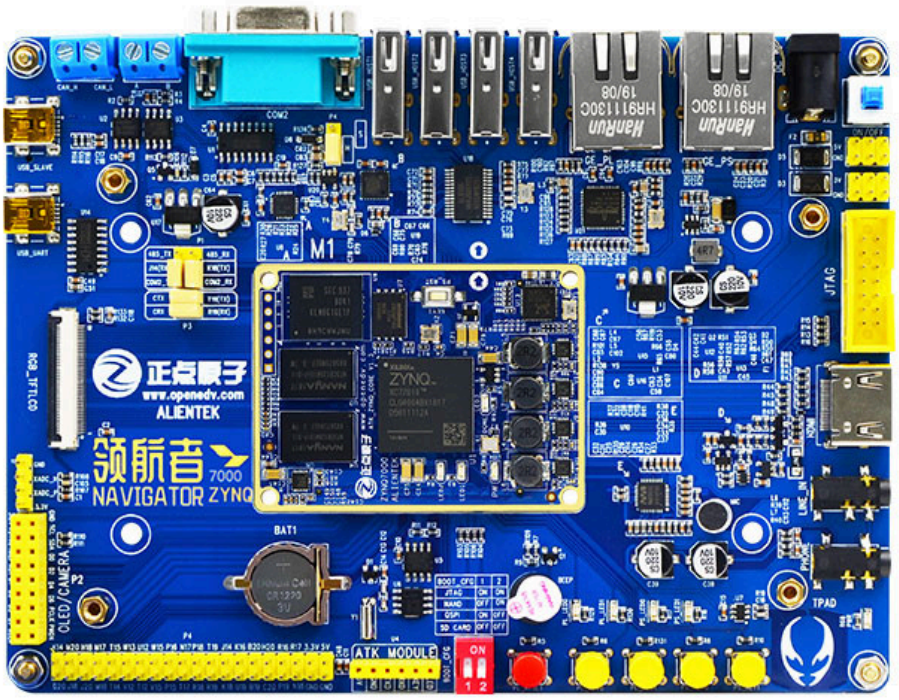
tinyTPU Project Artifacts

How to start

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

Hardware Design

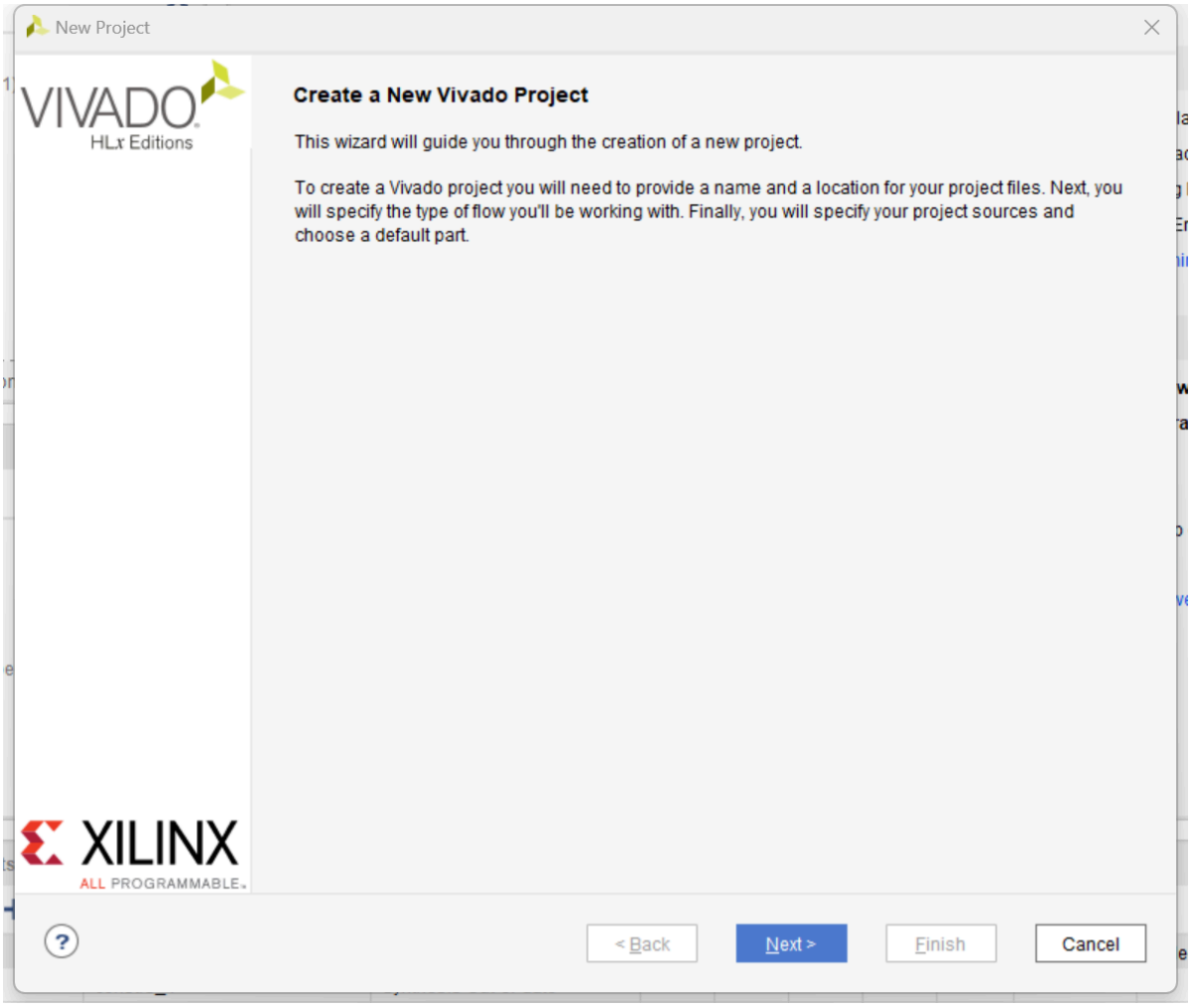
Hardware Platform Introduction



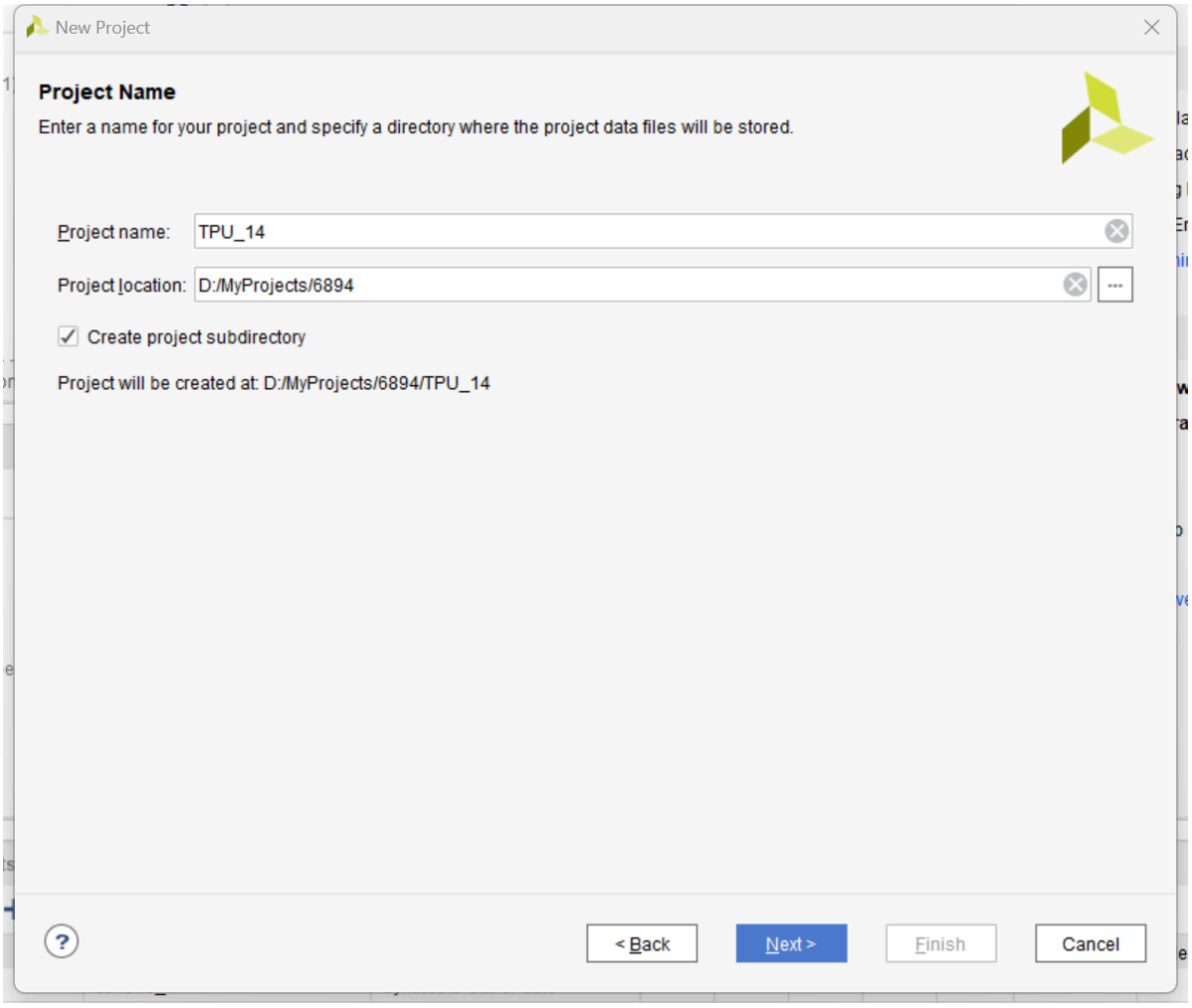
Hardware platform is Navigator ZYNQ7z020 board from ALENTEX v1. Development platform is VIvado 2017.4. You can find this board in www,openedv.com.

Create a Project

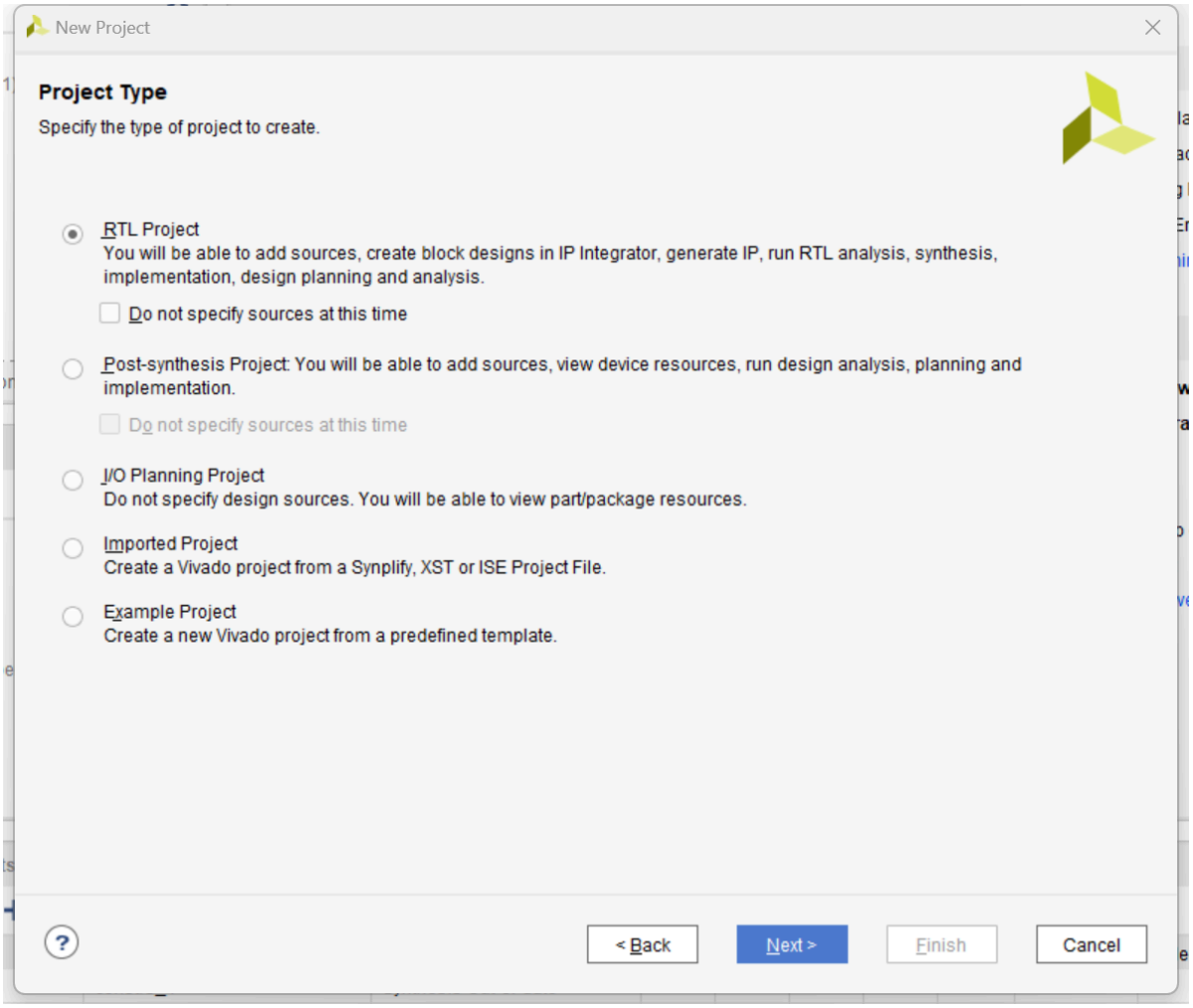
Click file -> new project or click create new project in main page.


New Project
VIVADO
HLx Editions
Create a New Vivado Project
This wizard will guide you through the creation of a new project.
To create a Vivado project you will need to provide a name and a location for your project files. Next, you will specify the type of flow you'll be working with. Finally, you will specify your project sources and choose a default part.
XILINX
ALL PROGRAMMABLE
< Back
Next >
Finish
Cancel

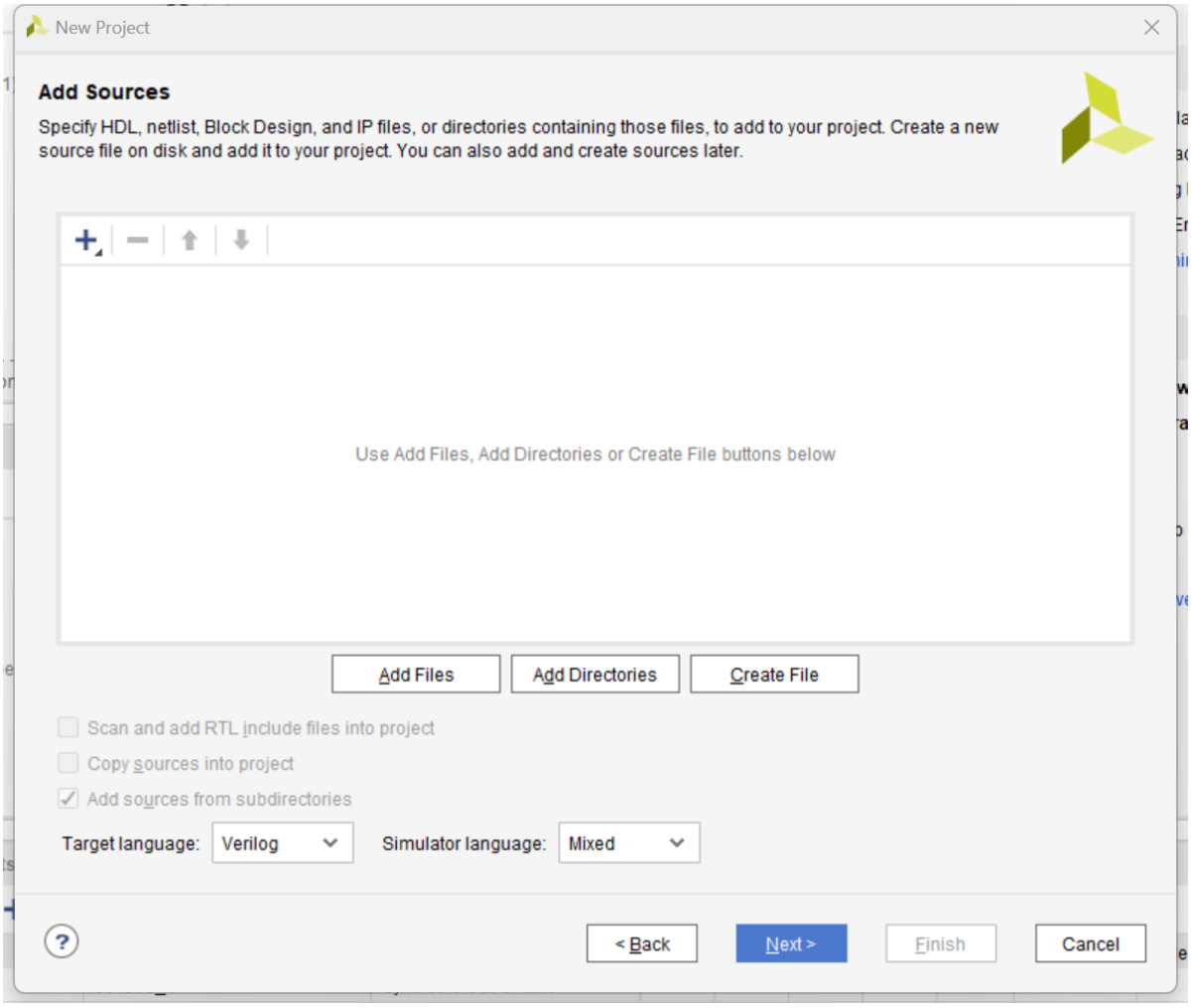
Click next.


New Project
Project Name
Enter a name for your project and specify a directory where the project data files will be stored.
Project name:
TPU_14
Project location:
D:/MyProjects/6894
Create project subdirectory
Project will be created at: D:/MyProjects/6894/TPU_14
< Back
Next >
Finish
Cancel

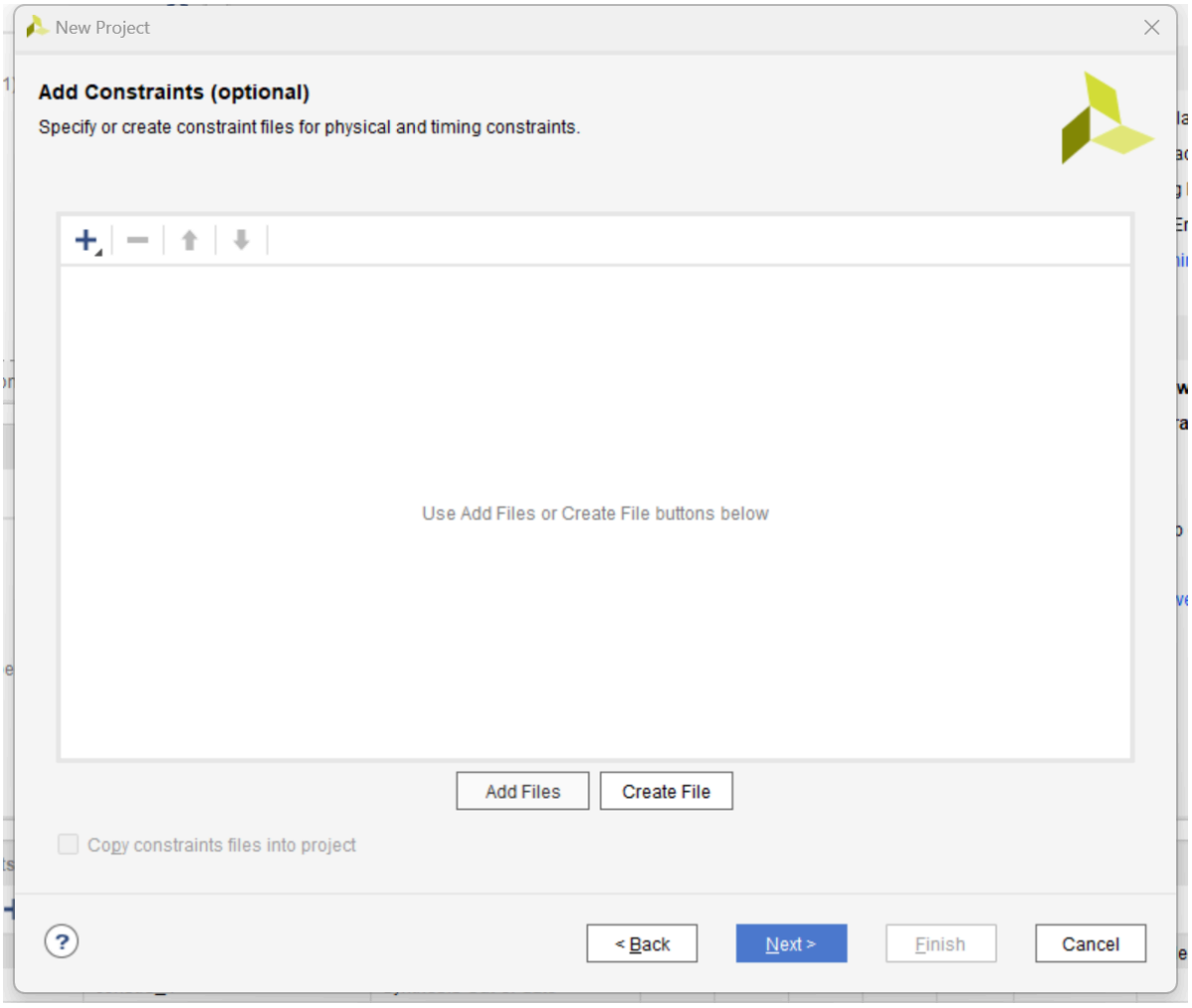
Rename your project as TPU_14 and press next.



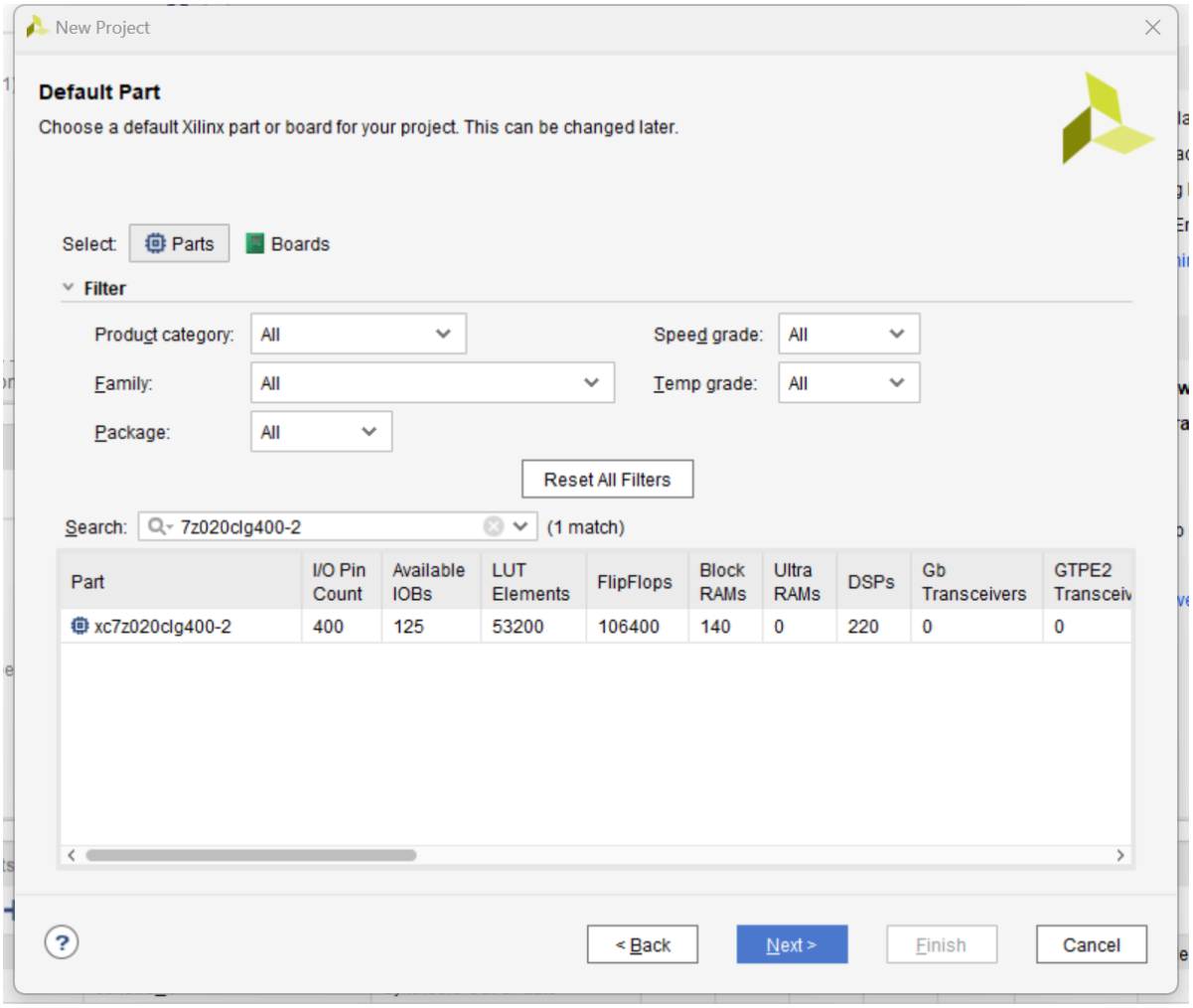
Choose RTL Project, and click next.


