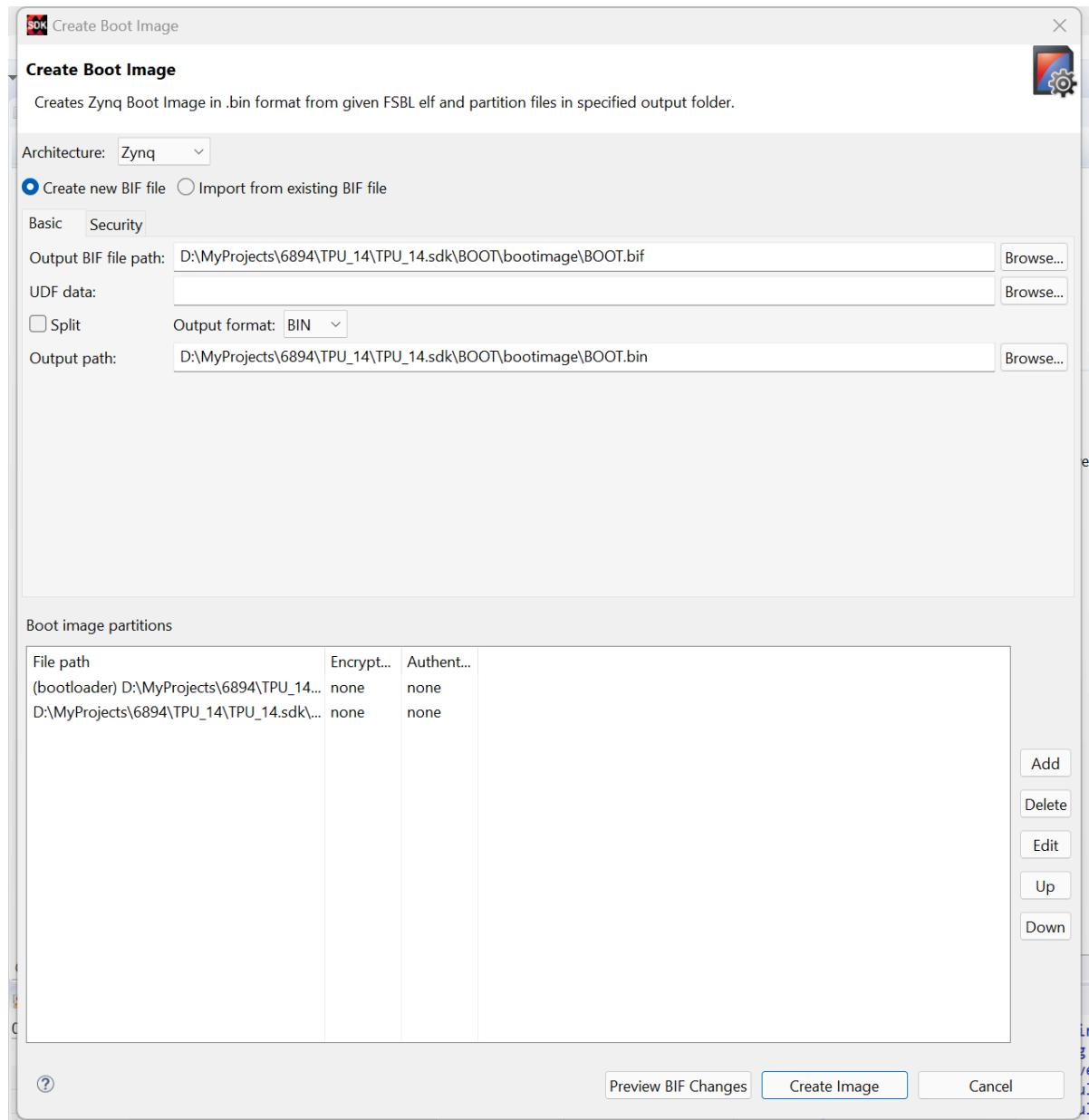
Create FSBL and Boot Image

Create a new application project with First Stage Bootloader (Zynq FSBL). Wait for a while for generation.