New Project
Add Sources
Specify HDL, netlist, Block Design, and IP files, or directories containing those files, to add to your project. Create a new source file on disk and add it to your project. You can also add and create sources later.
Use Add Files, Add Directories or Create File buttons below
Add Files
Add Directories
Create File
Scan and add RTL include files into project
Copy sources into project
Add sources from subdirectories
Target language: Verilog
Simulator language: Mixed
< Back
Next >
Finish
Cancel

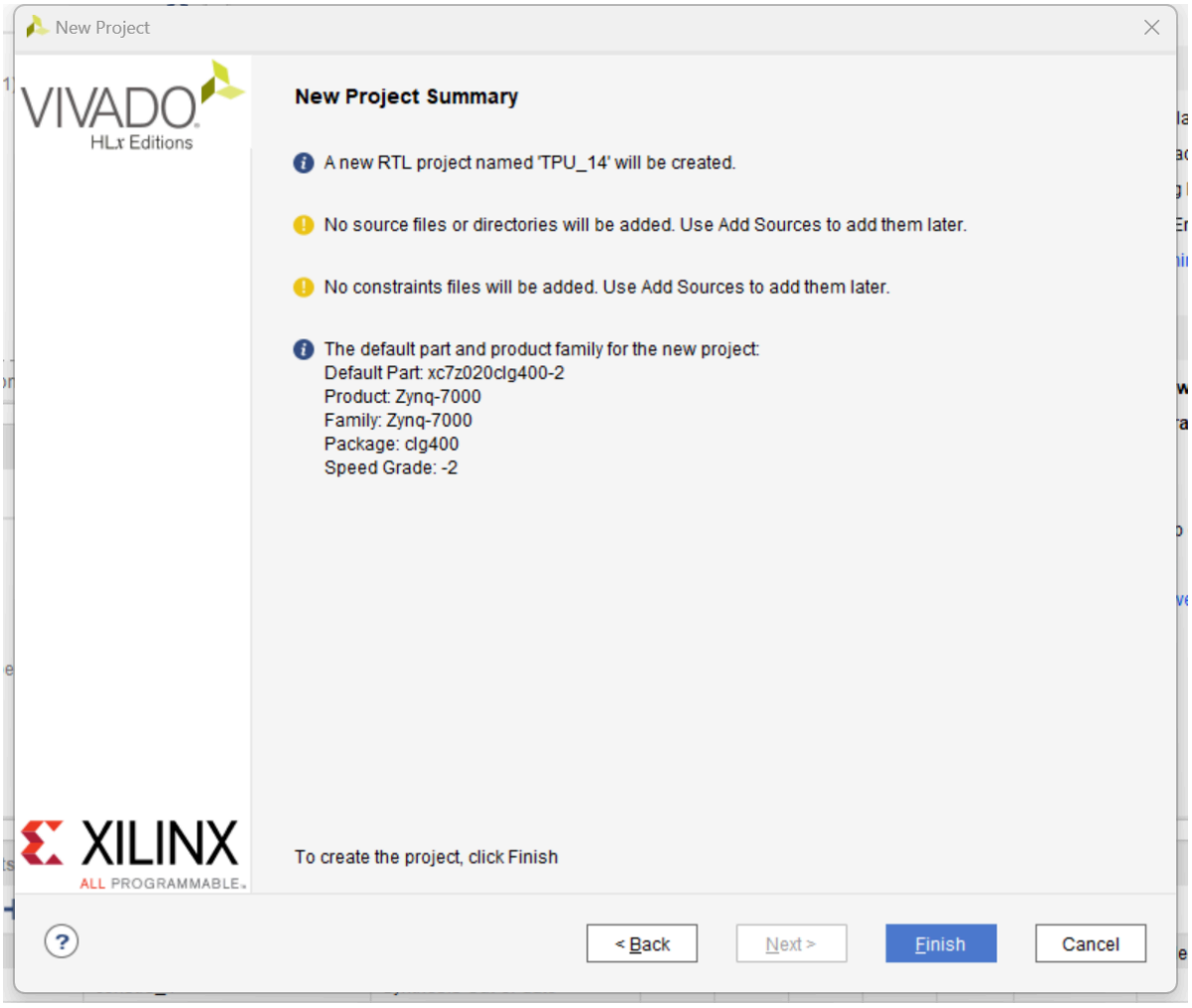
Click next.


New Project
Add Constraints (optional)
Specify or create constraint files for physical and timing constraints.
Use Add Files or Create File buttons below
Add Files
Create File
Copy constraints files into project
< Back
Next >
Finish
Cancel

Click next.



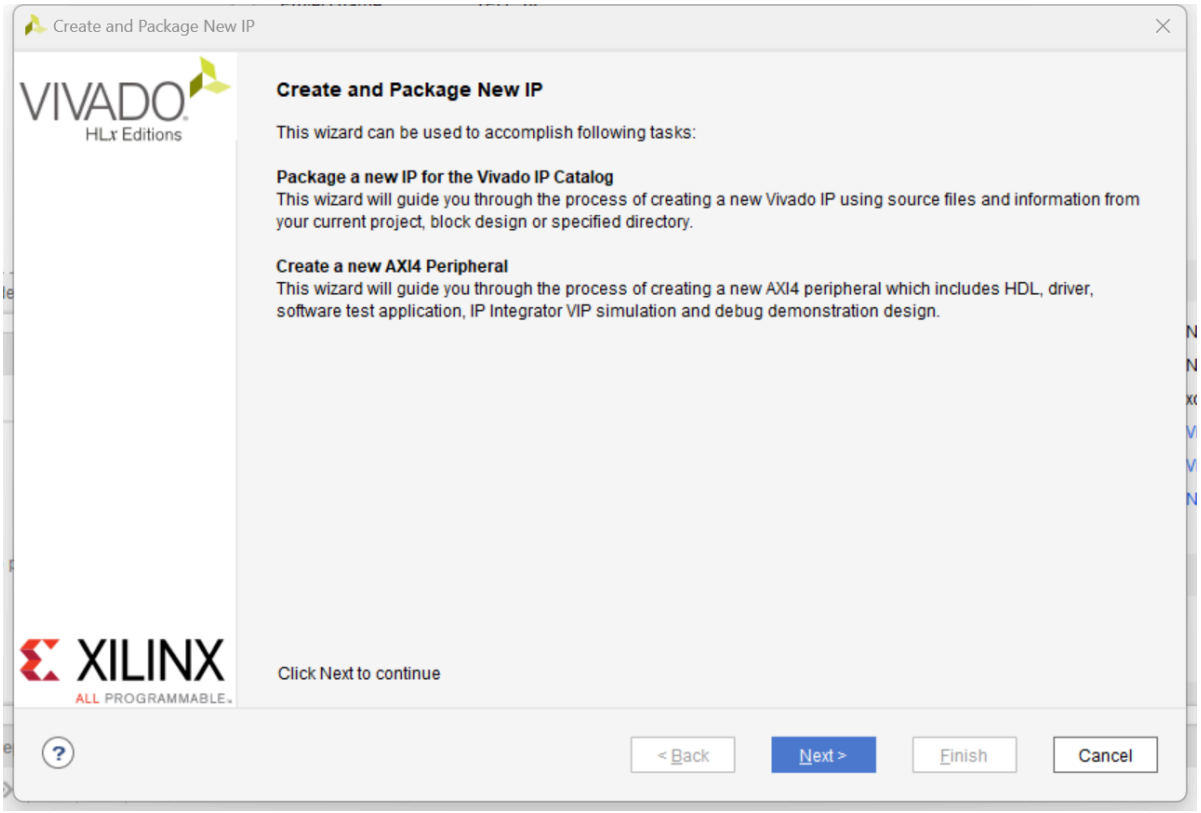
Choose the chip's Part name: xc7z020clg400-2, and choose the part and press next.

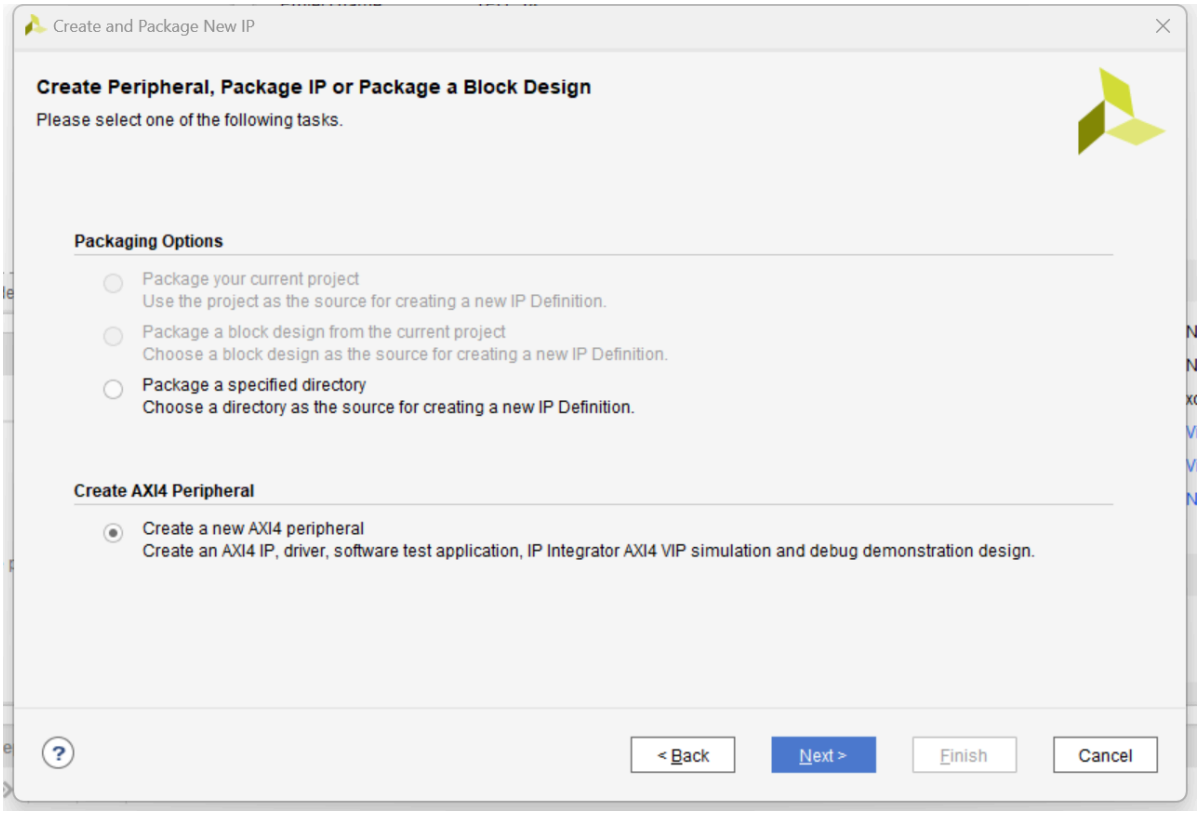


Click finish, and you will create one empty project.

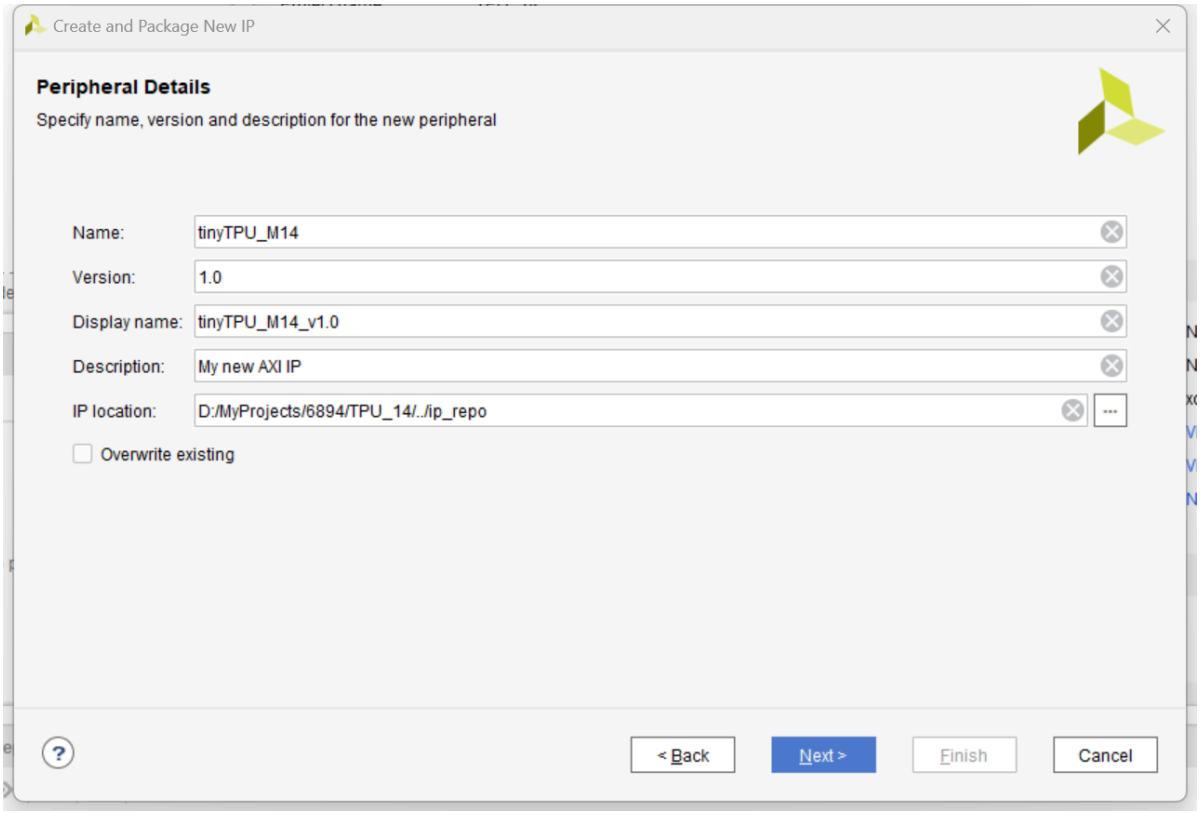
Create IP

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

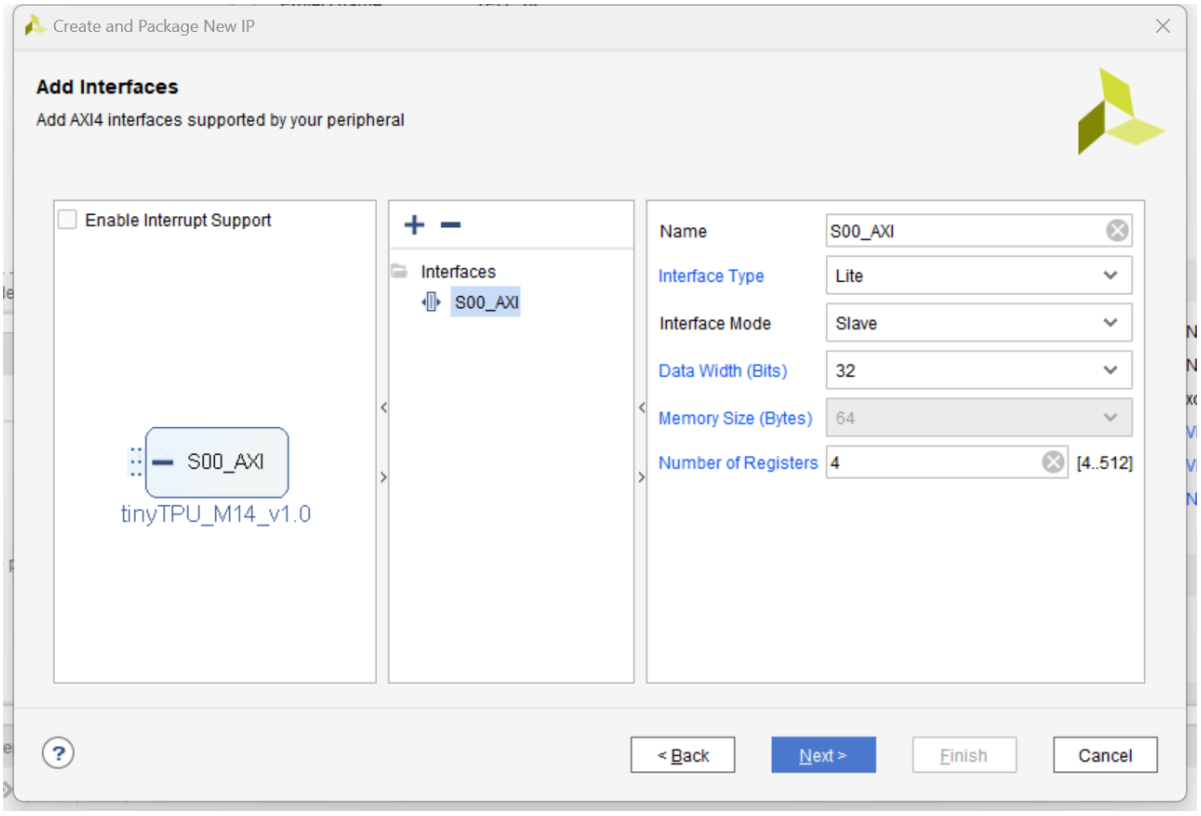




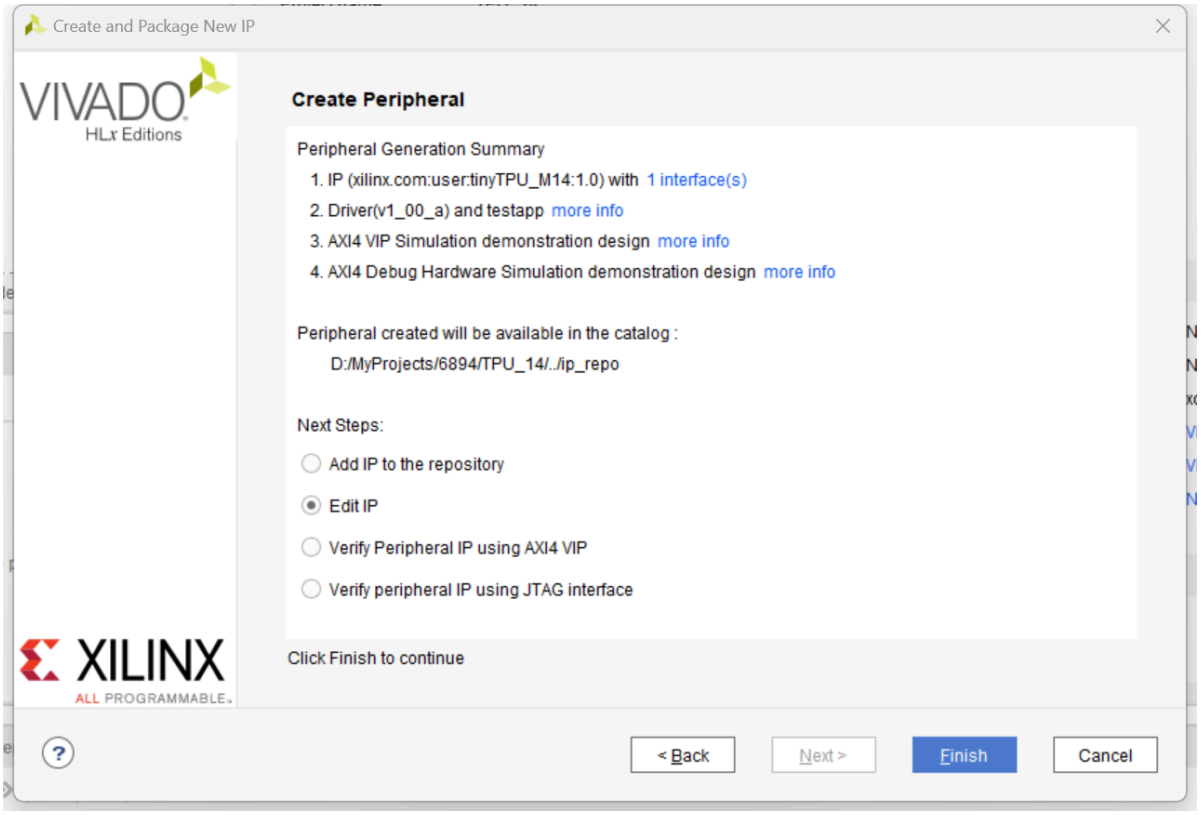
Choose Create a new AXI4 peripheral and click Next.



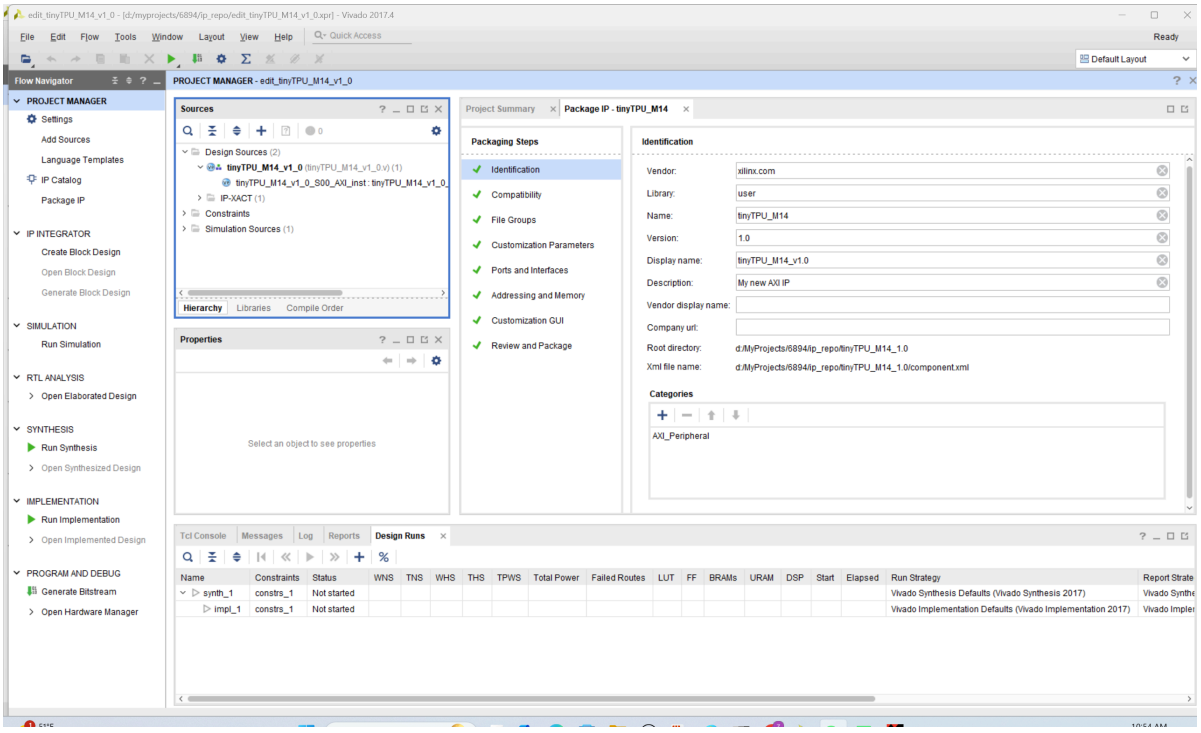
Rename our IP as tinyTPU_M14 and keep others default. Then click Next.



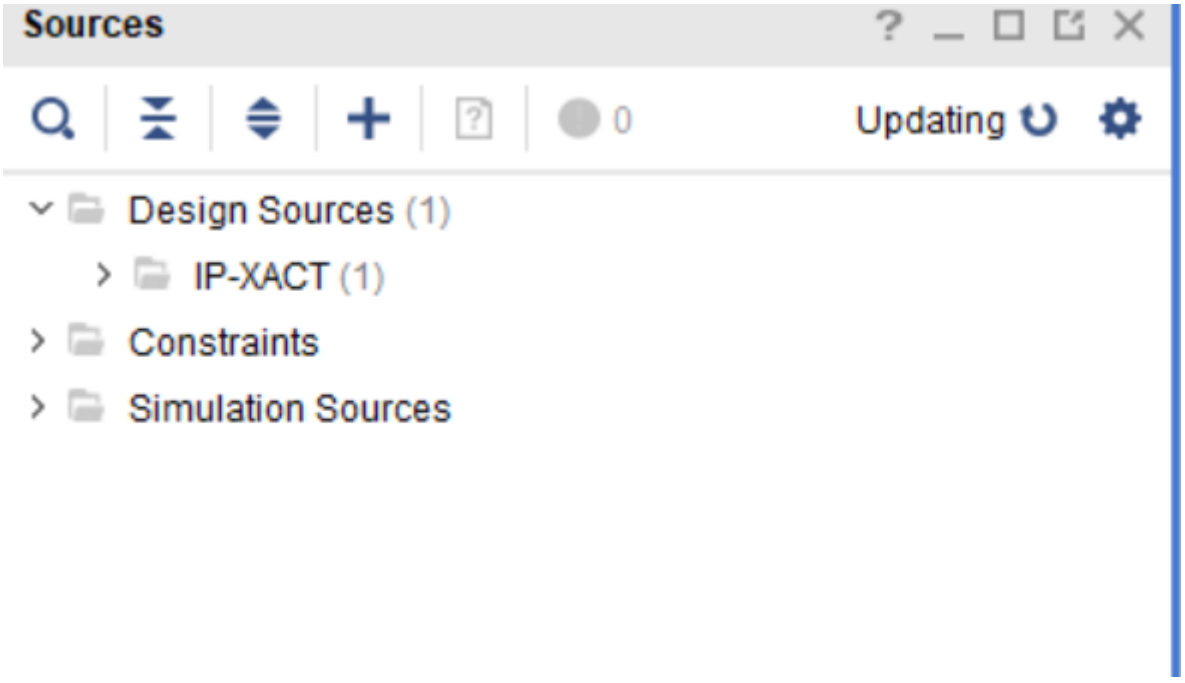
Keep these parameters as default and click Next.



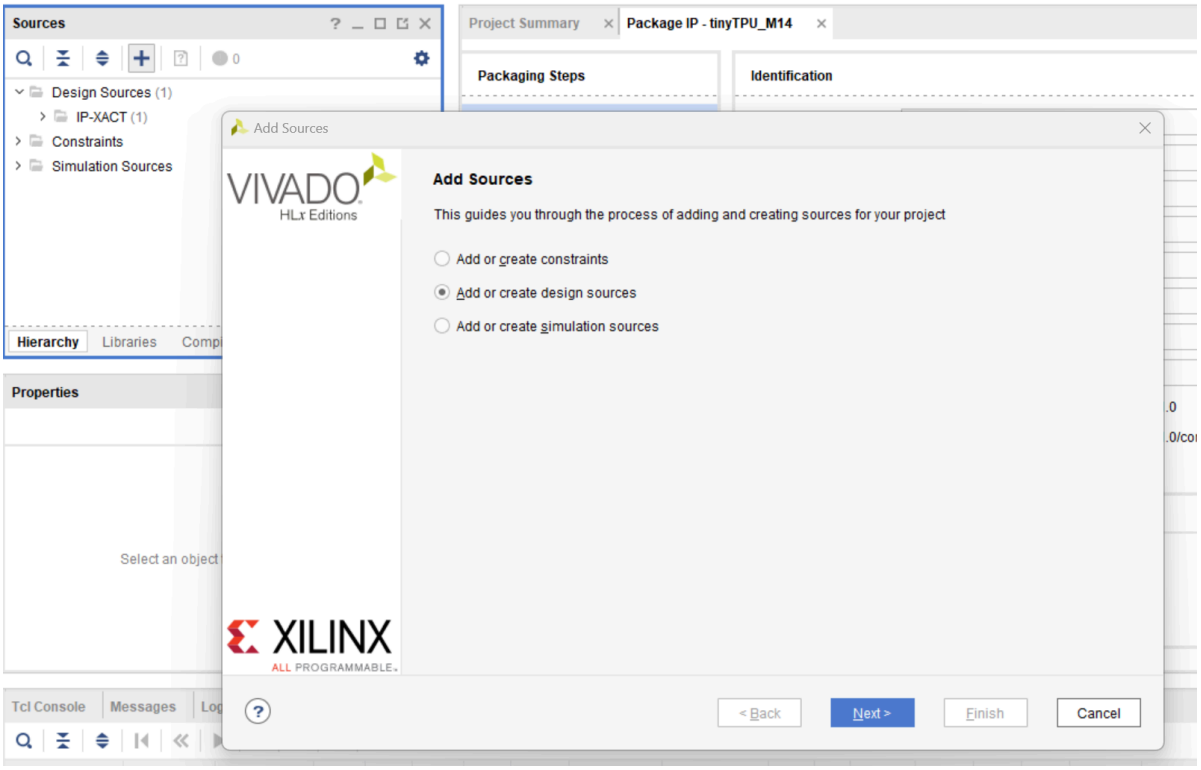
Choose Edit IP and click finish.



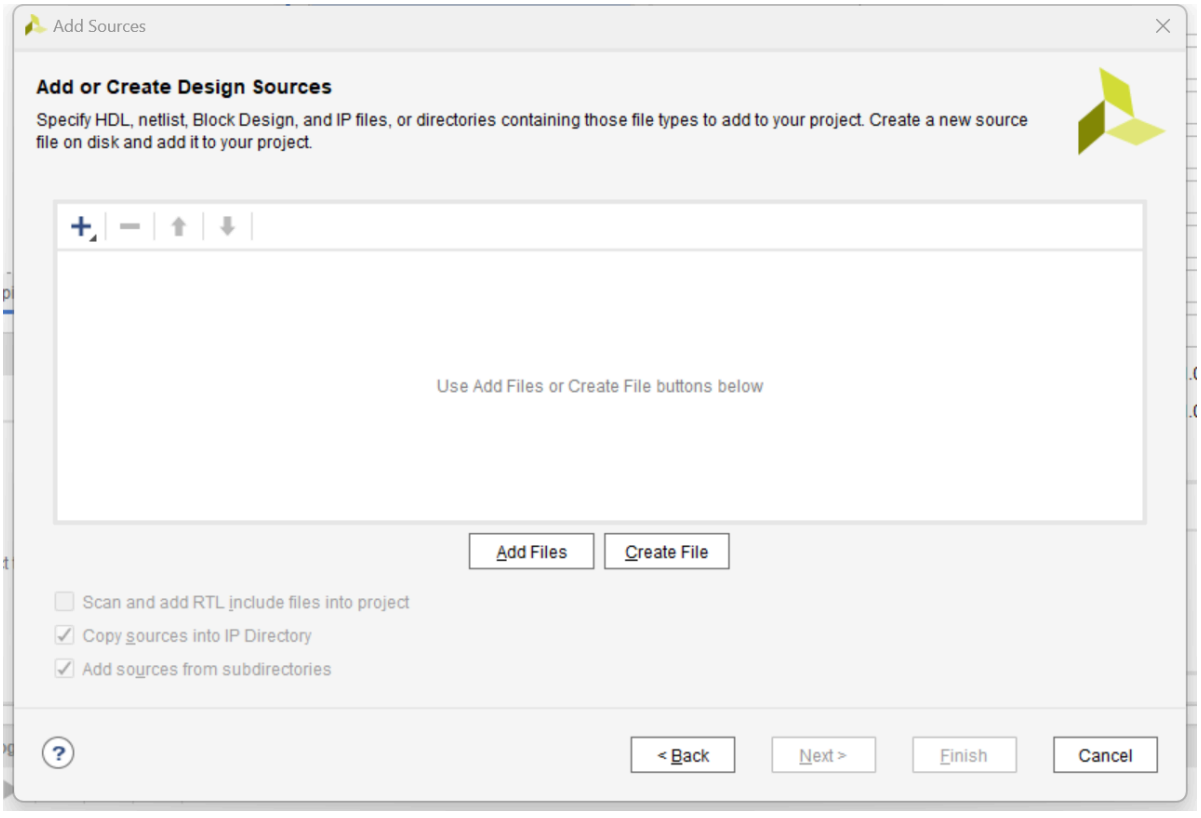
Then, you will find this window will be opened. There will be two generated verilog files, delete them.



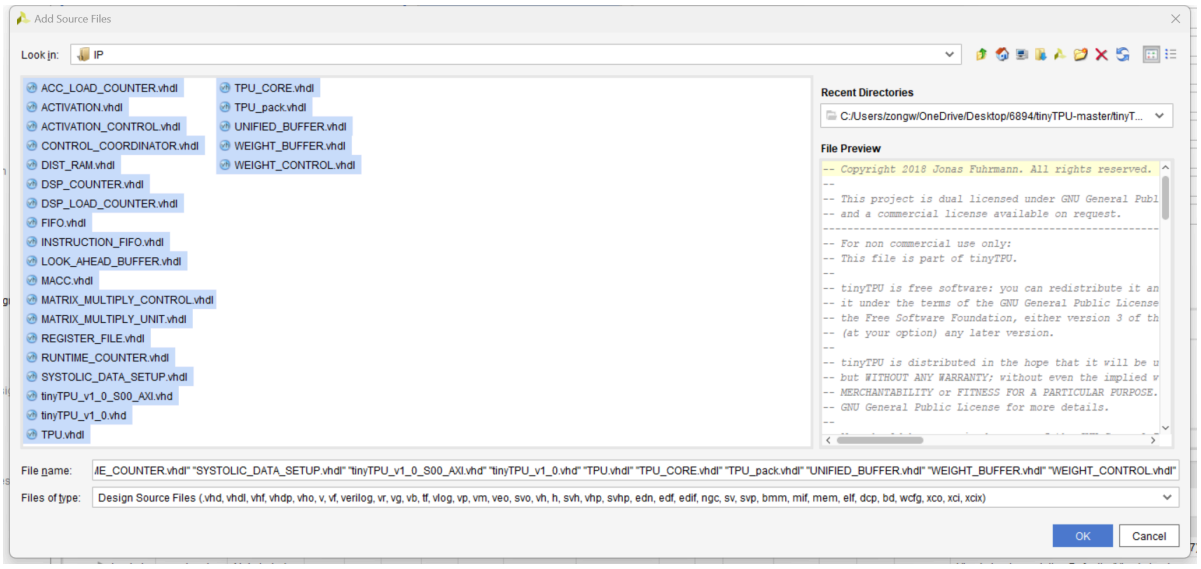
Just right click the files and choose delete and then you will see the empty Sources.



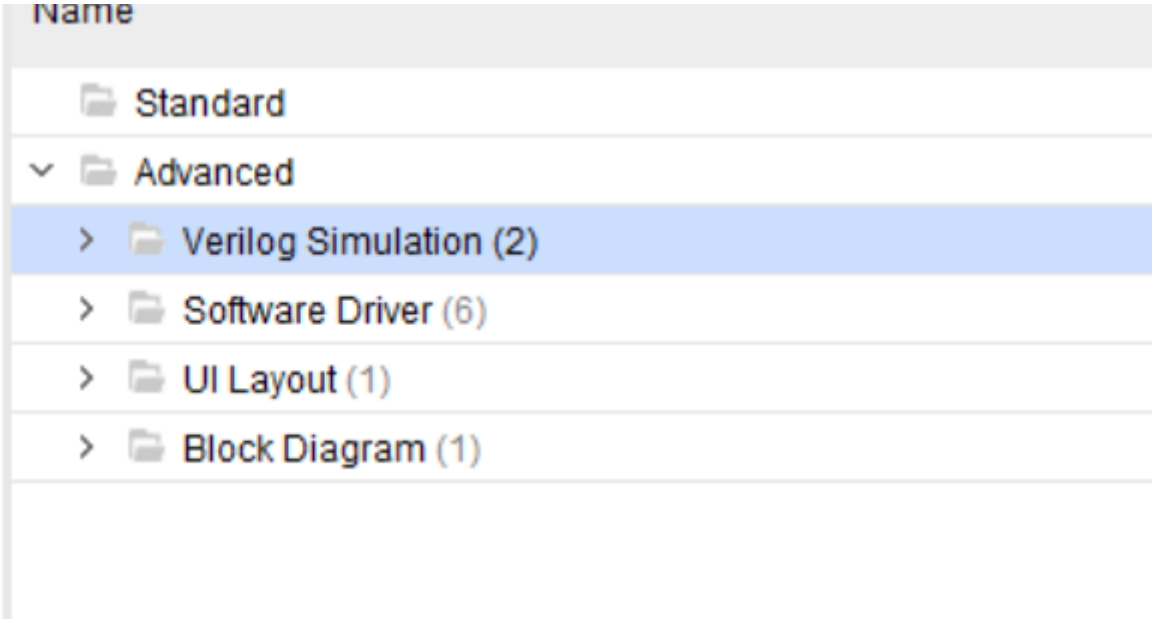
Then, click the "+" to add our VHDL IP source. Choose Add or create design sources, then click Next.



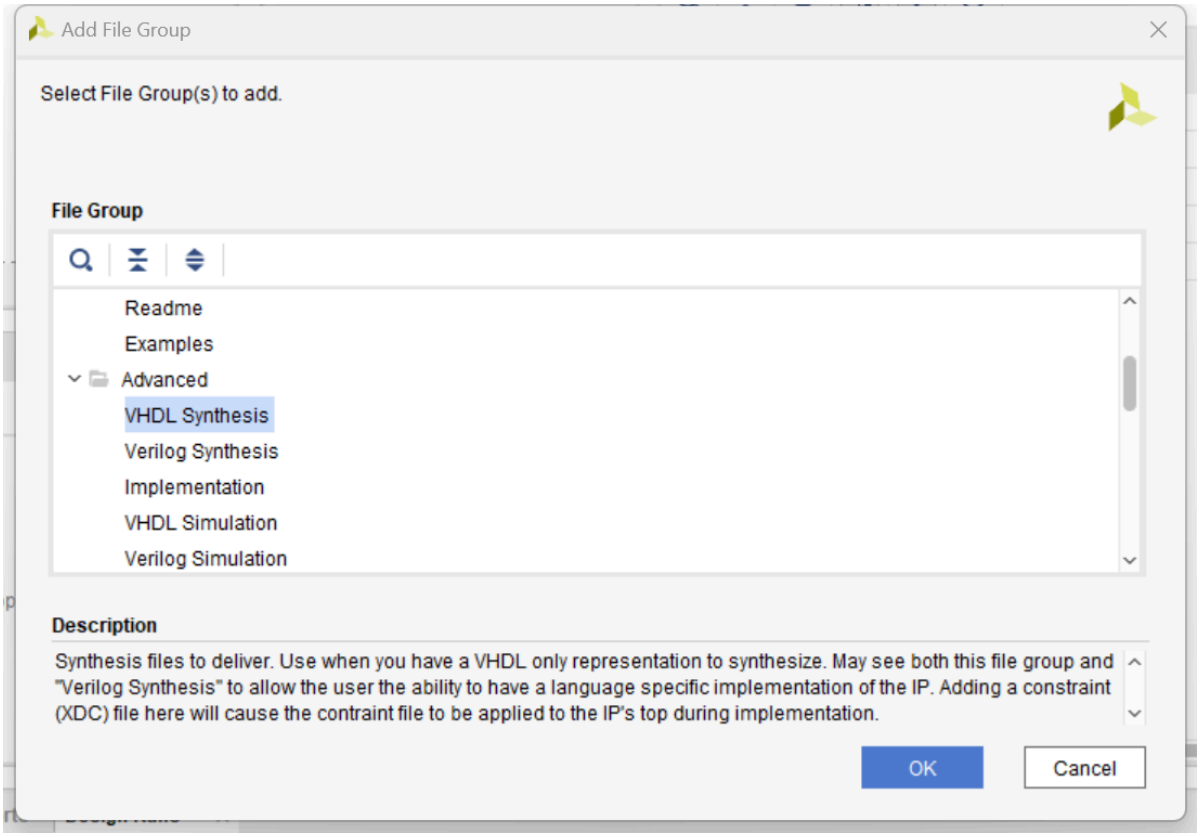
Click Add FIles.



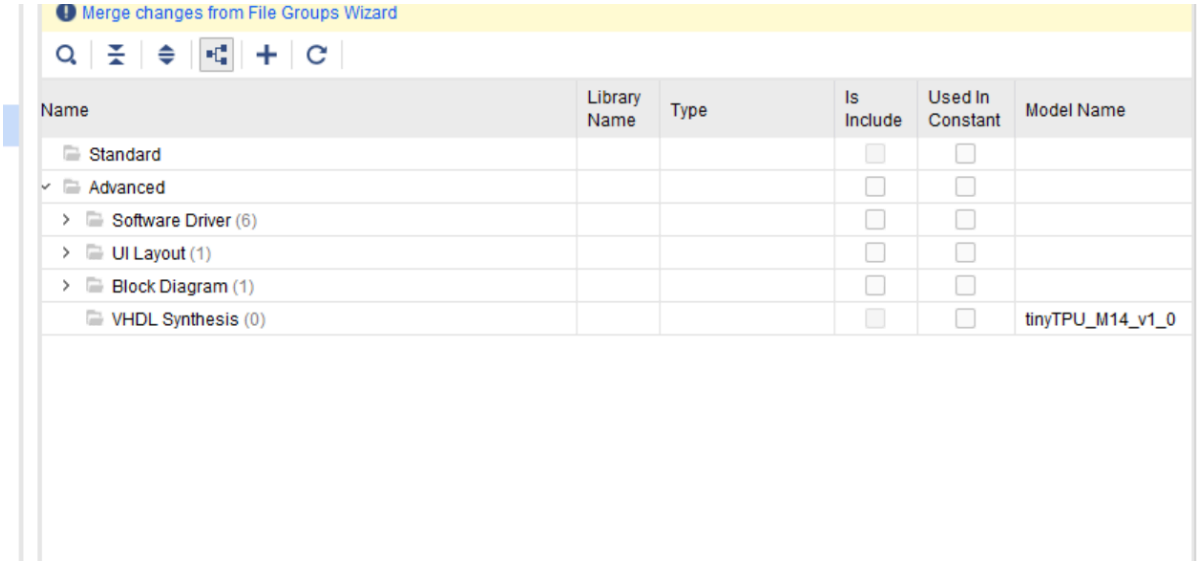
Select all vhdl files down loaded from github and click Ok.



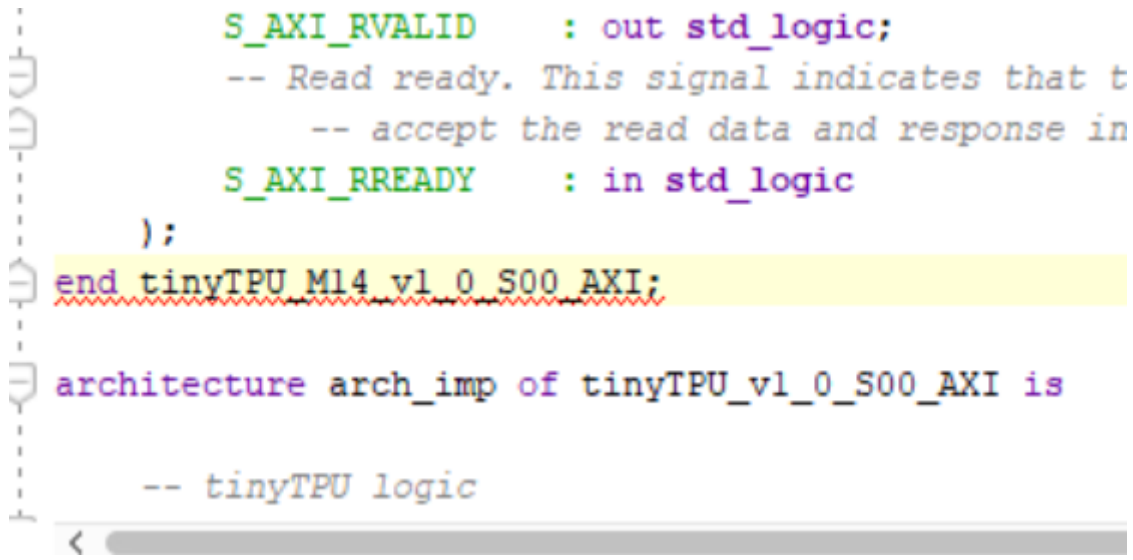
Then you must remove Verilog SImulation group and Verilog Synthesis group.



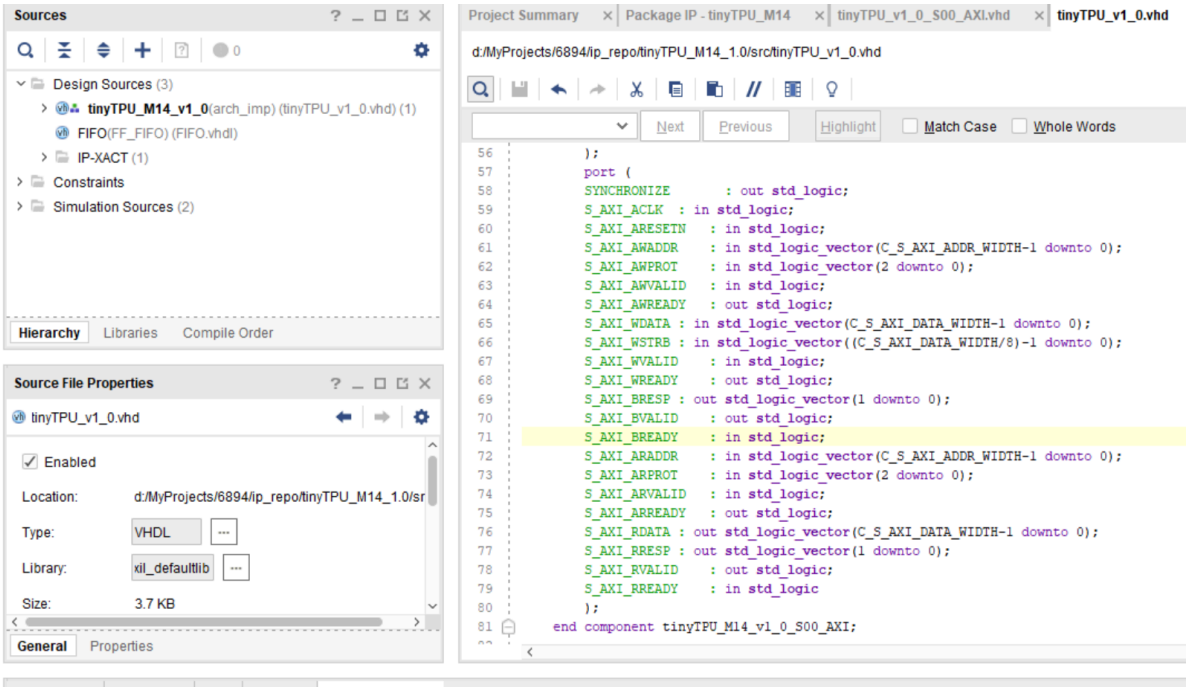
Then add new VHDL Synthesis group and click OK.



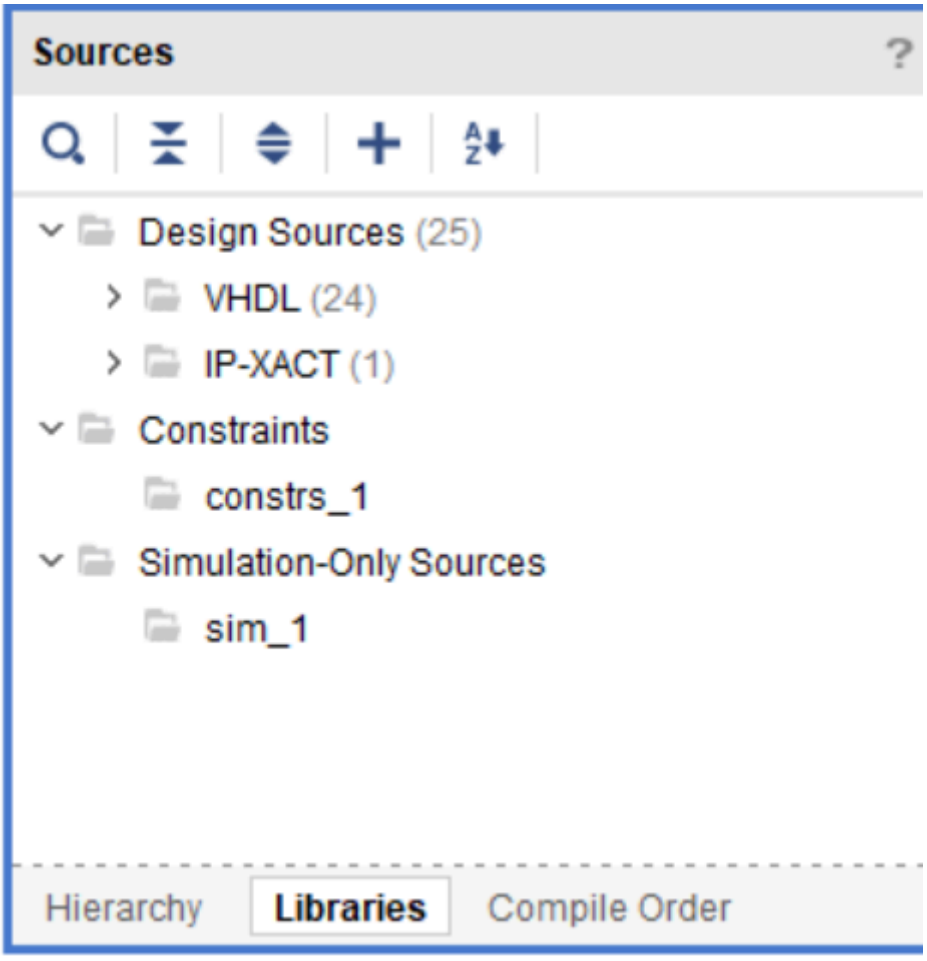
Rename model name as tinyTPU_M14_v1_0.



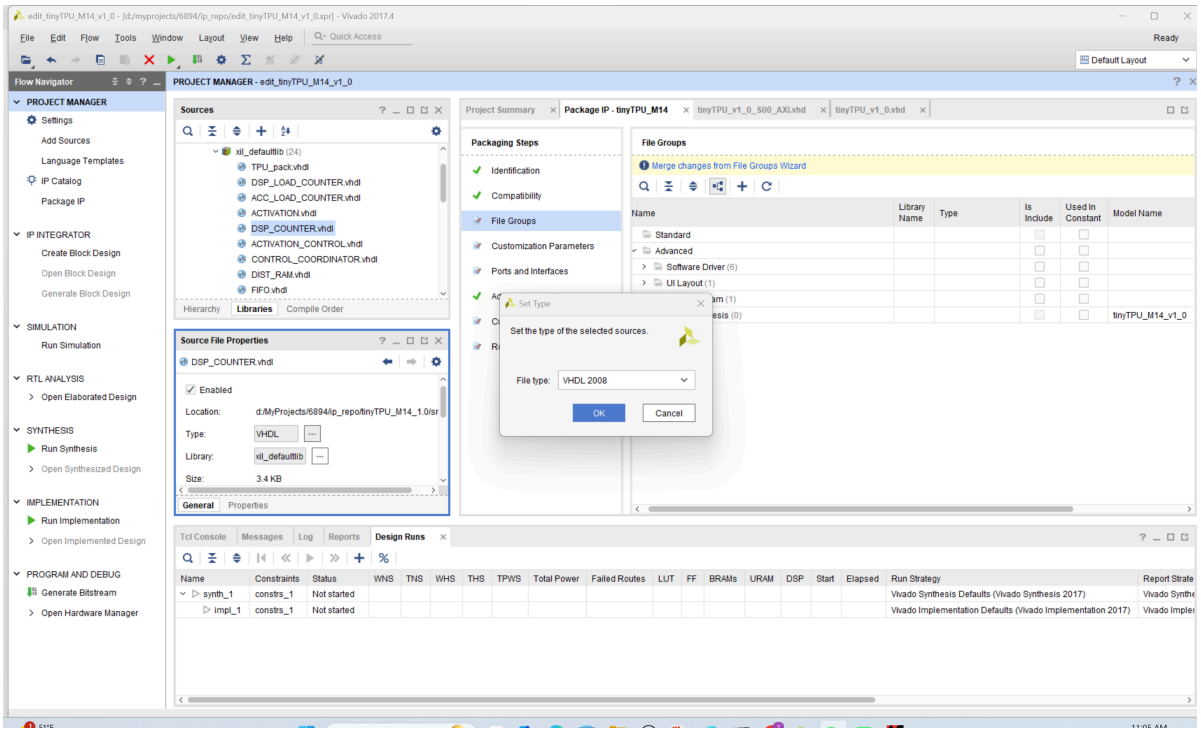
Add _M14 in all tinyTPU_V1_0 like this in picture.