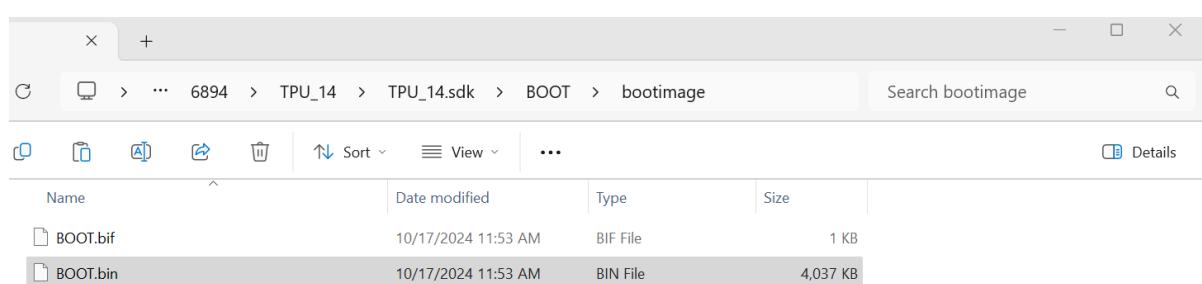




Right click on the main project and select Create Boot Image. The FSBL and Bitstream will be selected automatically.



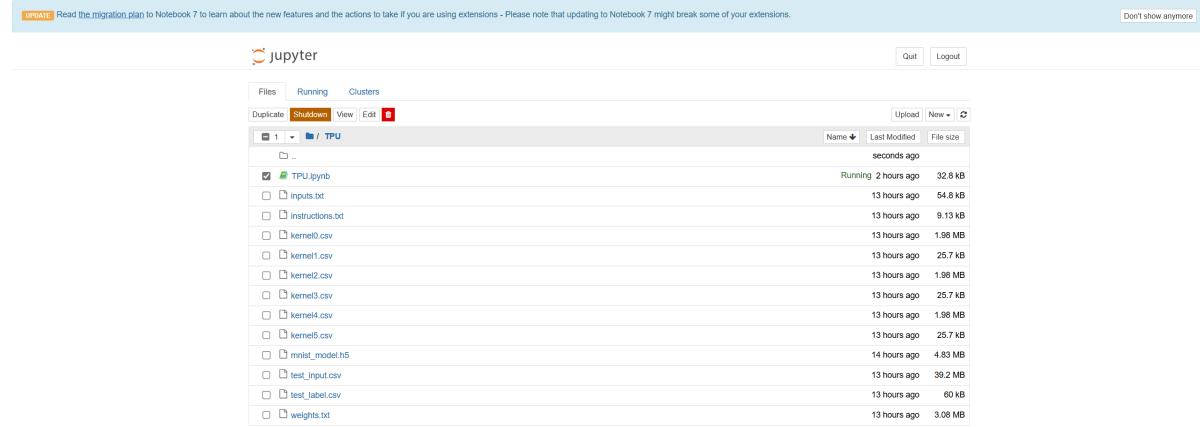
Click Create Image and copy the generated file BOOT.bin to the SD card.



Congratulations, You have generated software system on ZYNQ. Select boot mode as boot from SD card with red switch on the edge of board. Inject SD card, power on and then you can see the Yellow LED on the core board will be bright. If not, press reset key on core board and see if the LED is bright. If it is bright, it means PS is working.

Model of Machine learning

Open Jupyter Notebook and run TPU.ipynb and you will see the follow document generated.



Name	Last Modified	File size
TPU.ipynb	2 hours ago	32.8 kB
inputs.txt	13 hours ago	54.8 kB
instructions.txt	13 hours ago	9.13 kB
kernel0.csv	13 hours ago	1.98 MB
kernel1.csv	13 hours ago	25.7 kB
kernel2.csv	13 hours ago	1.98 MB
kernel3.csv	13 hours ago	25.7 kB
kernel4.csv	13 hours ago	1.98 MB
kernel5.csv	13 hours ago	25.7 kB
mnist_model.h5	14 hours ago	4.83 MB
test_input.csv	13 hours ago	39.2 MB
test_label.csv	13 hours ago	60 kB
weights.txt	13 hours ago	3.08 MB

Copy weights.txt, inputs.txt and instructions.txt into SD card.

Notice, we can change the instructions.txt name as instruct.txt, because the name of file should be less than 8 characters, or it will be an error. Then remove all lines in weights.txt with all 0 lines and remain 28729 lines, or there will be too much "bad address" warning when running.