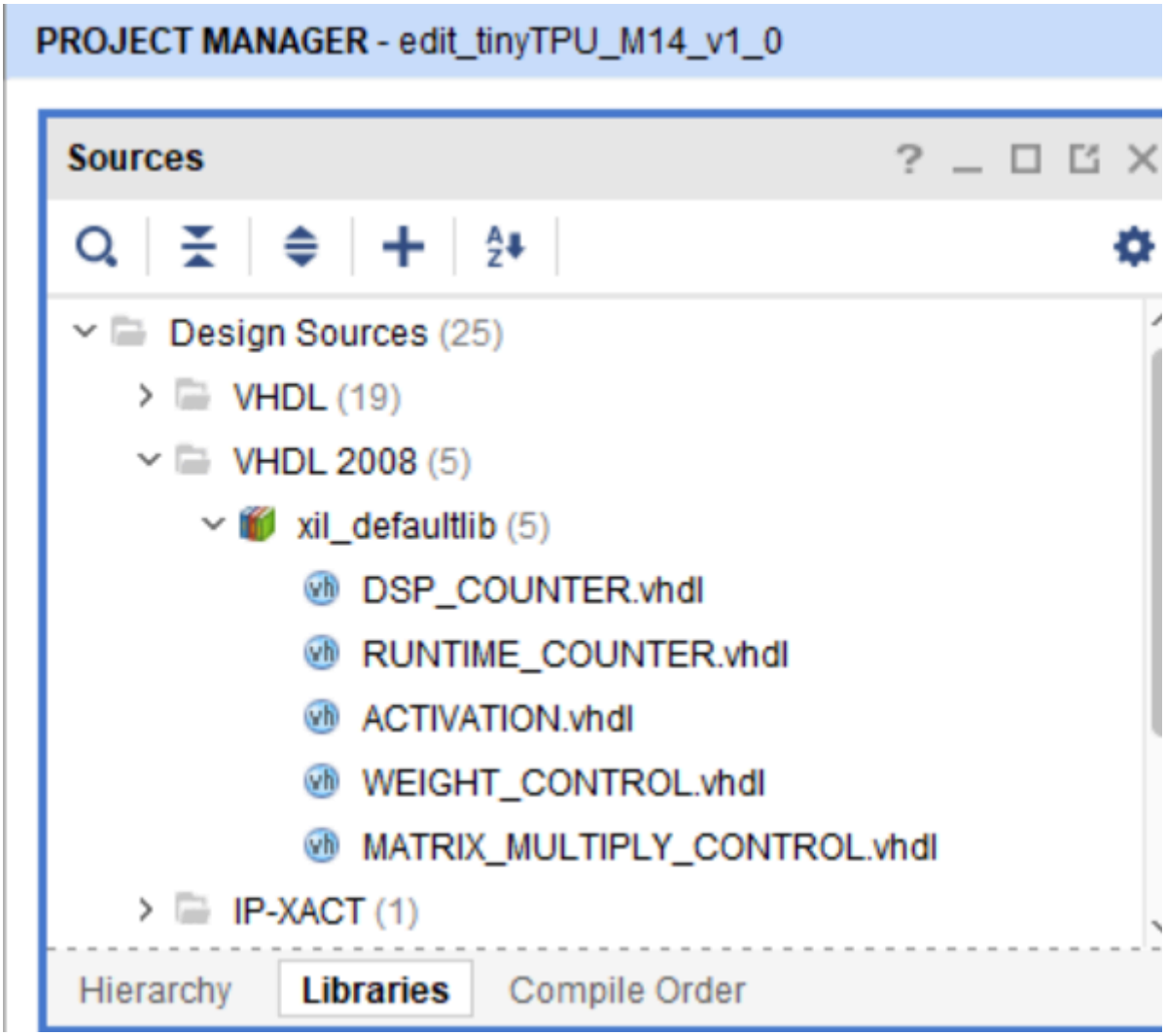
Two AXI files should be changed, then there will be no errors and Design Sources will add _M14 in the name.



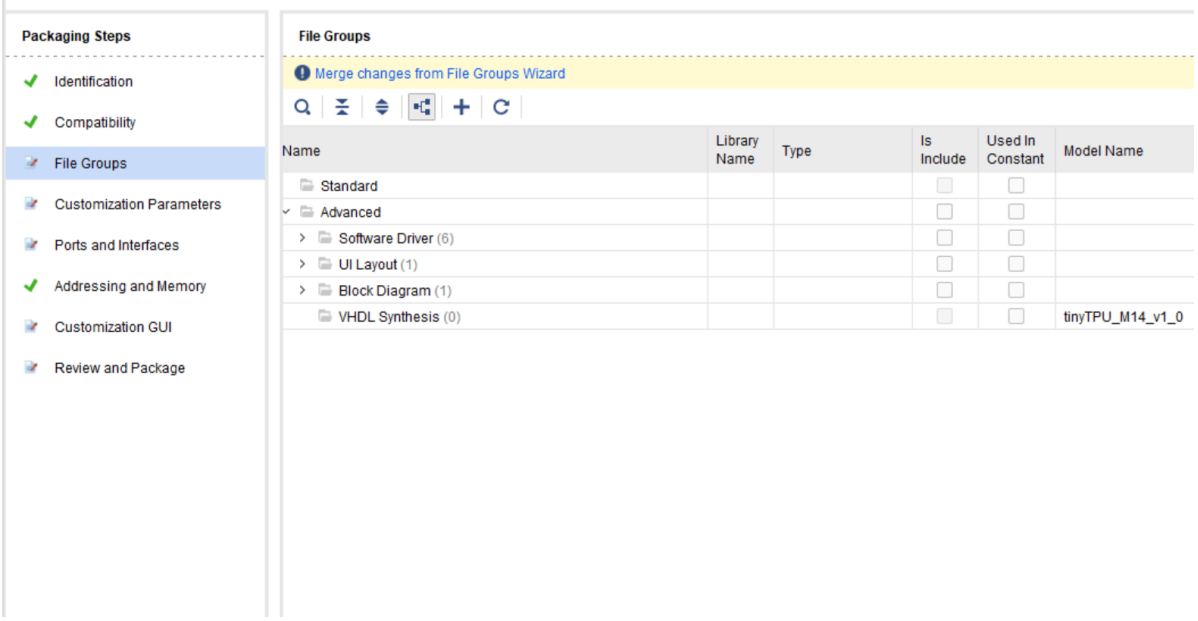
You can double check in Libraries and you can see VHDL(24).



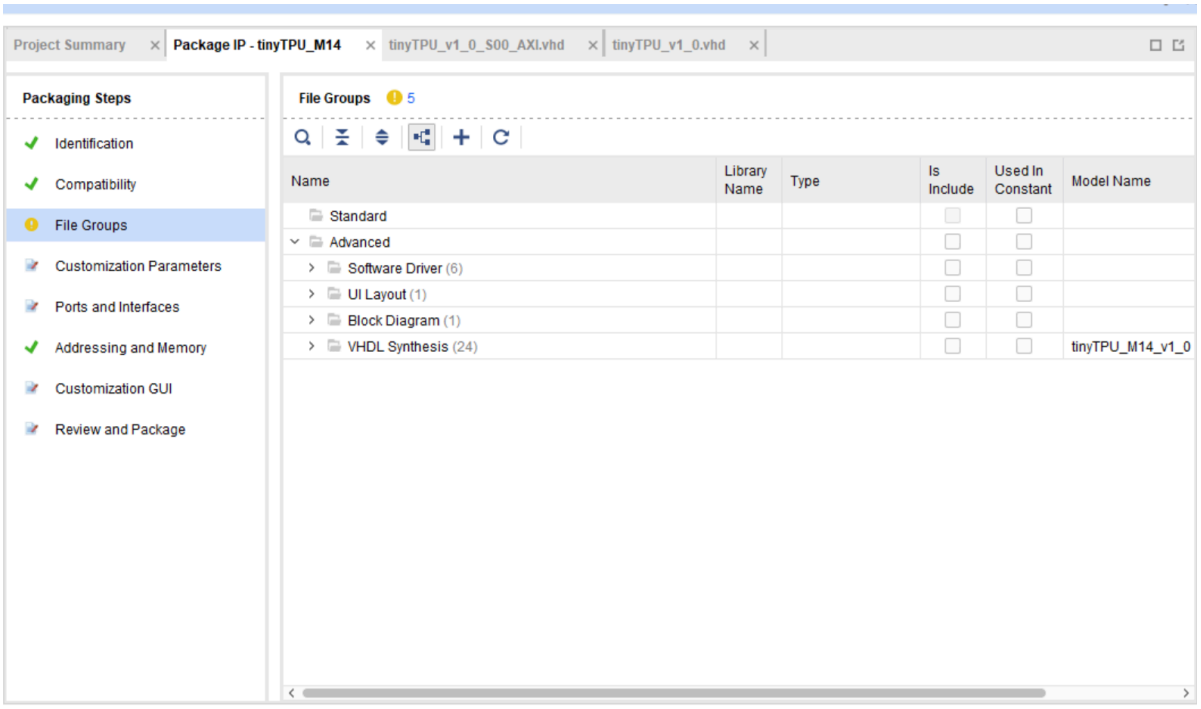
Then change five files type name from VHDL to VHDL2008. Includes **DSP_COUNTER, RUNTIME_COUNTER, ACTIVATION, WEIGHT_CONTROL and MATRIX_MULTIPLY_CONTROL**.



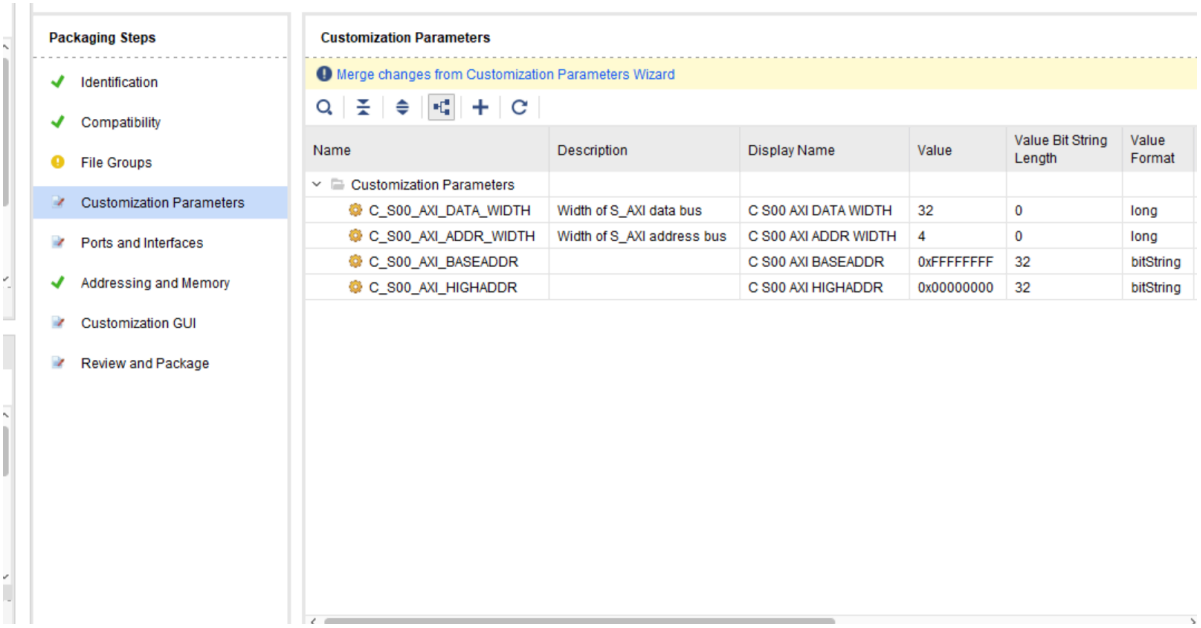
After change, you can see there will be five VHDL 2008 type files and 19 VHDL files.



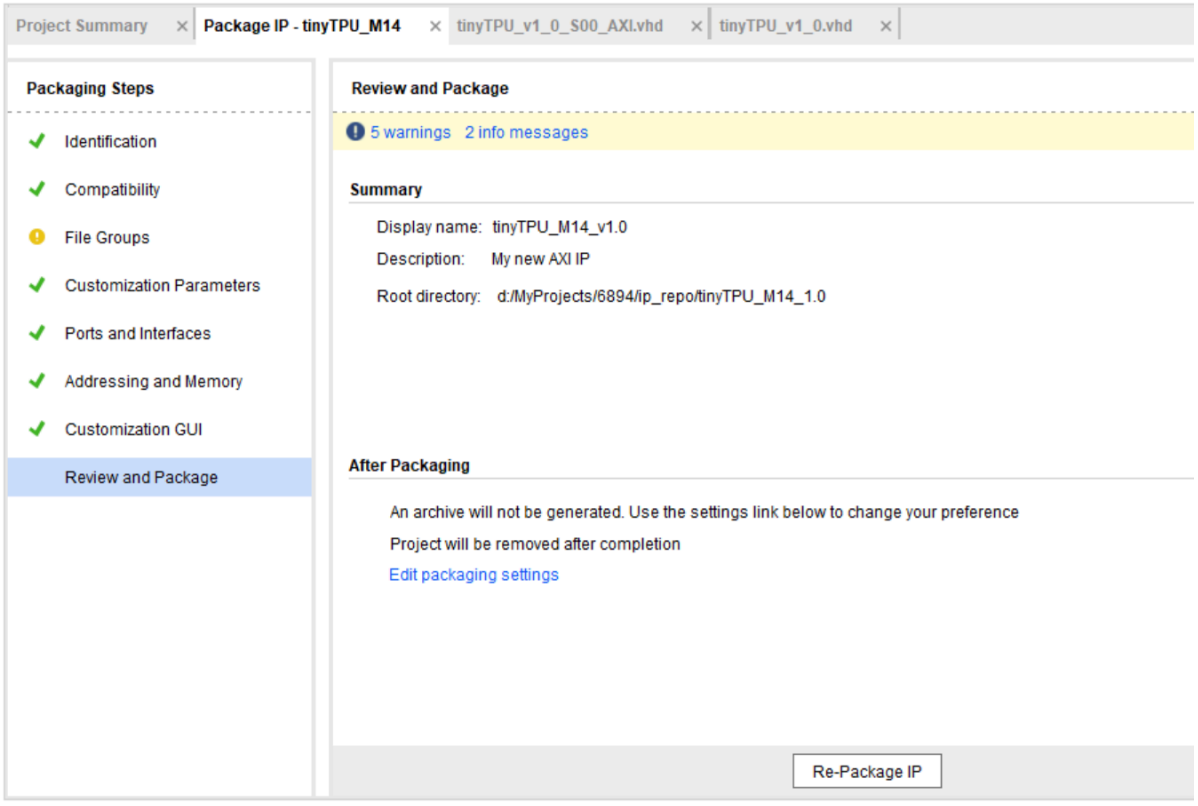
Then we can merge File Groups.



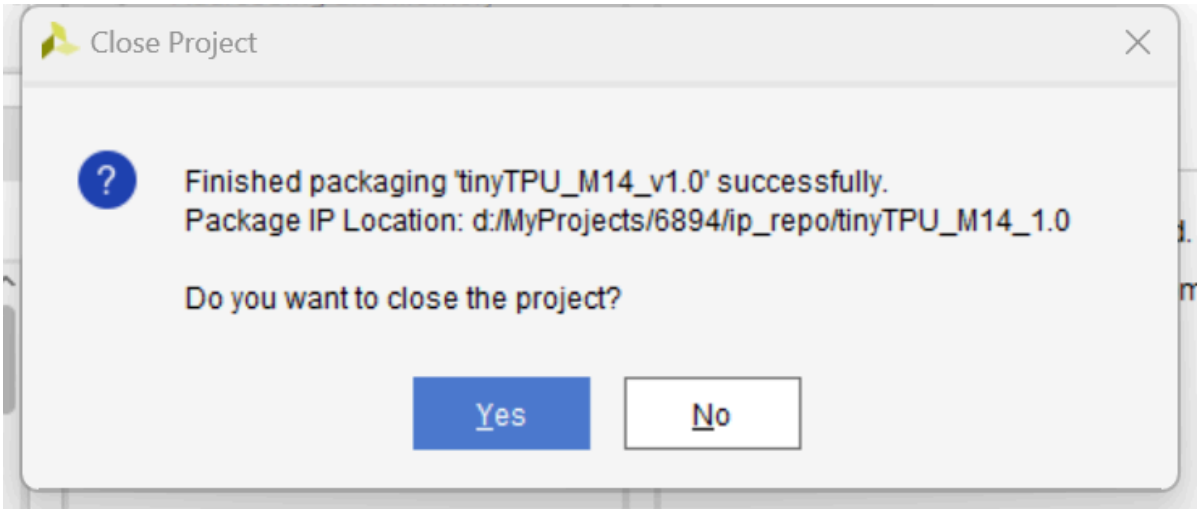
There will be 5 warnings, no worries.



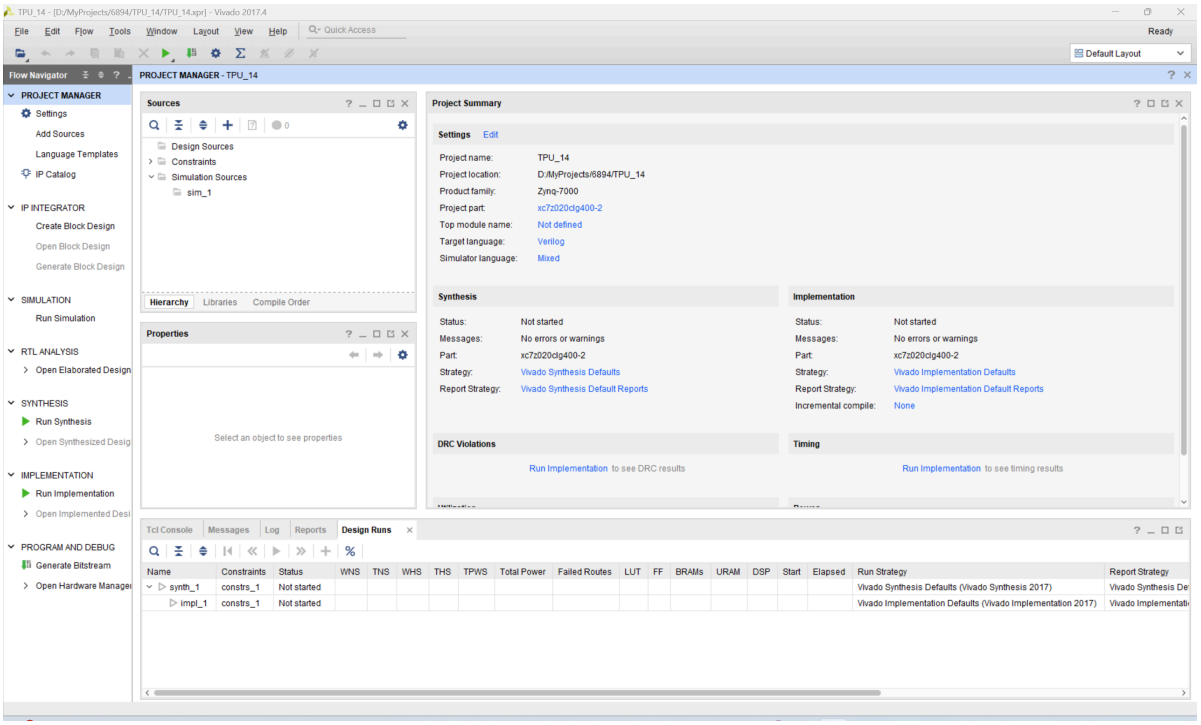
Merge Parameters and the rest will be merged automaticly.



Click Re-Package IP.

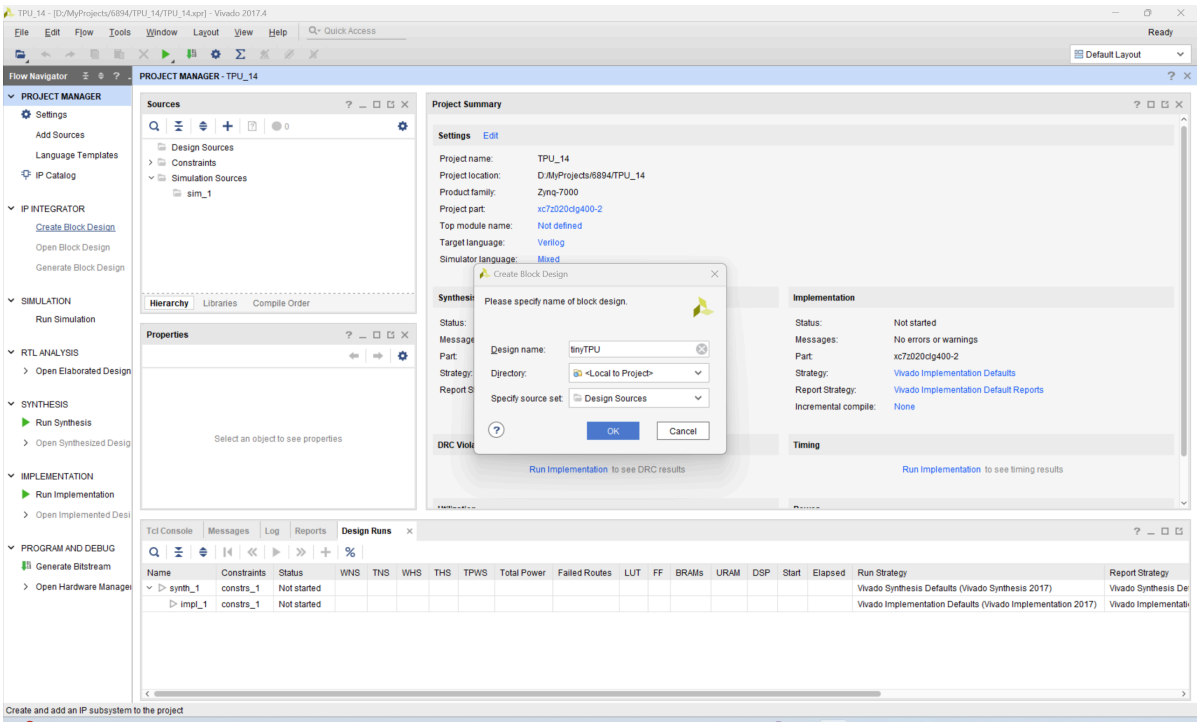


Just click Yes.

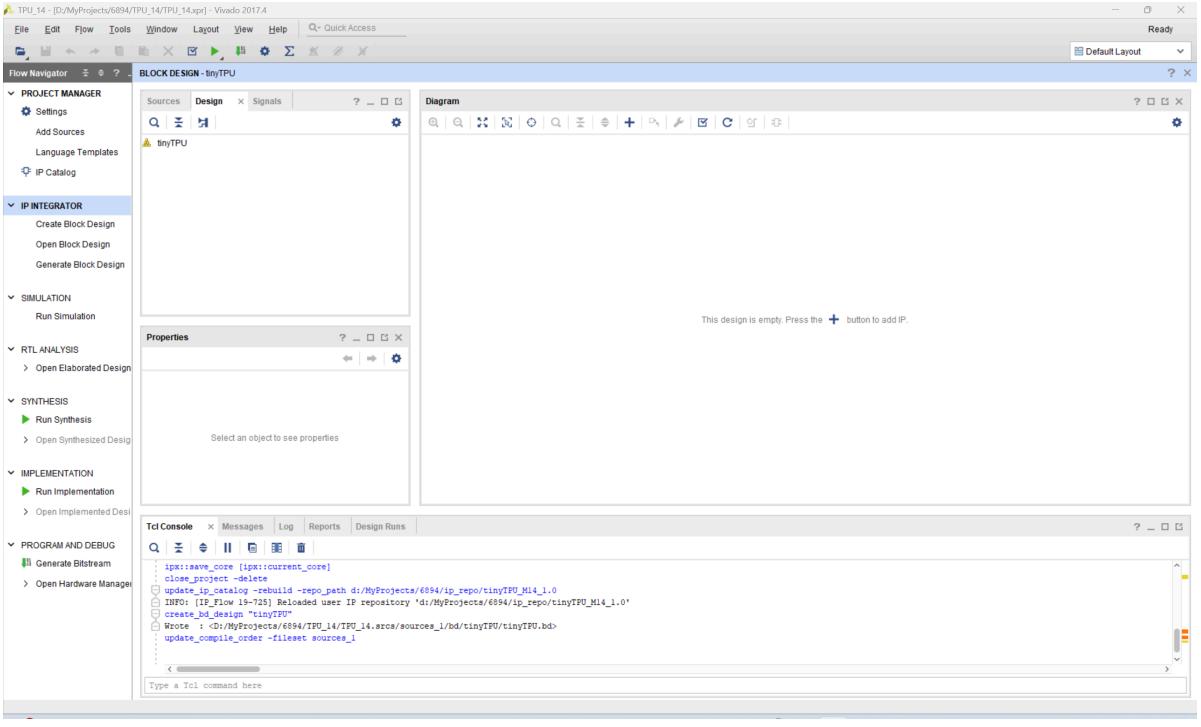


Then you will come back to empty project you created as TPU_14.

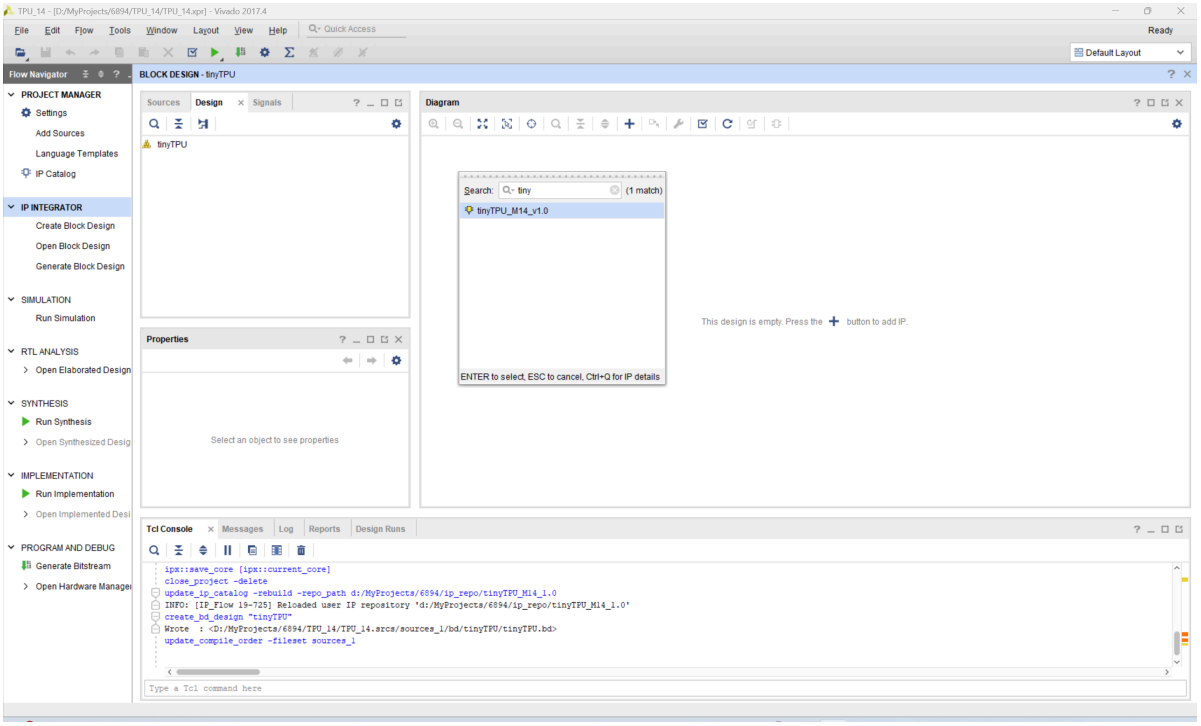
Create Block Design



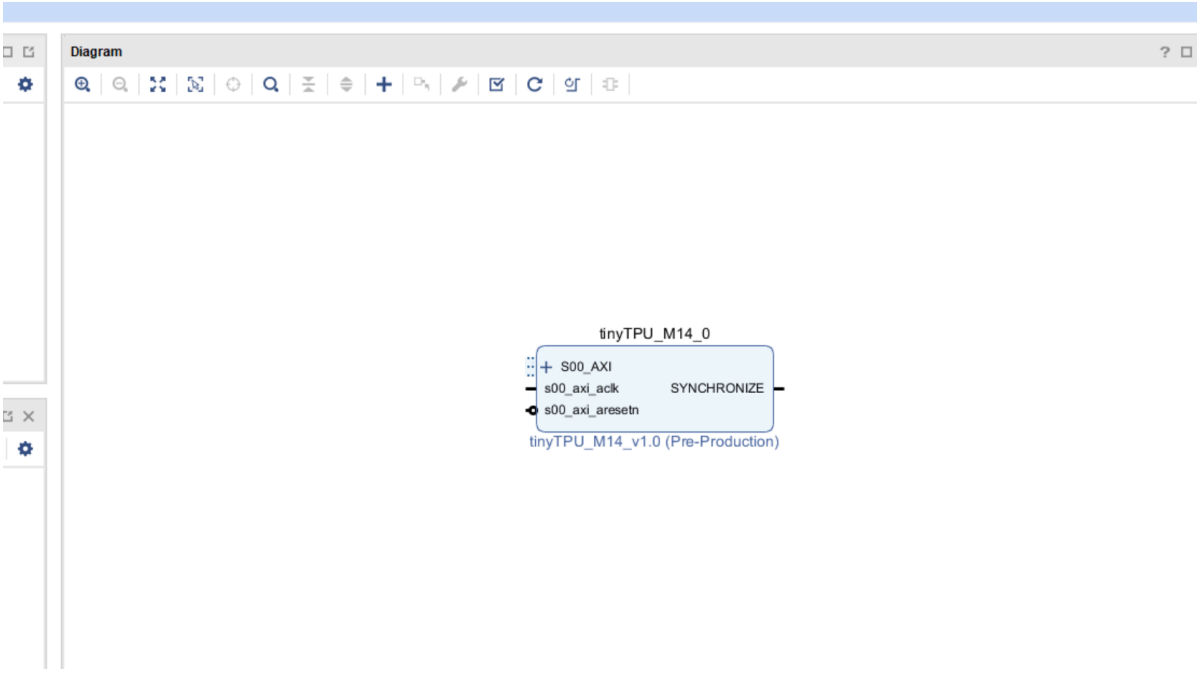
Press left list Create Block Design, then there will be a window to rename your Design name as tinyPTU.



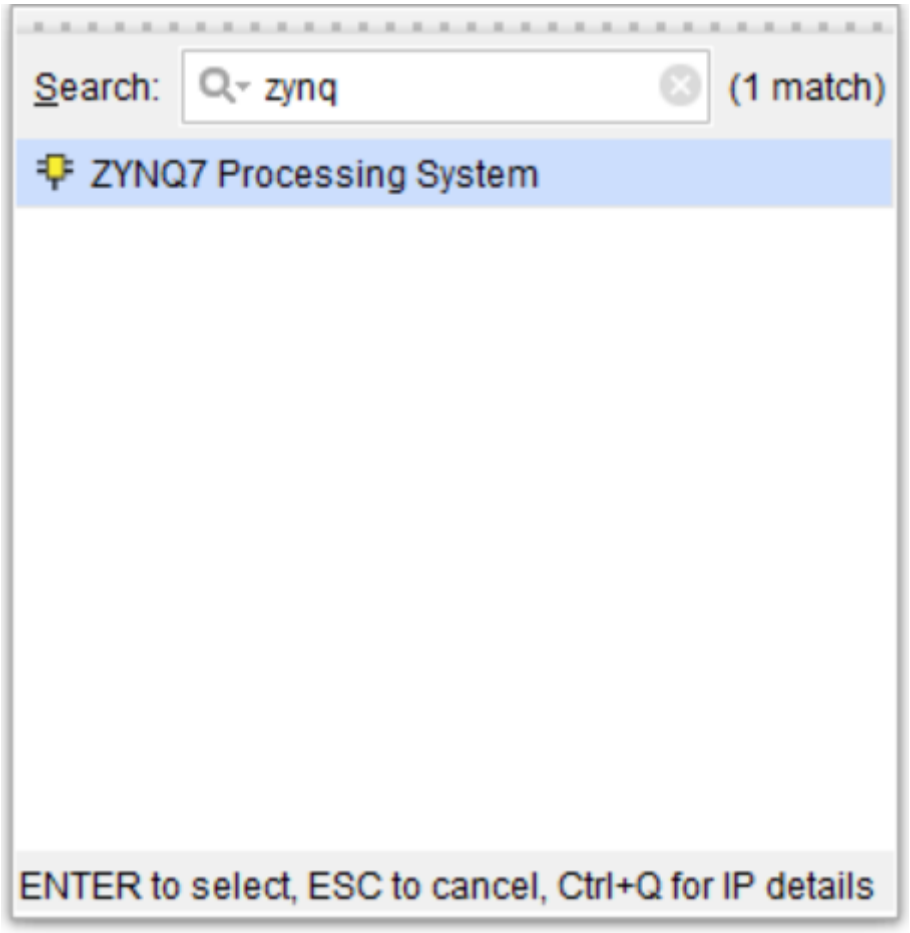
There will be an empty block.



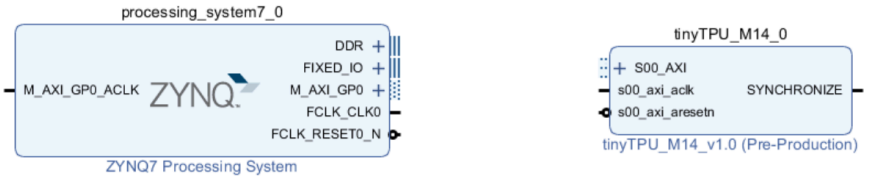
Click "+" to create IP block of tinyTPU. Search tinyTPU_M14_v1.0 and click the IP.



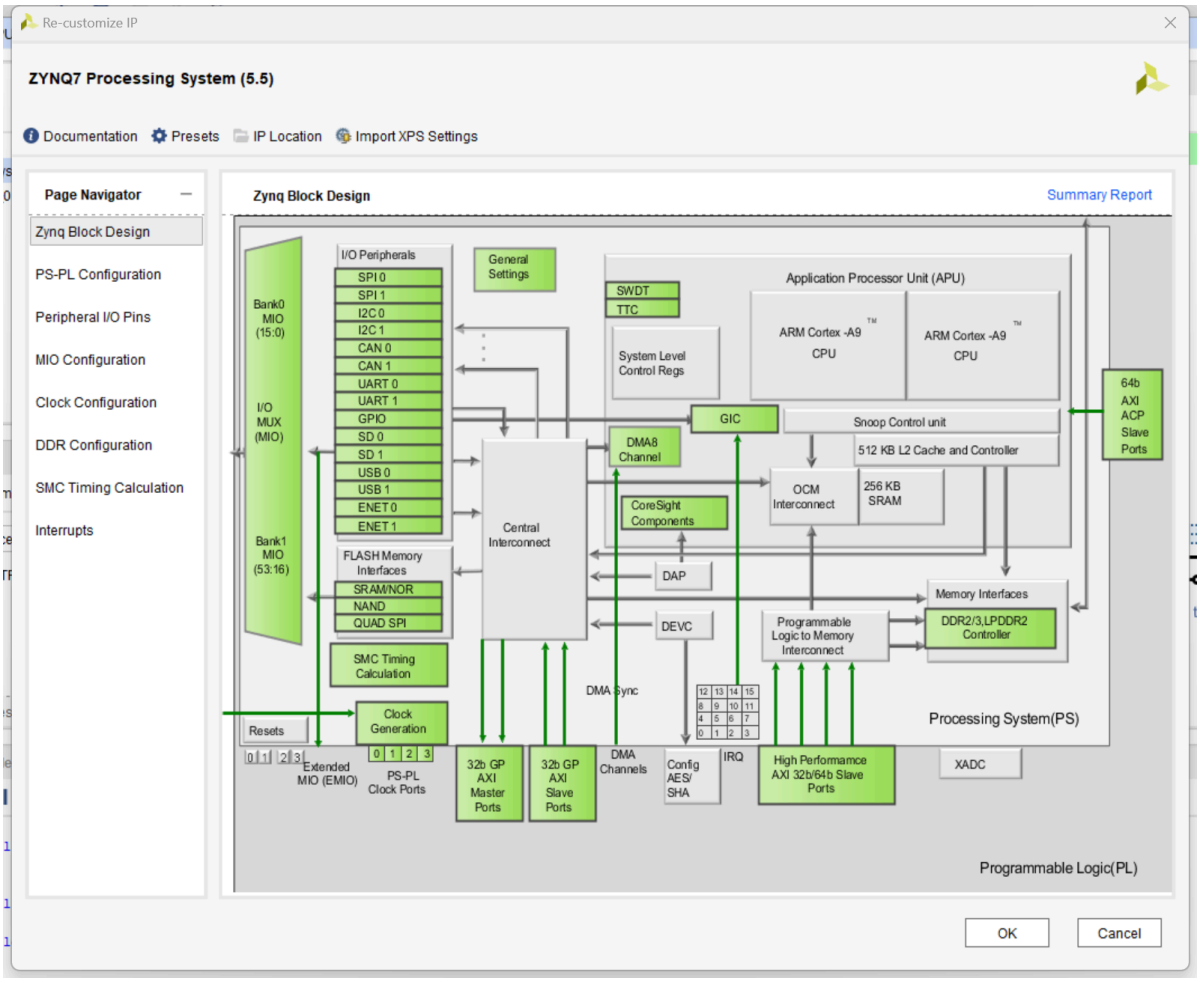
Here it will be placed on the design.



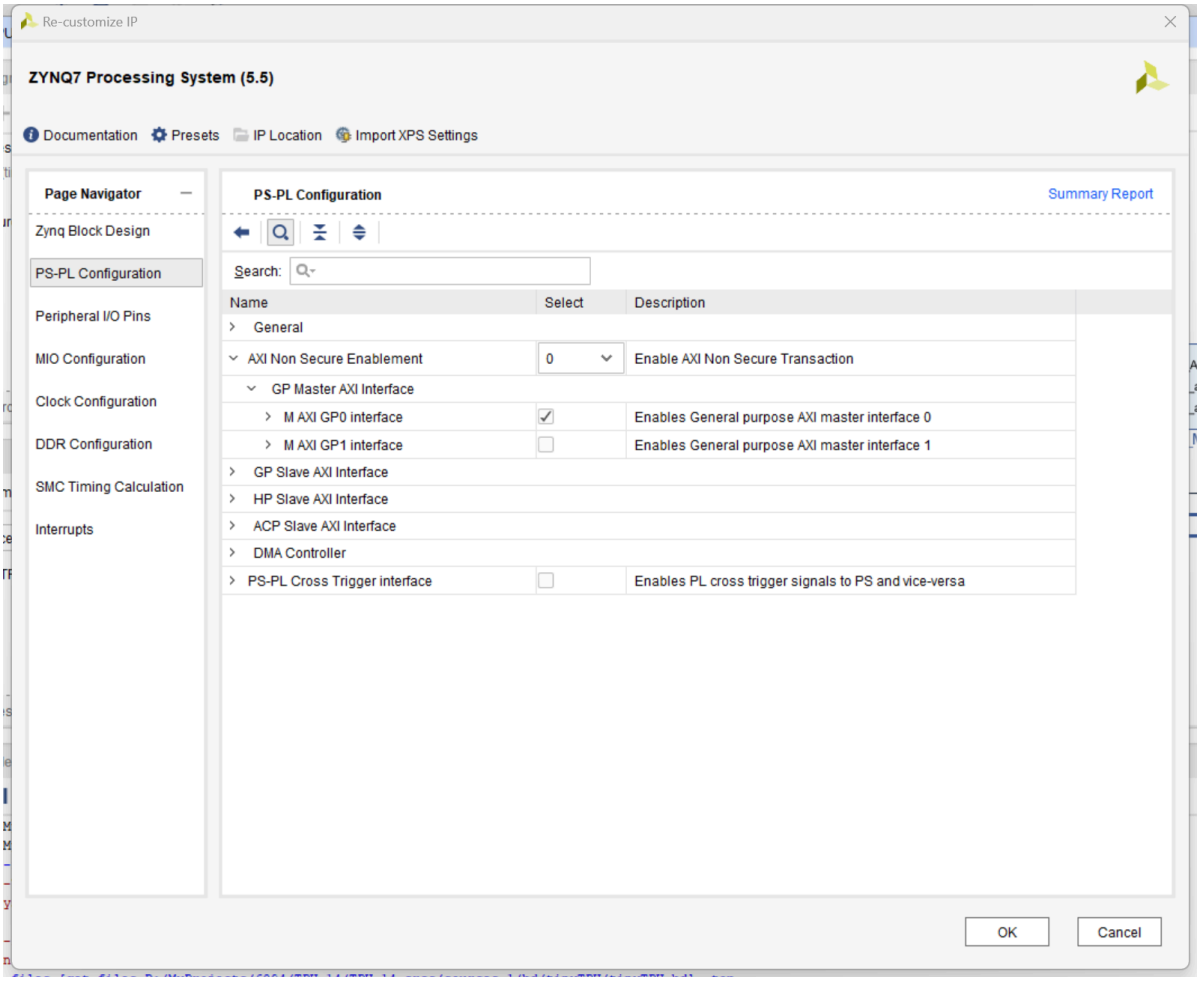
Similarly, search ZYNQ to find the process core.



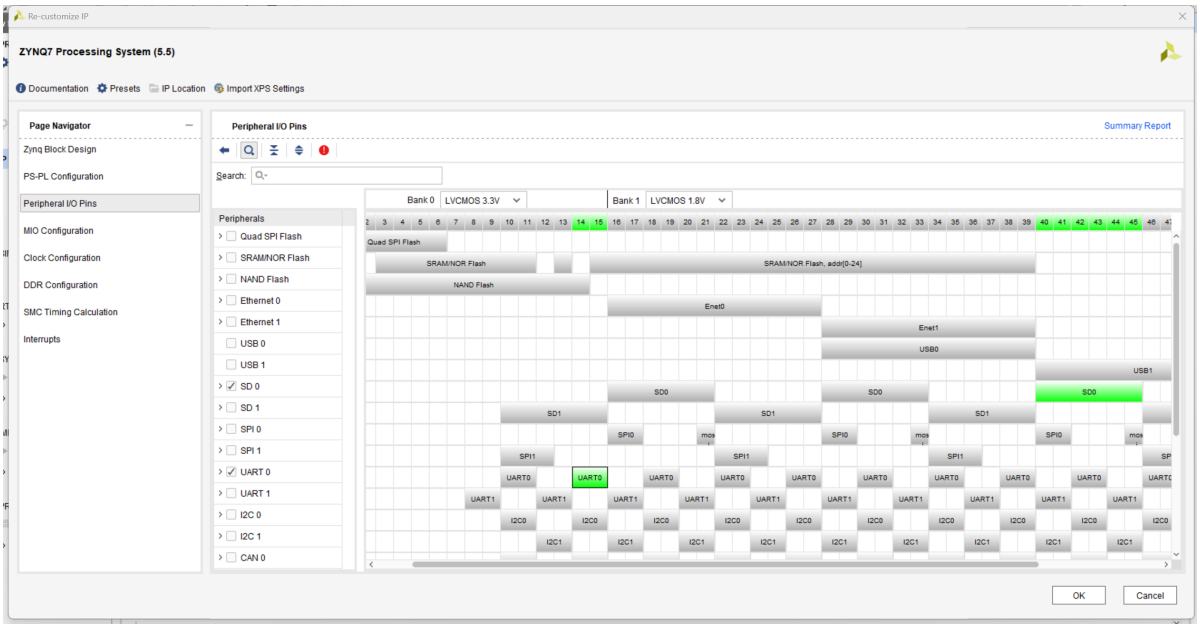
Place it here and double click the ZYNQ.



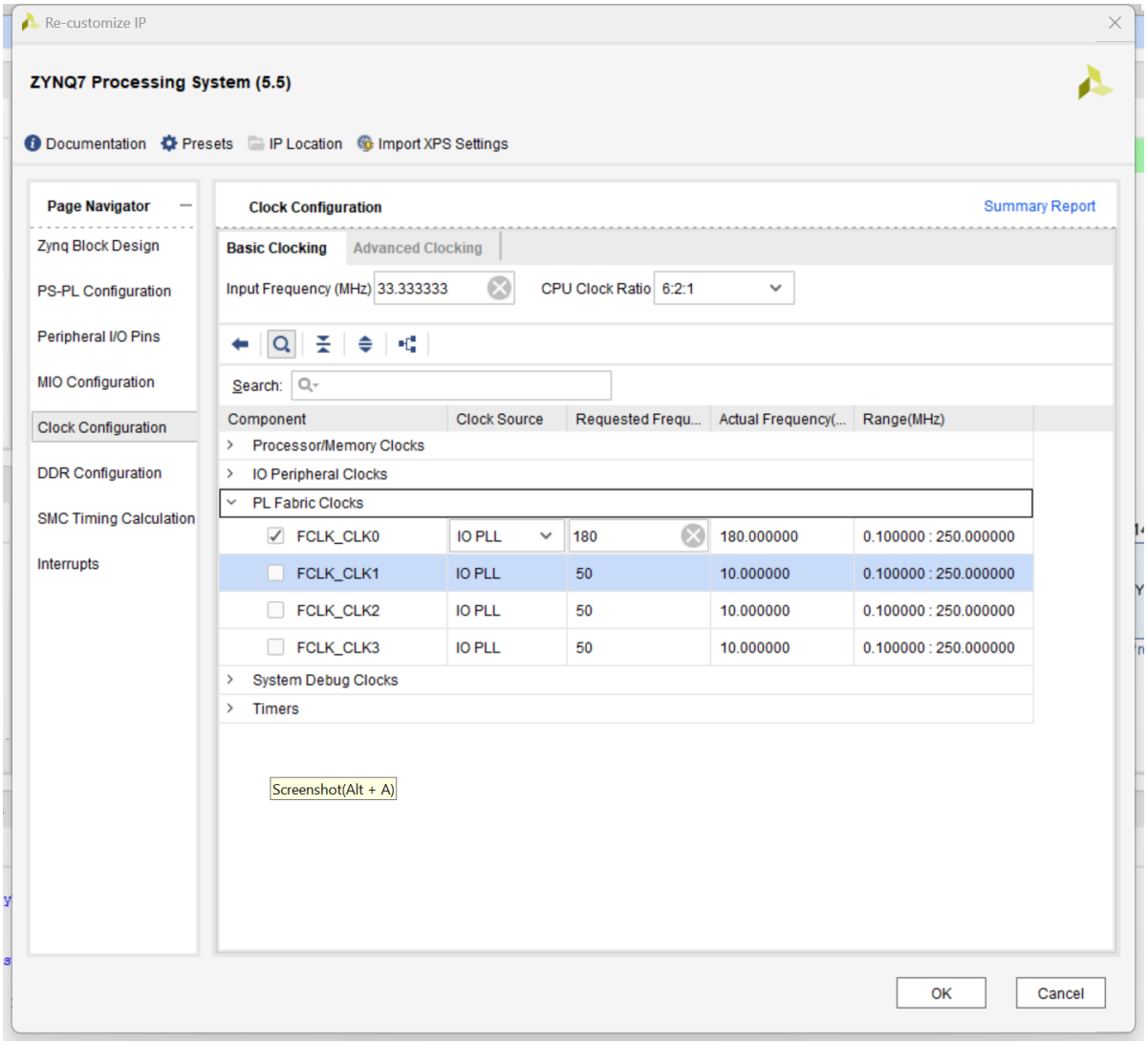
This window will be opened.



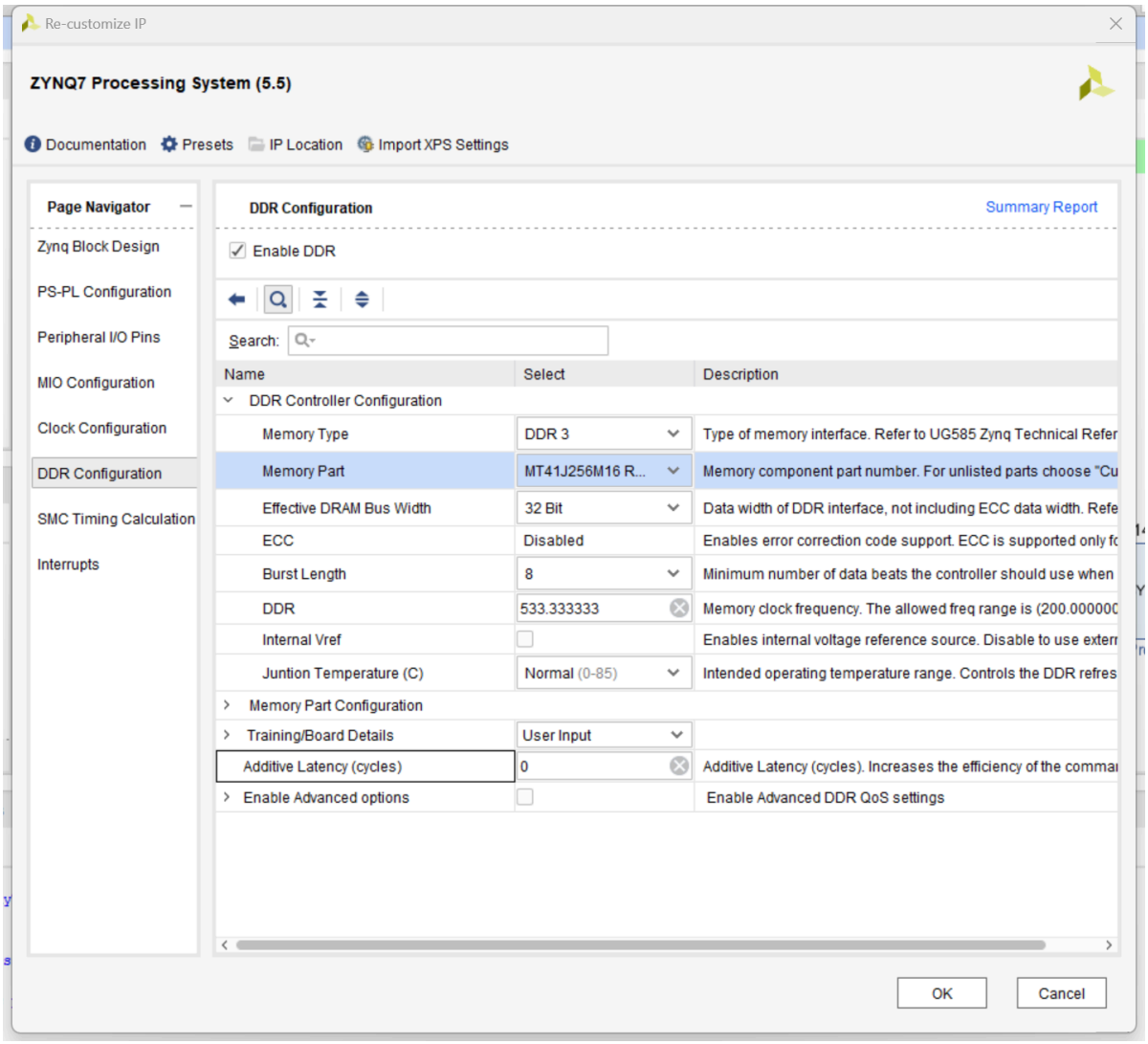
Enable M AXI GP0 interface.



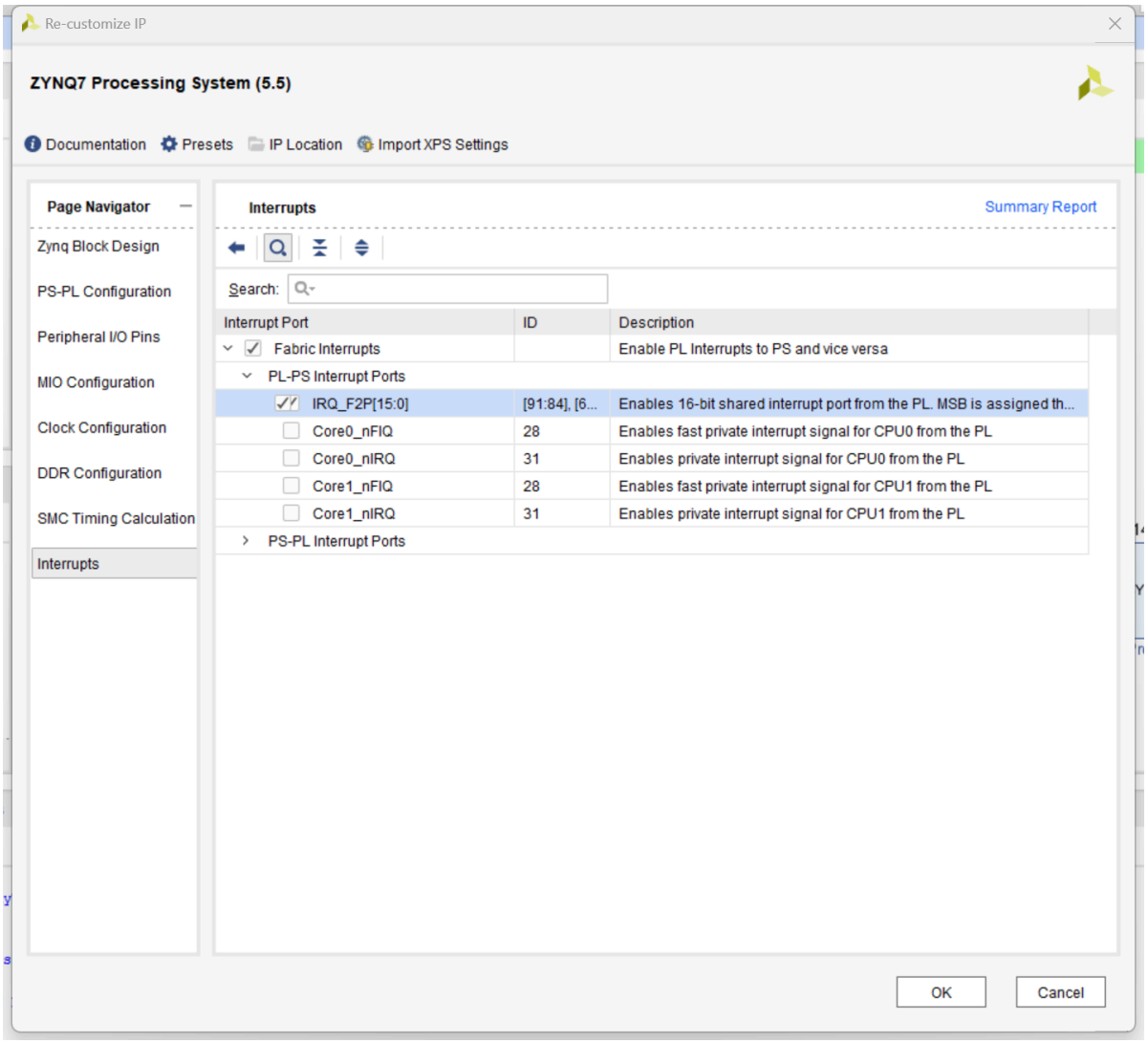
Enable SD0 with pin40-pin45 and UART0 with pin14 pin15. Select Bank1 LVCMOS1.8V.



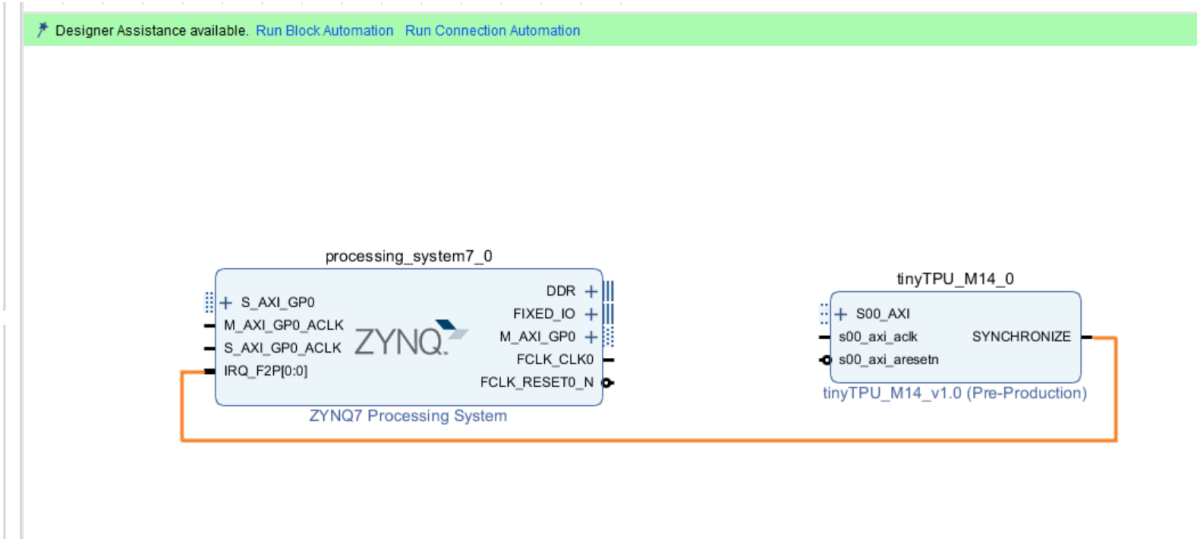
Choose FCLK_CLK0 IO PLL with 180MHz.



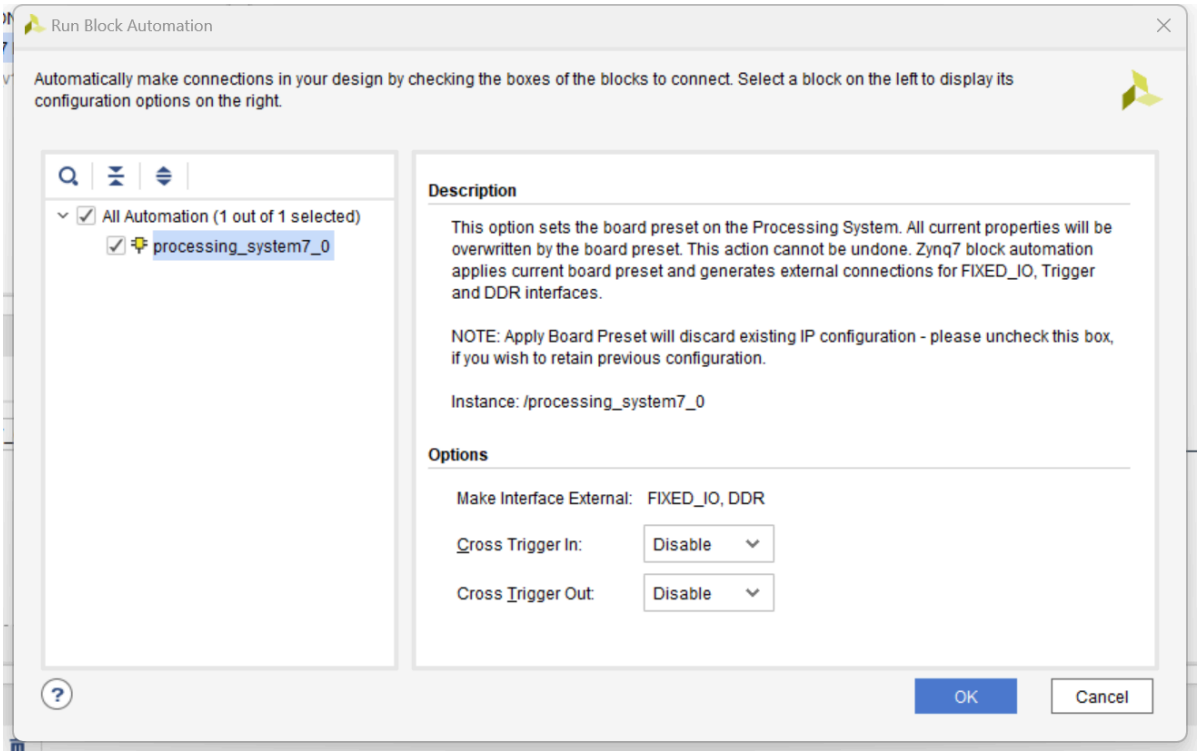
Select Memory Part as MT41J256M16 RE-125.



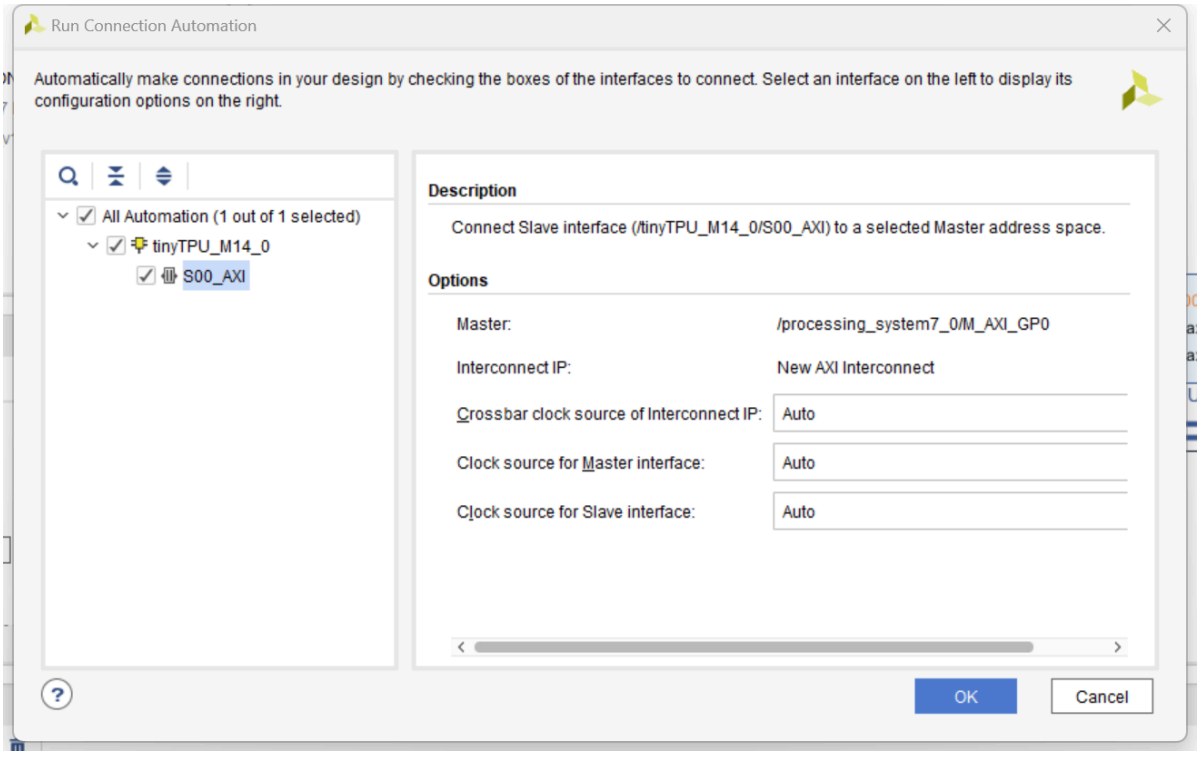
Enable interrupts.



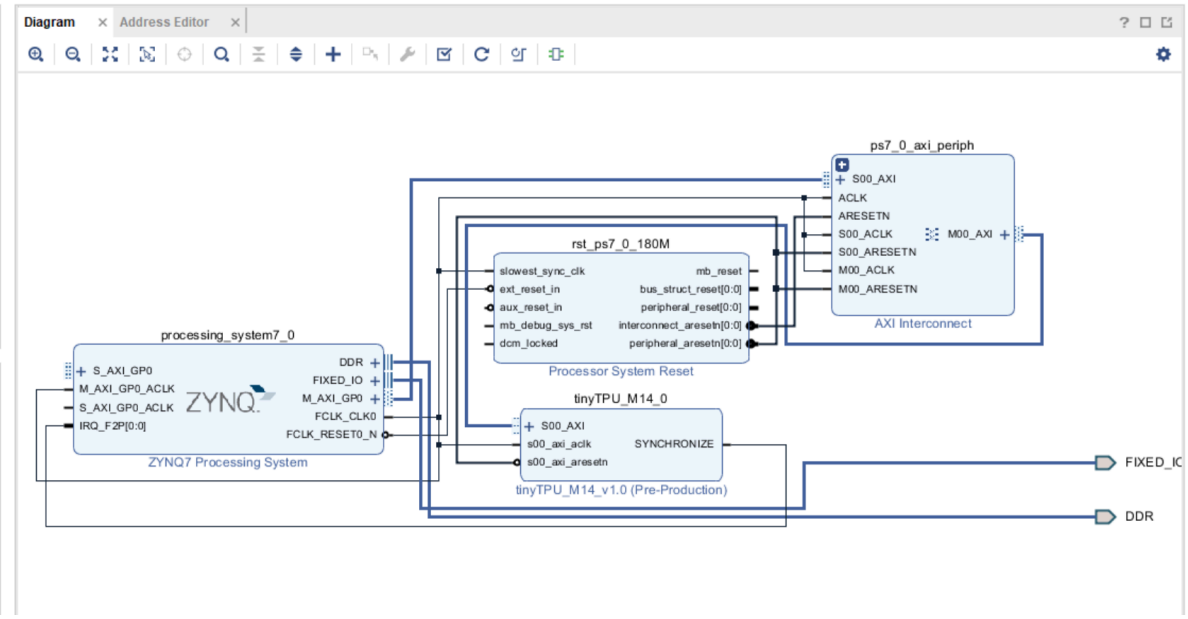
Connect IRQ with SYNCHRONIZE.



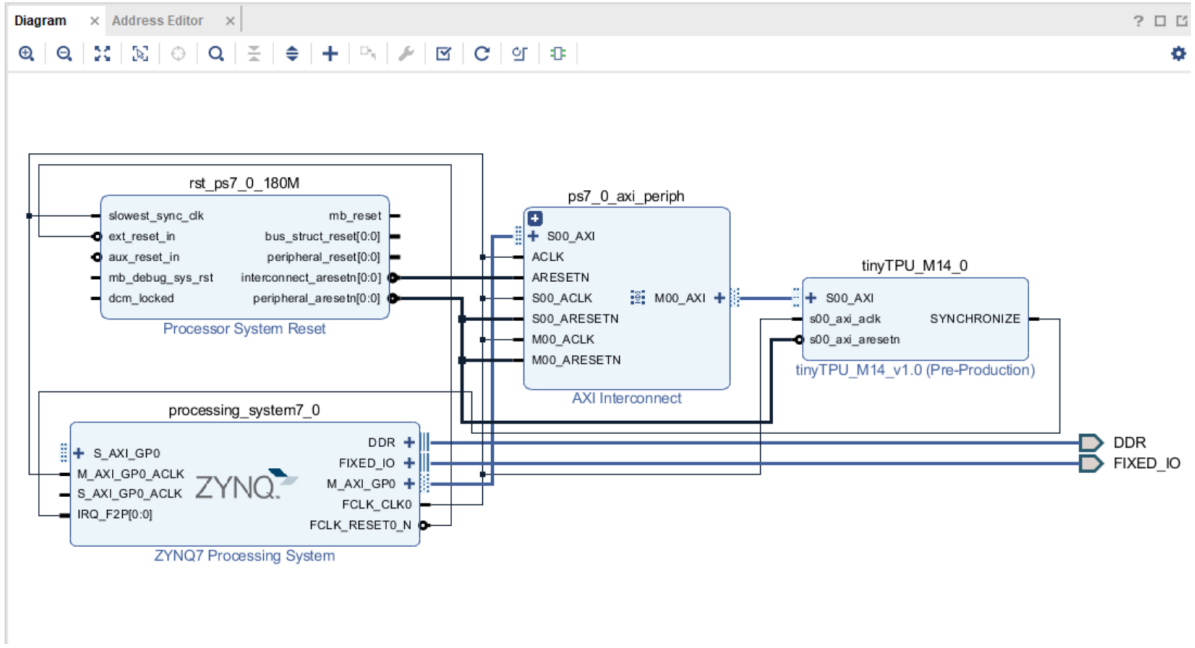
Click Run Block Automation and click OK.



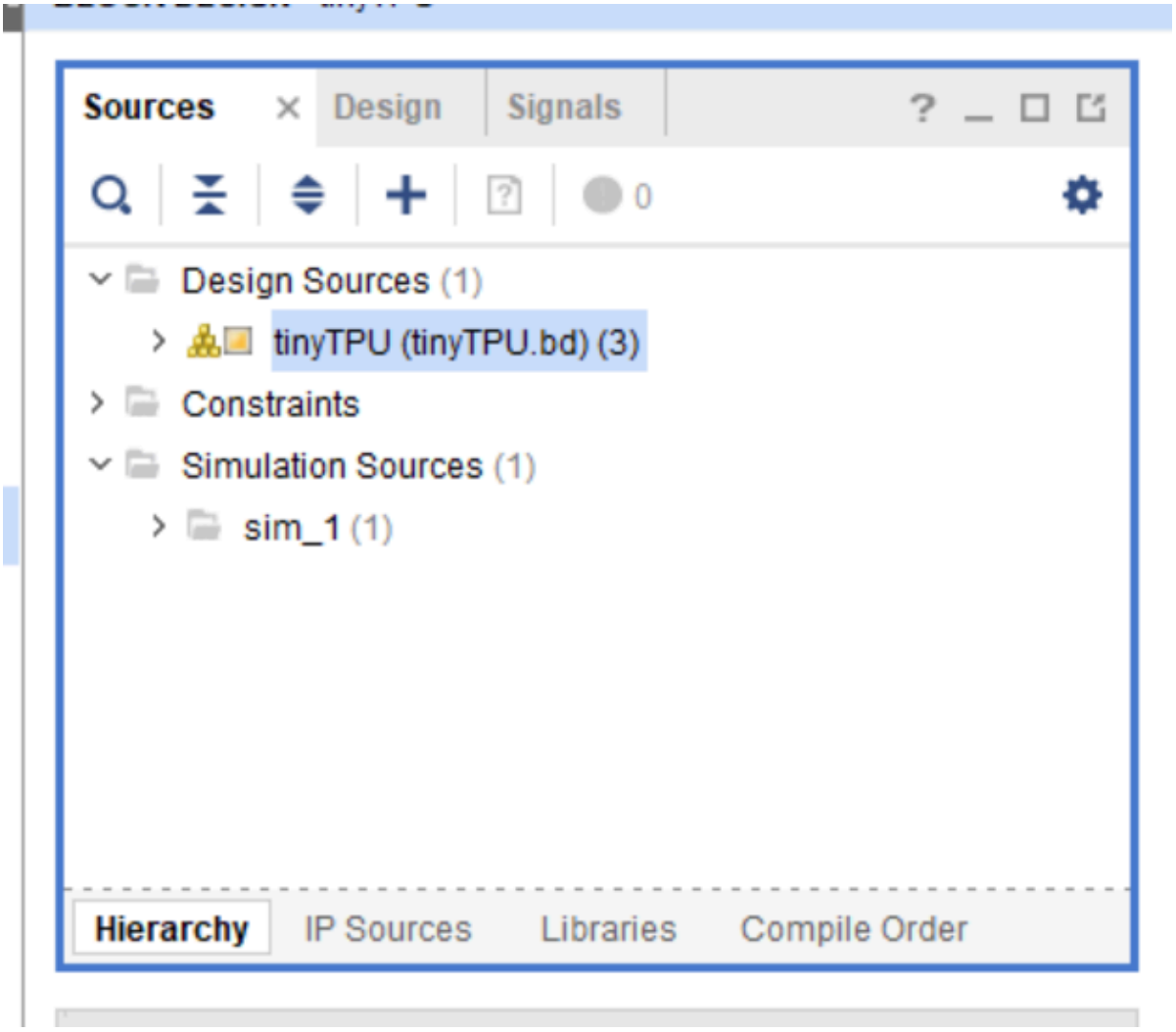
Click Run Connection Automation and click OK.



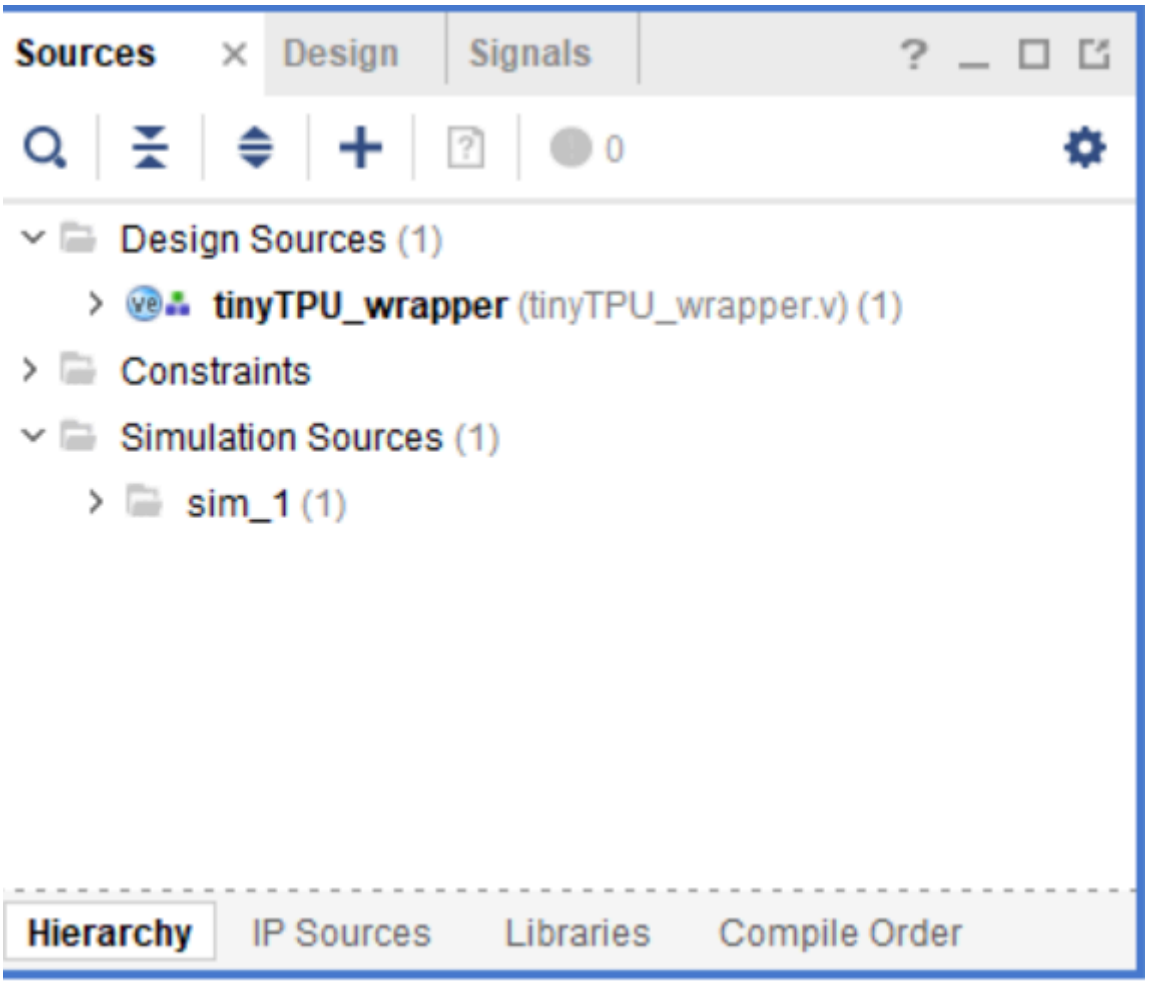
Here will be the Block Design.



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.

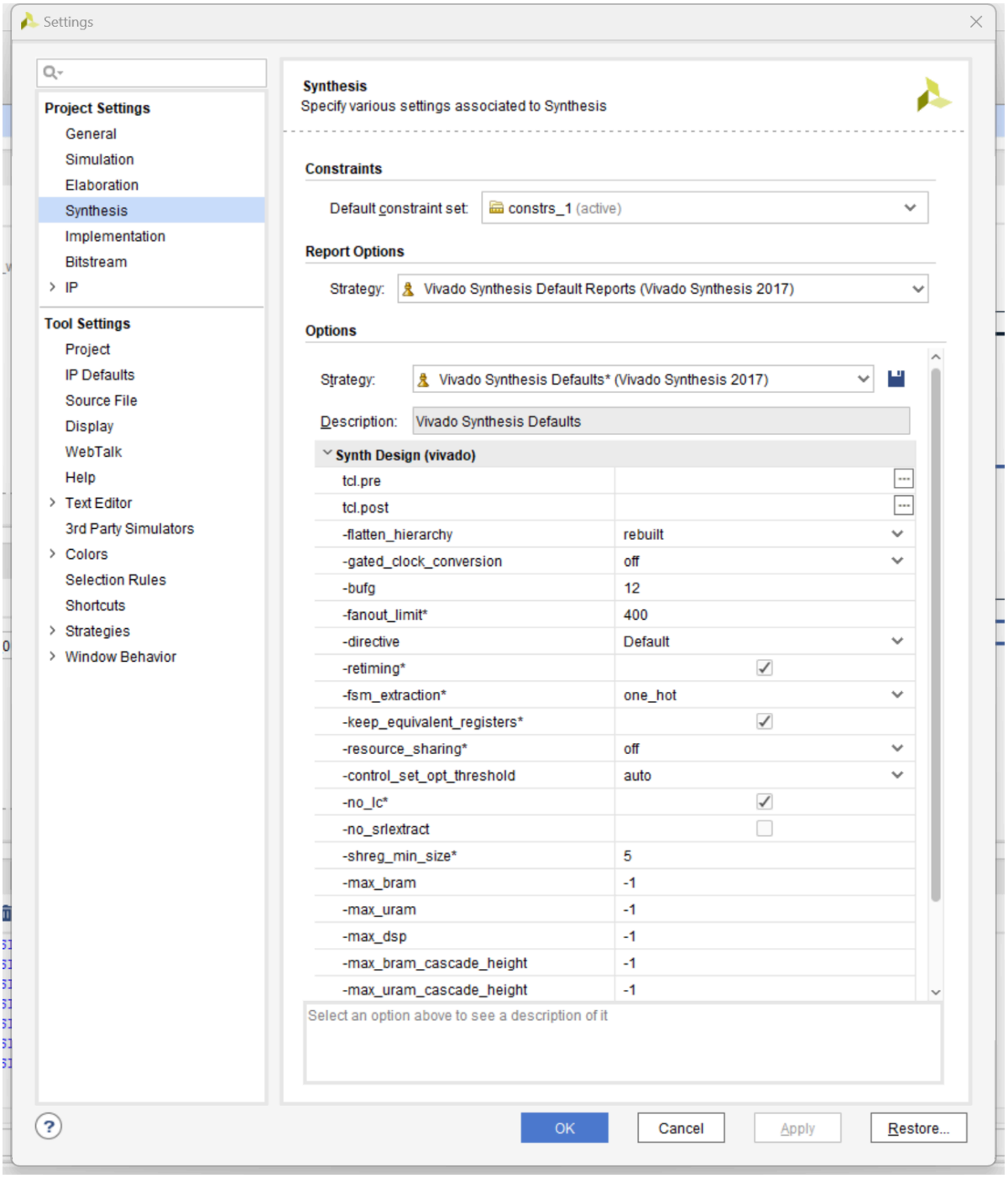


It will be like this.

Synthesis

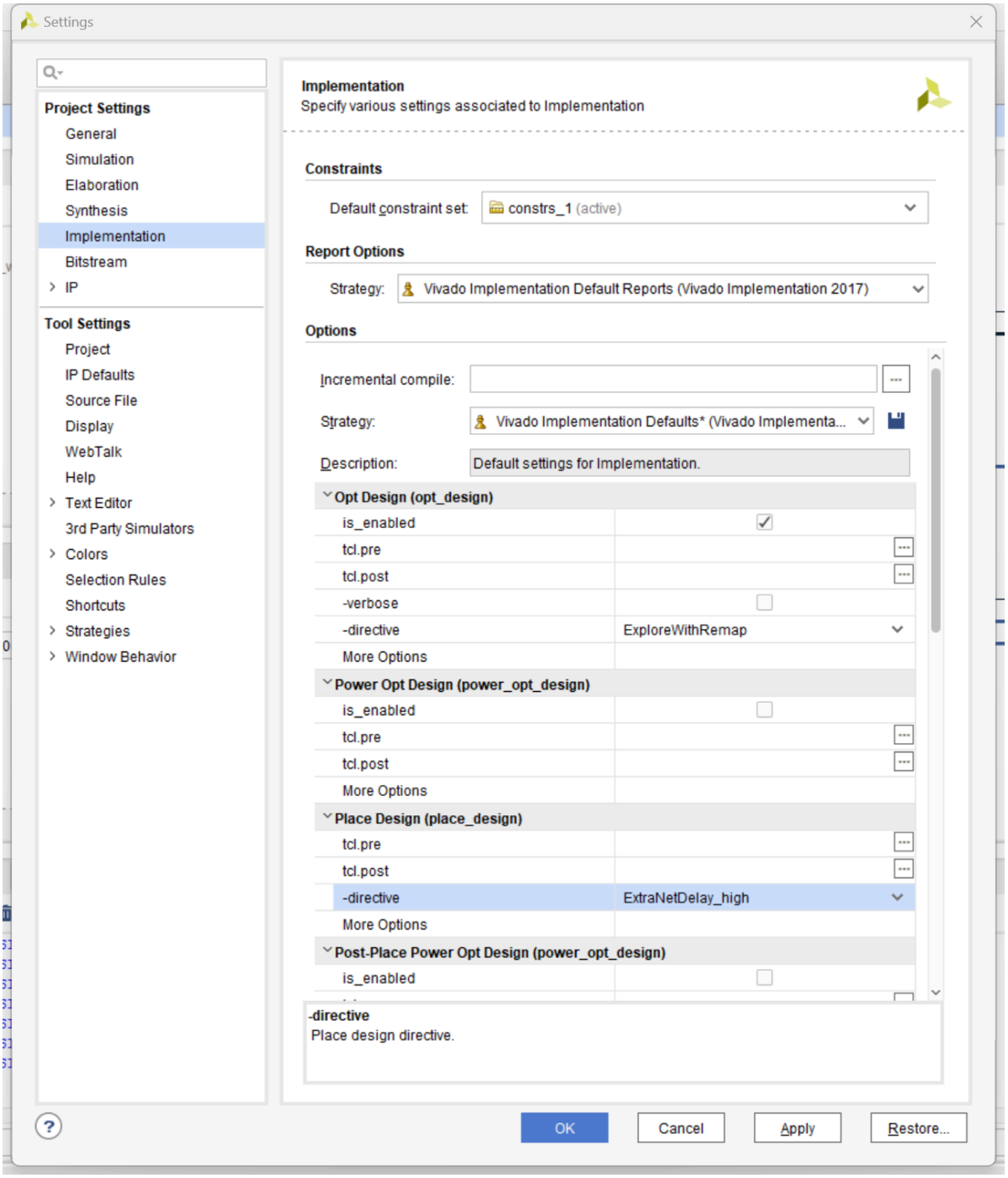
Click Tools -> Setting and click Synthesis. Change the settings shown below.

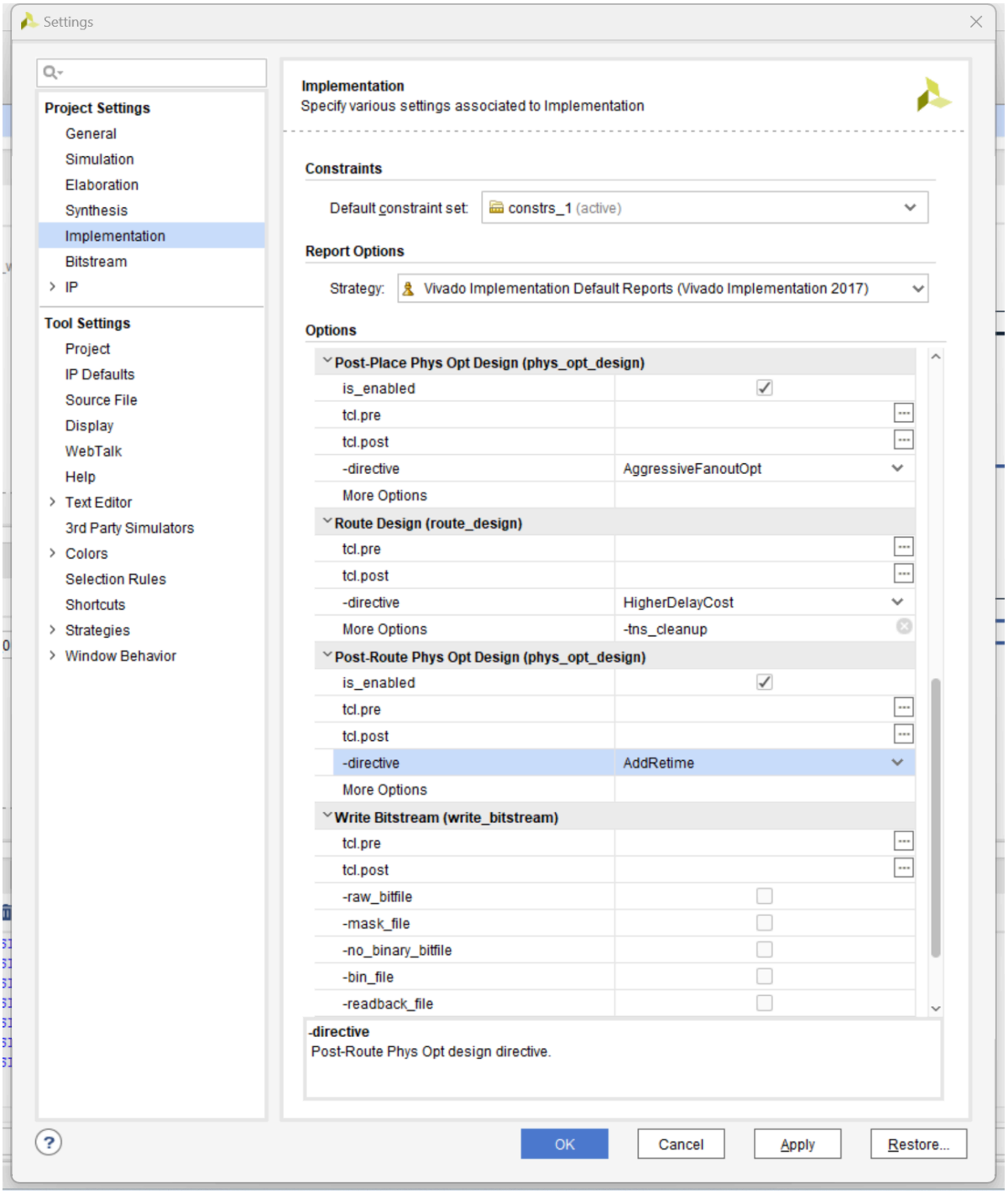
You can run synthesis now.



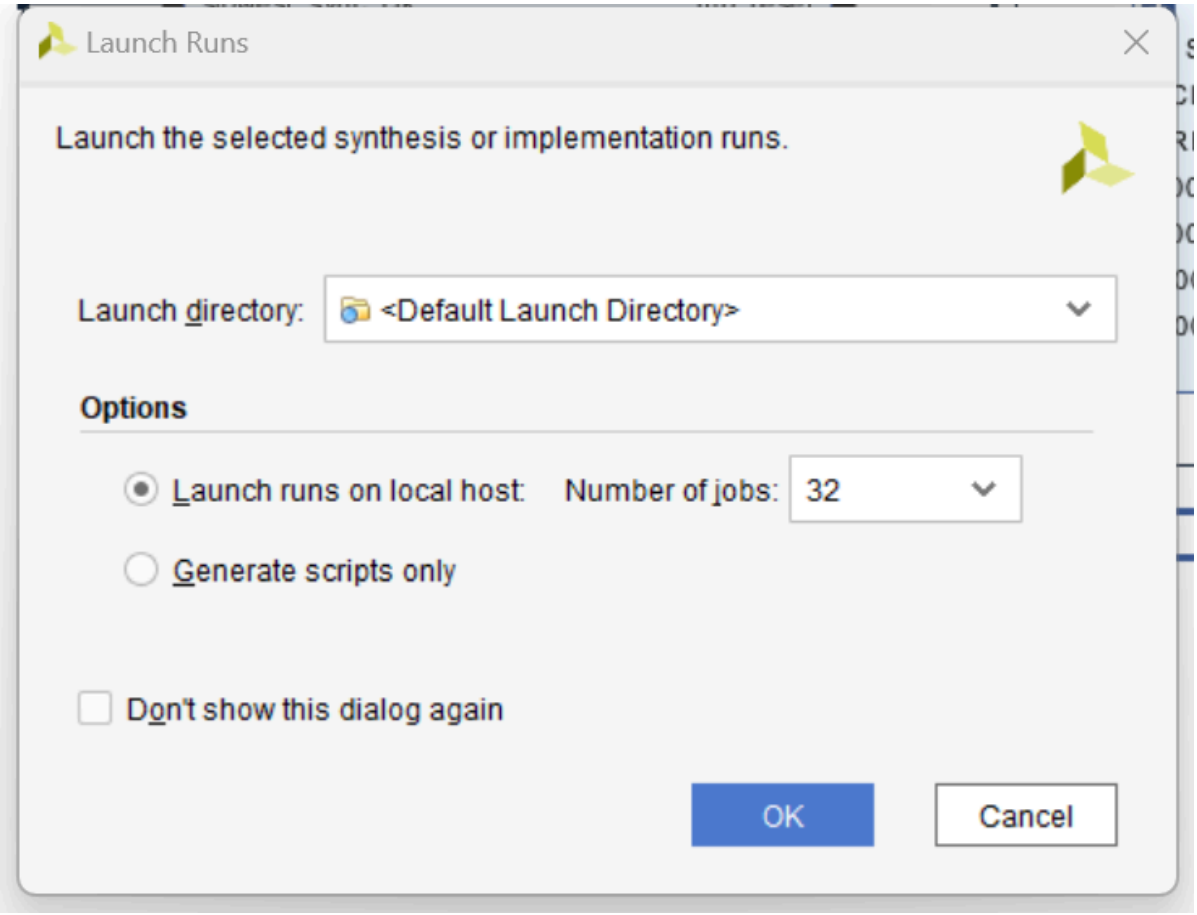
Implement

Similarly click Implementation and change the settings like below.

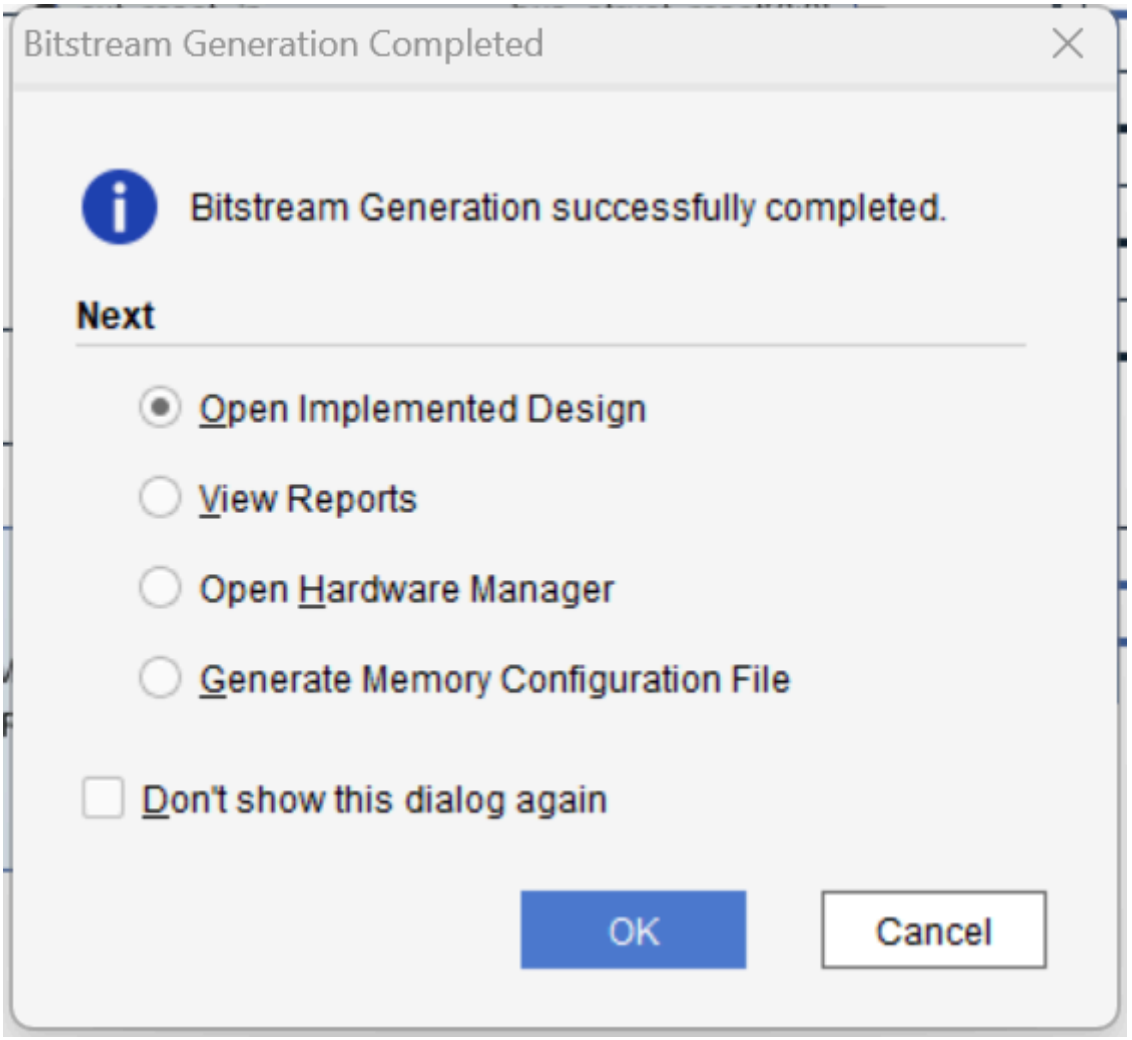

Settings
Project Settings
General
Simulation
Elaboration
Synthesis
Implementation
Bitstream
IP
Tool Settings
Project
IP Defaults
Source File
Display
WebTalk
Help
Text Editor
3rd Party Simulators
Colors
Selection Rules
Shortcuts
Strategies
Window Behavior
Implementation
Specify various settings associated 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
tcl.pre
tcl.post
-verbose
-directive
ExploreWithRemap
More Options
Power Opt Design (power_opt_design)
is_enabled
tcl.pre
tcl.post
More Options
Place Design (place_design)
tcl.pre
tcl.post
-directive
ExtraNetDelay_high
More Options
Post-Place Power Opt Design (power_opt_design)
is_enabled
-directive
Place design directive.
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.



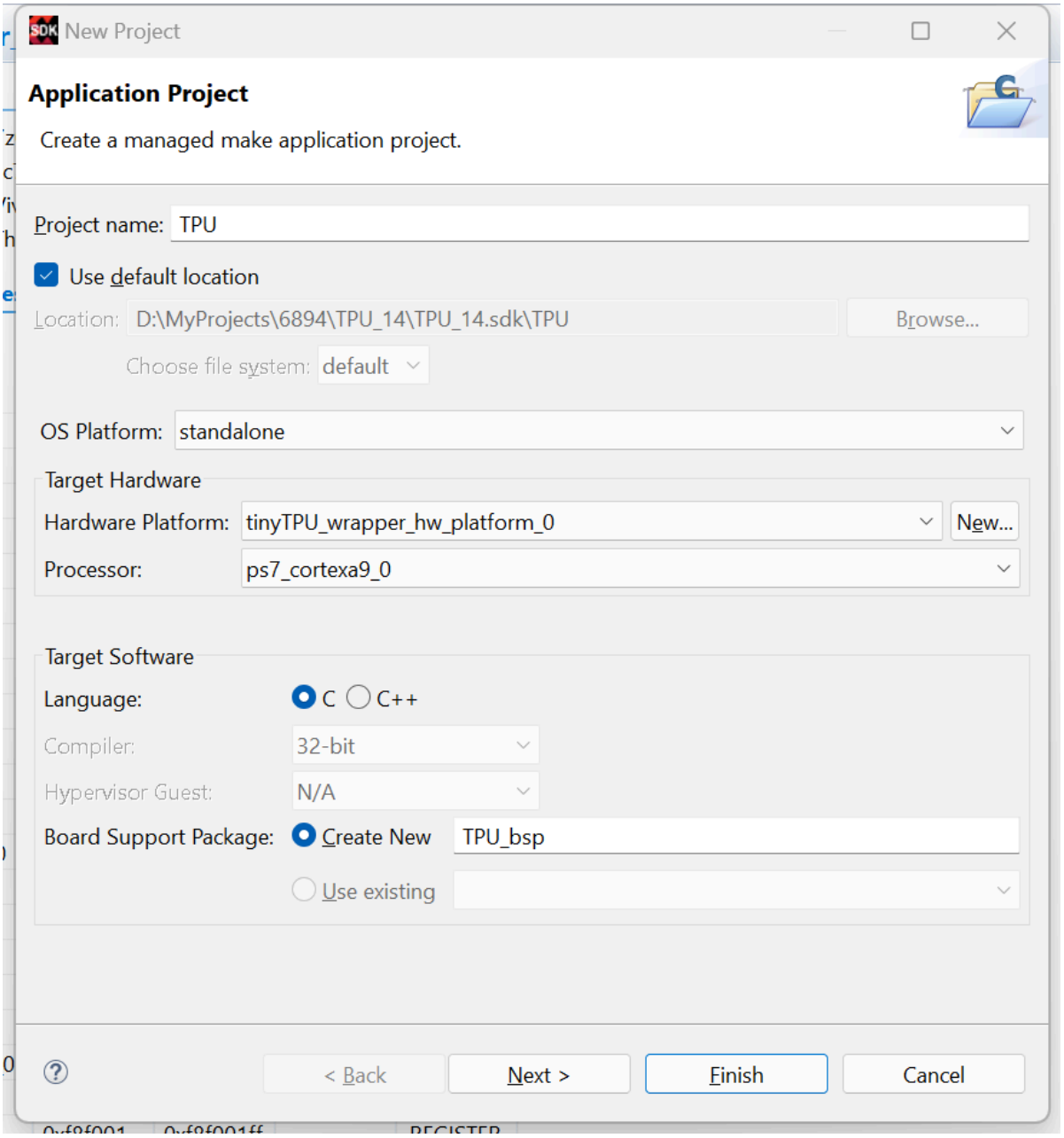
Here you can see you finished Bitstream Generation.

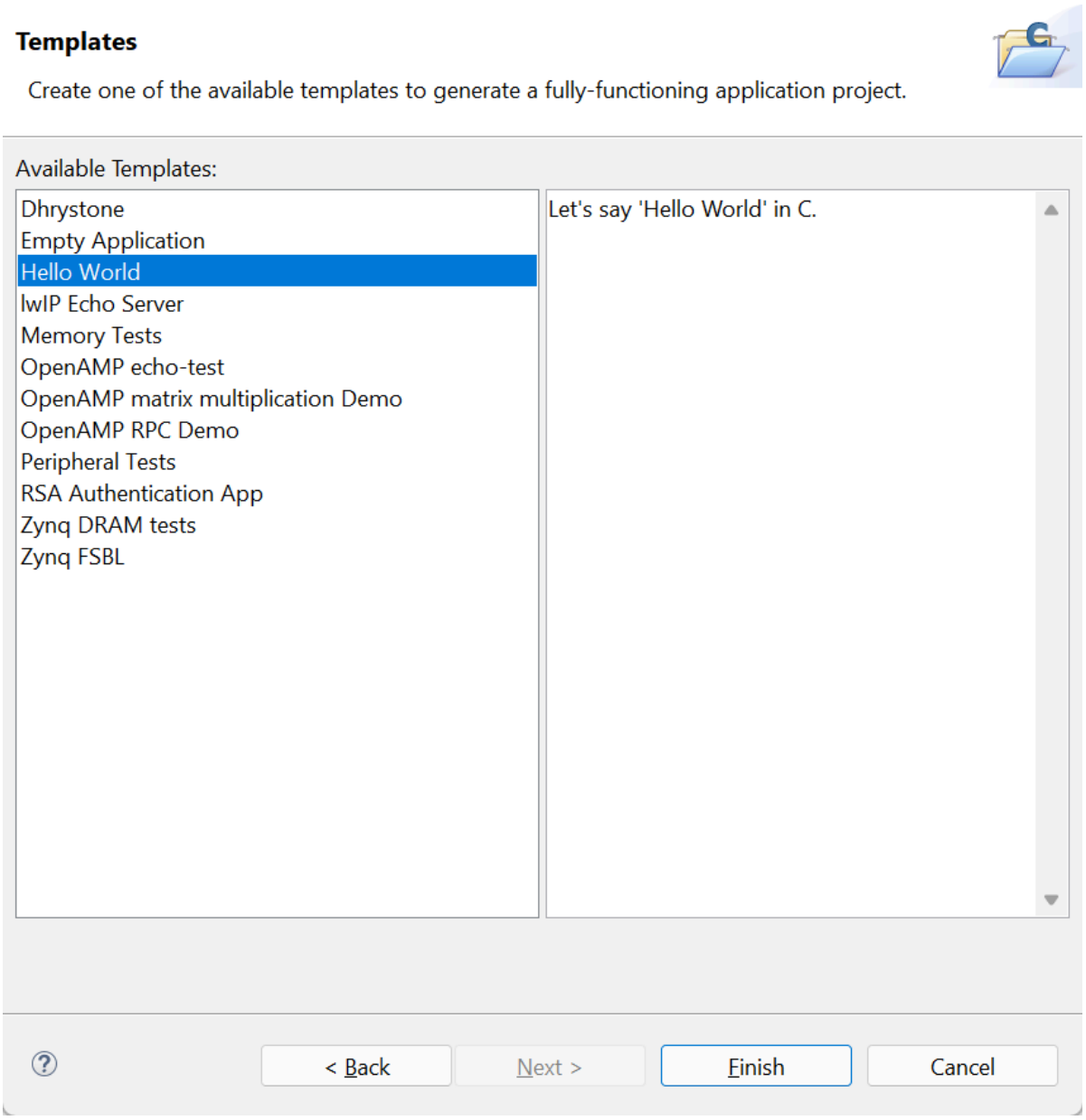
Congratulations, you have finished hardware design part.

Software Design

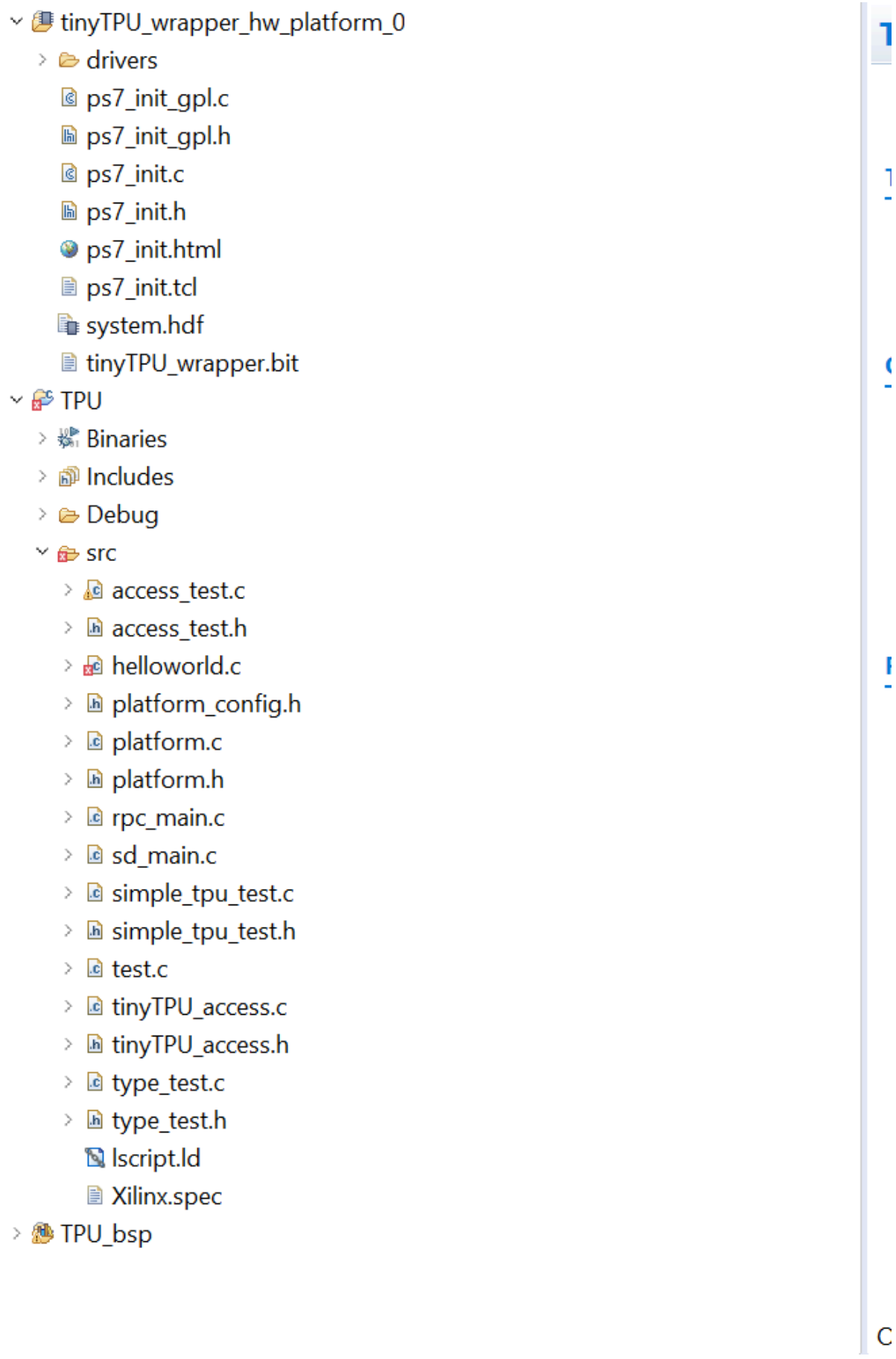
Add Sample Project in Xilinx SDK

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.





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

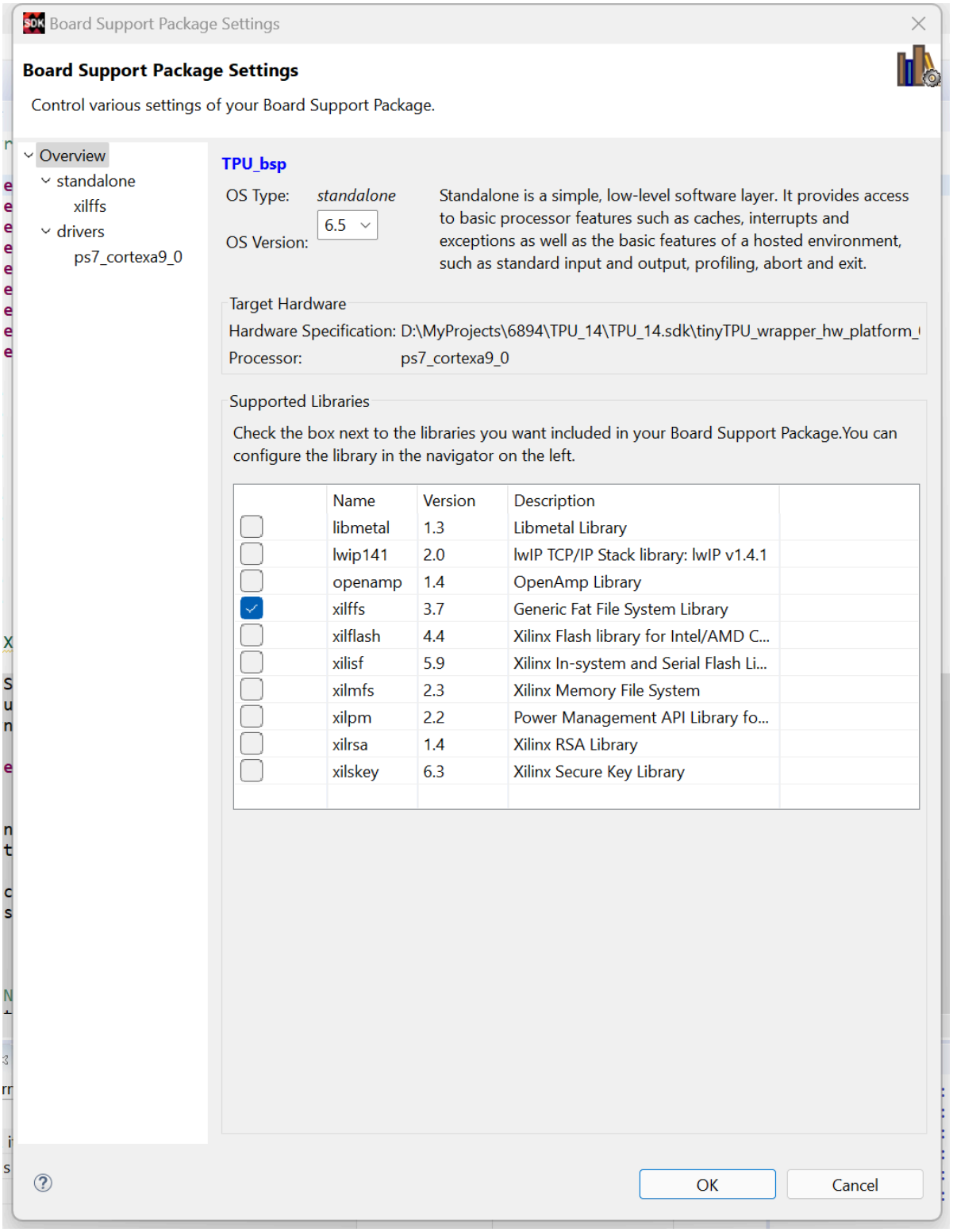


Add all .c and .h files in EB_C/src and overwrite exist files.

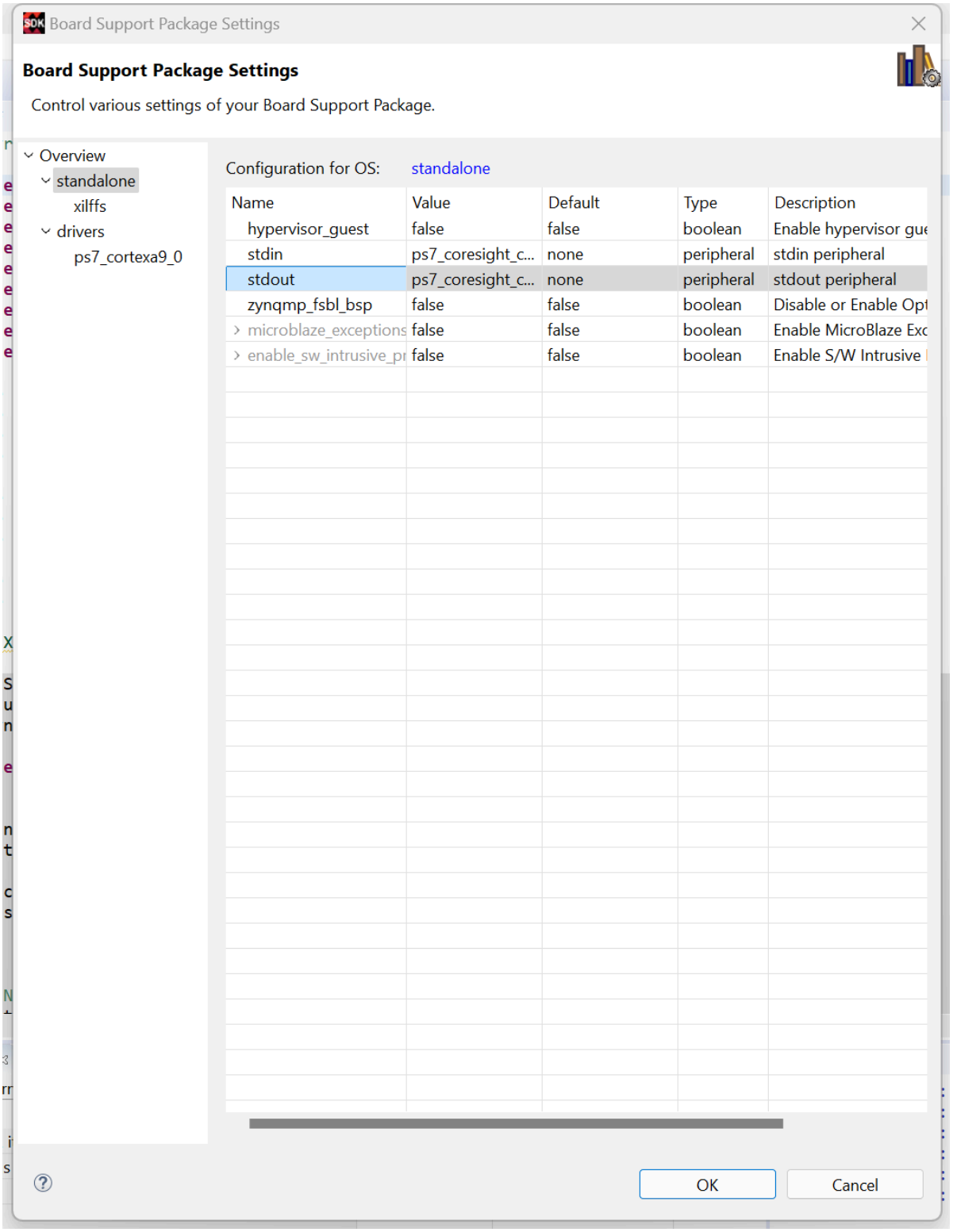
```
// Copyright 2018 Jonas Fuhrmann. All rights reserved.

#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;

#ifdef 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".

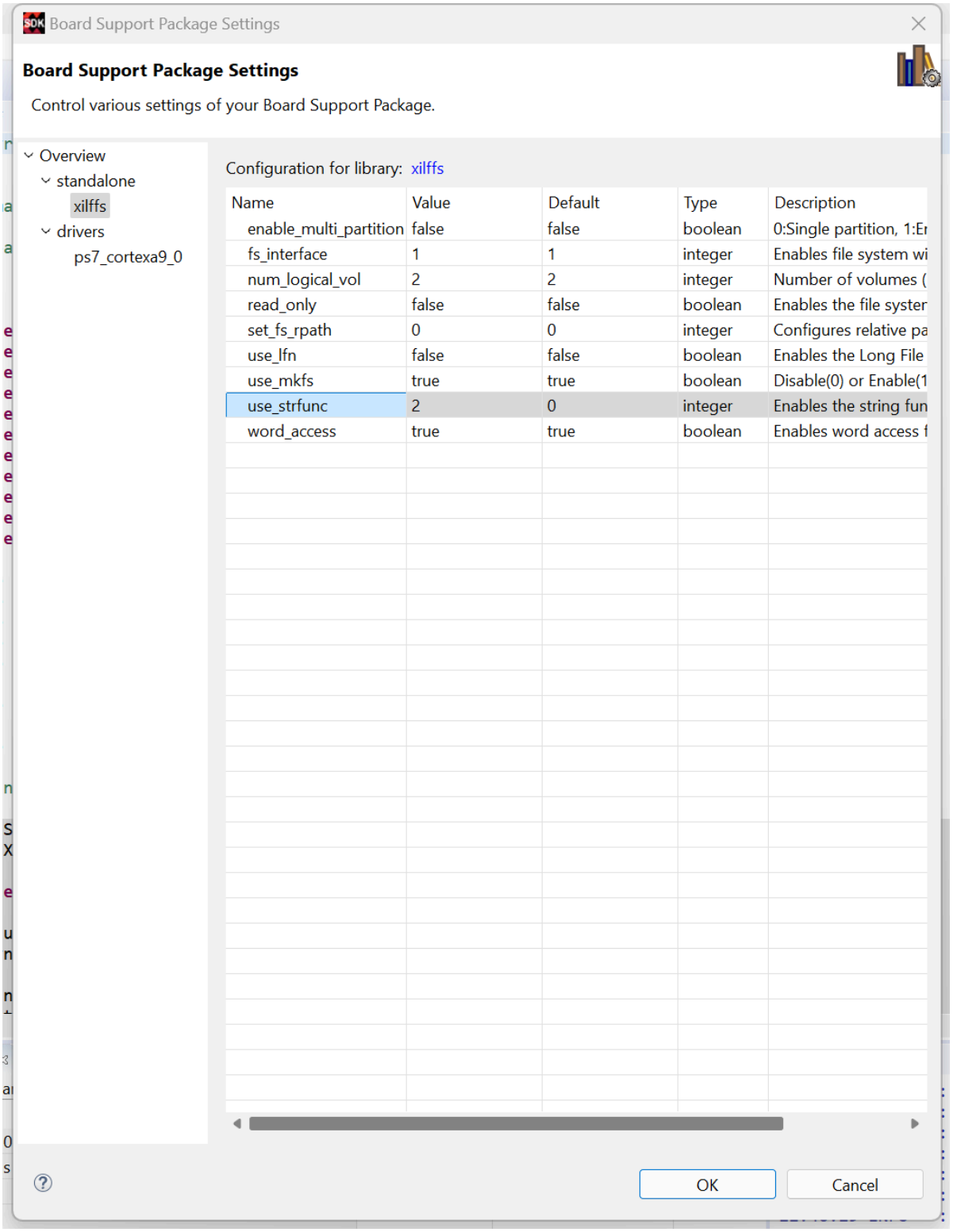
Right click BSP and select Board Support Package Settings and check the xilffs library.



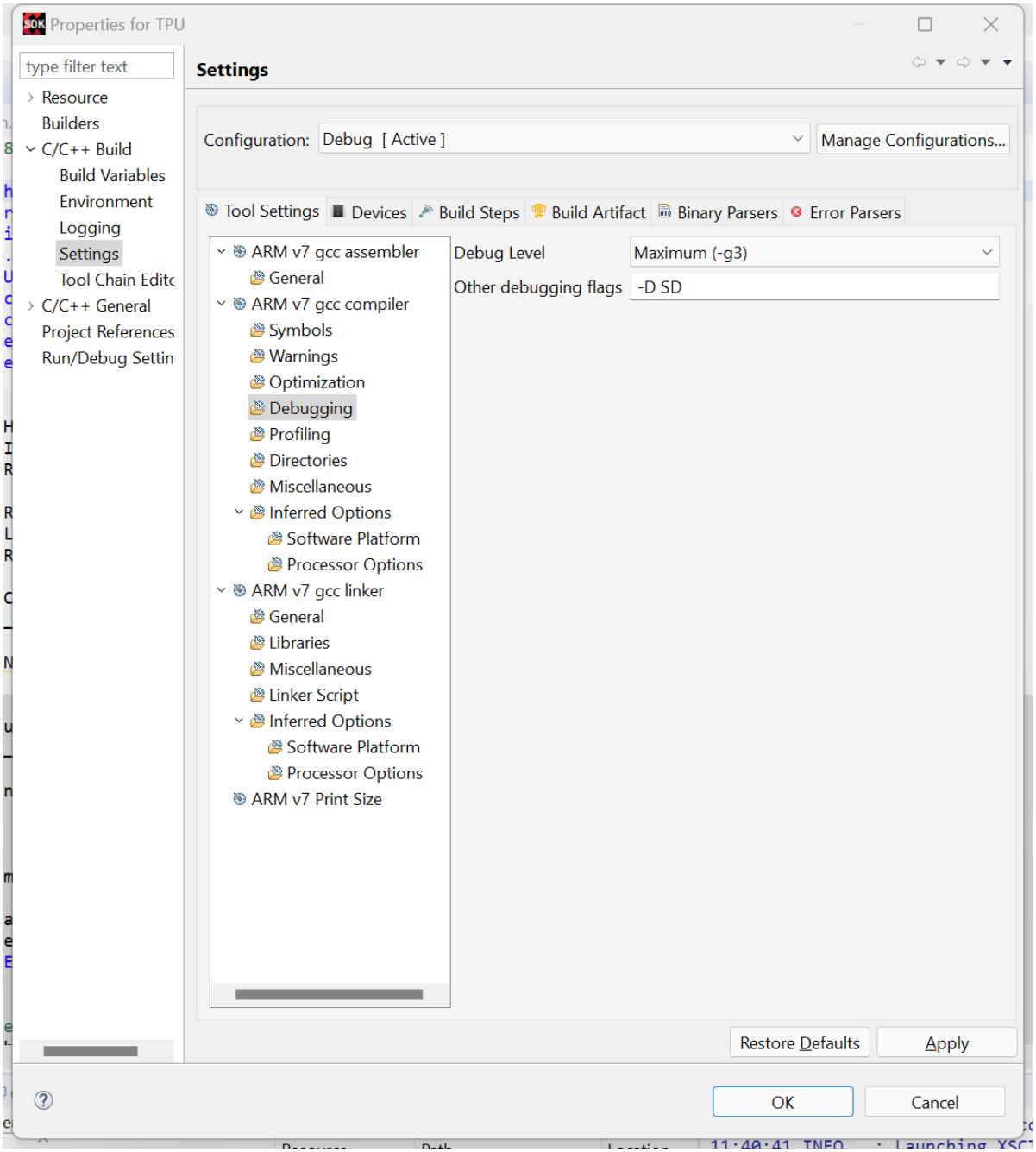
Click standalone, change stdin value as picture and same as stdout.



Click xilffs and change use_strfunc Value as 2.

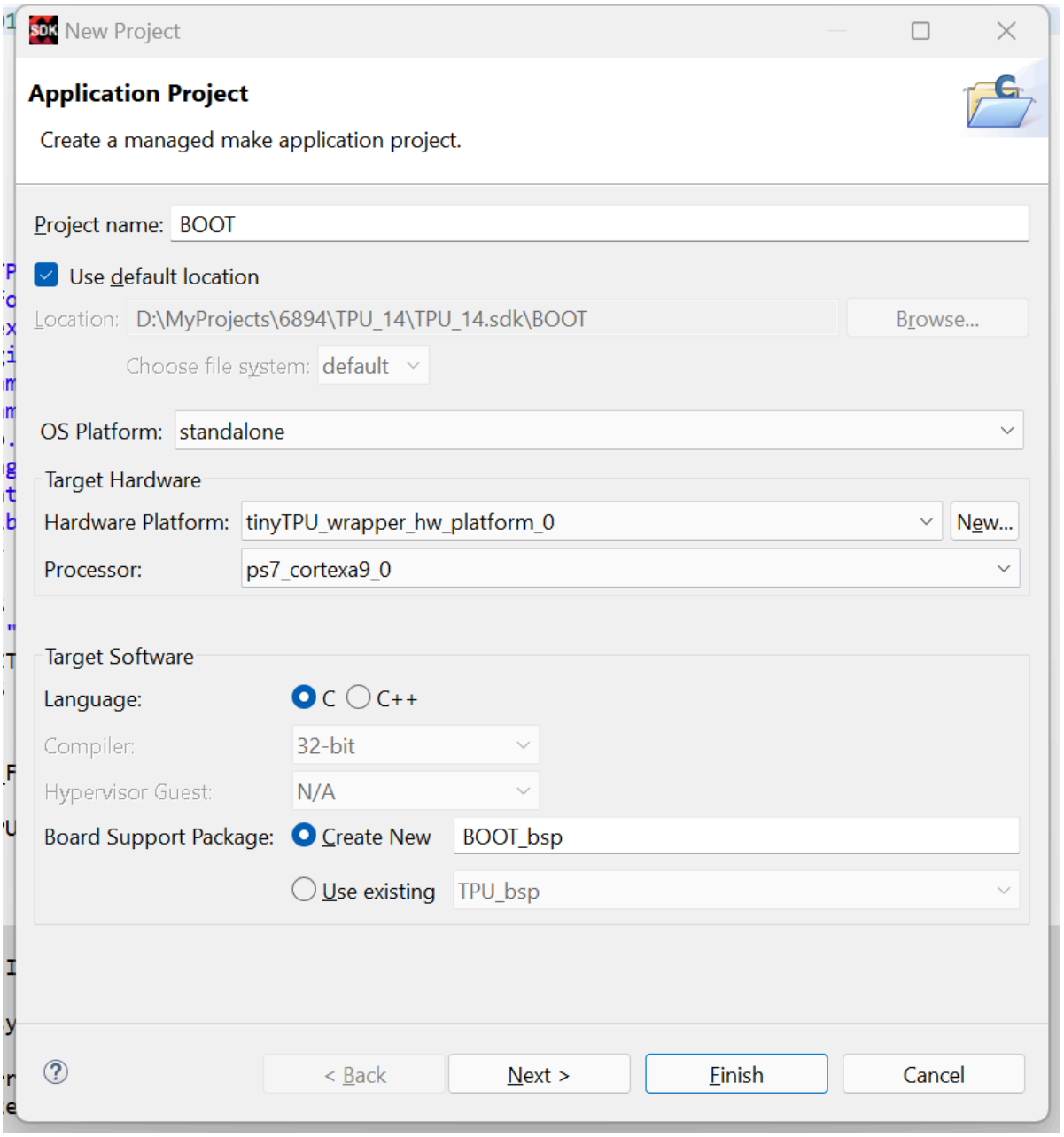


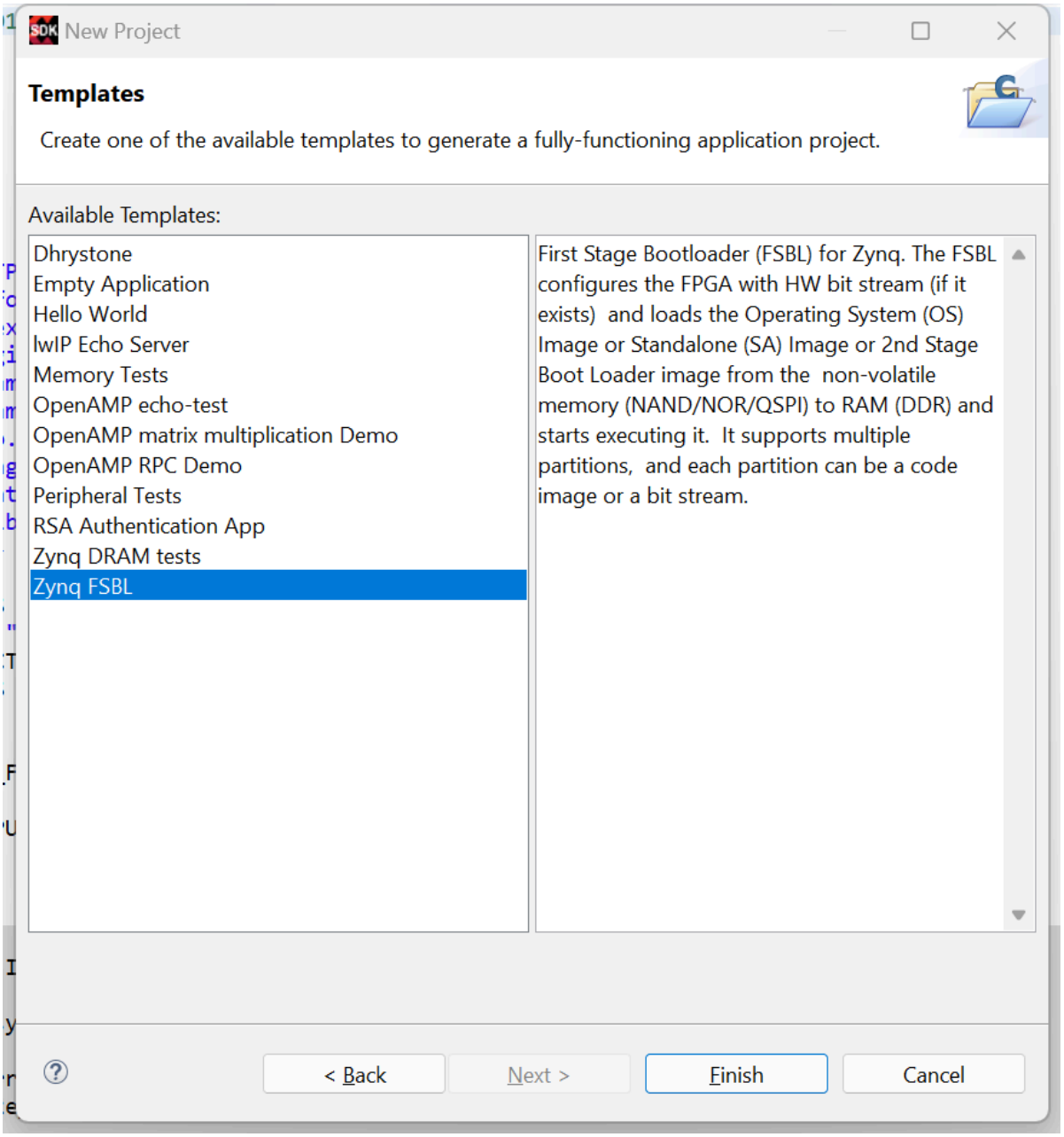
Now right click on the project and choose C/C++ Build Settings. Add the shown line to Debugging.



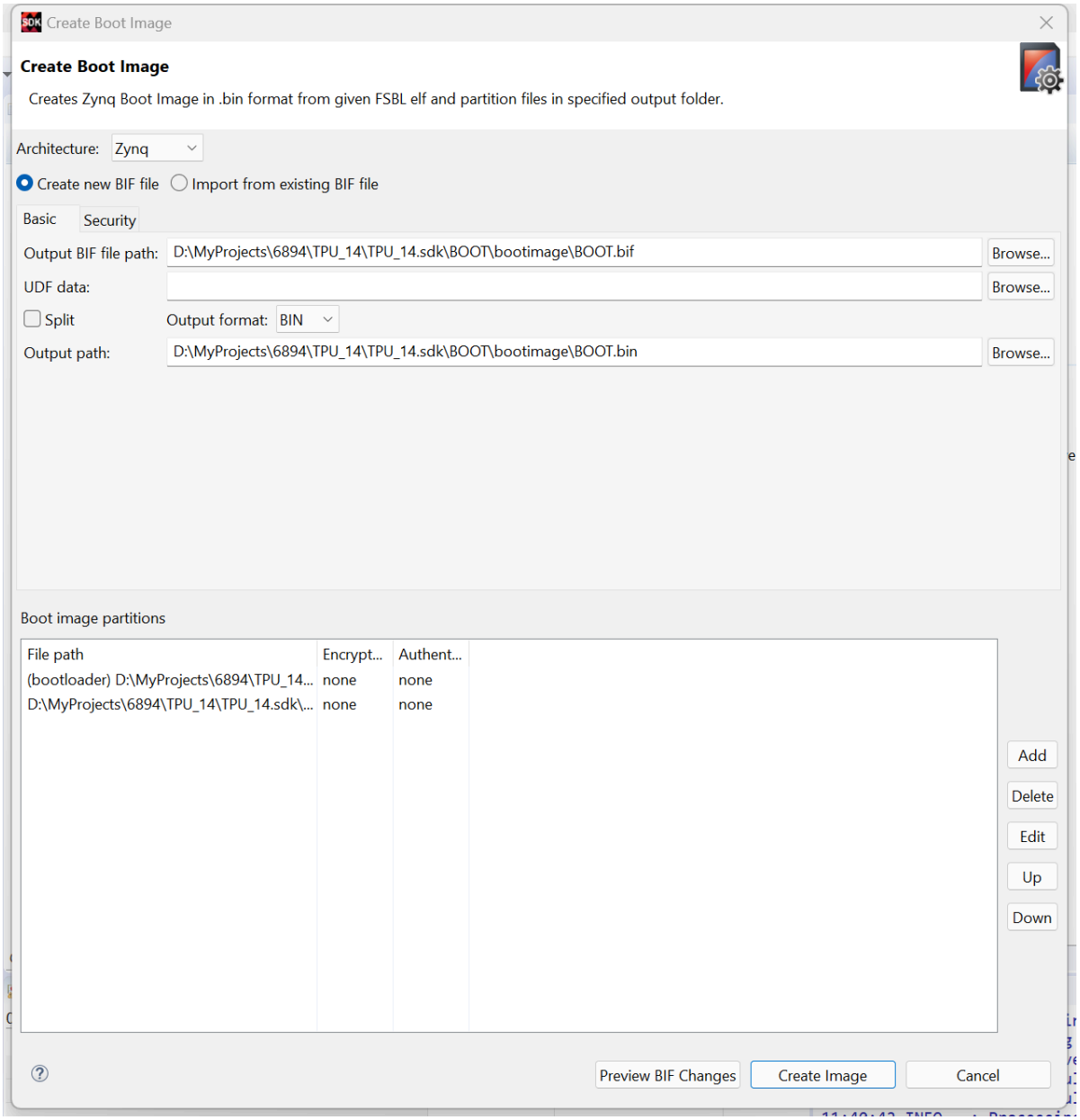
Create FSBL and Boot Image

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

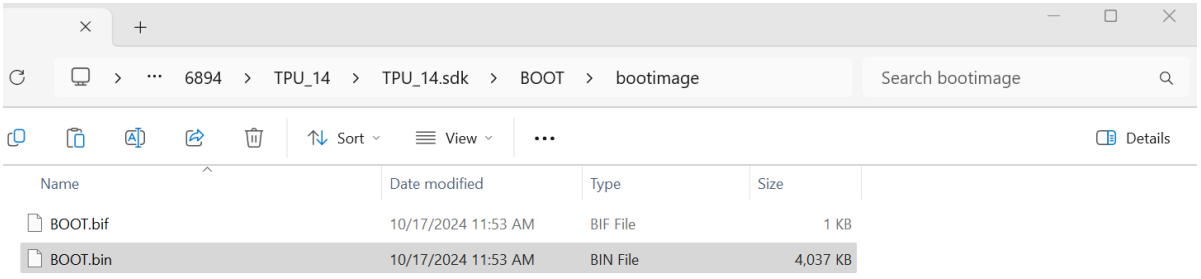

New Project
Application Project
Create a managed make application project.
Project name: BOOT
Use default location
Location: D:\MyProjects\6894\TPU_14\TPU_14.sdk\BOOT
Browse...
Choose file system: default
OS Platform: standalone
Target Hardware
Hardware Platform: tinyTPU_wrapper_hw_platform_0
New...
Processor: ps7_cortexa9_0
Target Software
Language: C C++
Compiler: 32-bit
Hypervisor Guest: N/A
Board Support Package: Create New BOOT_bsp
Use existing TPU_bsp
< Back
Next >
Finish
Cancel



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



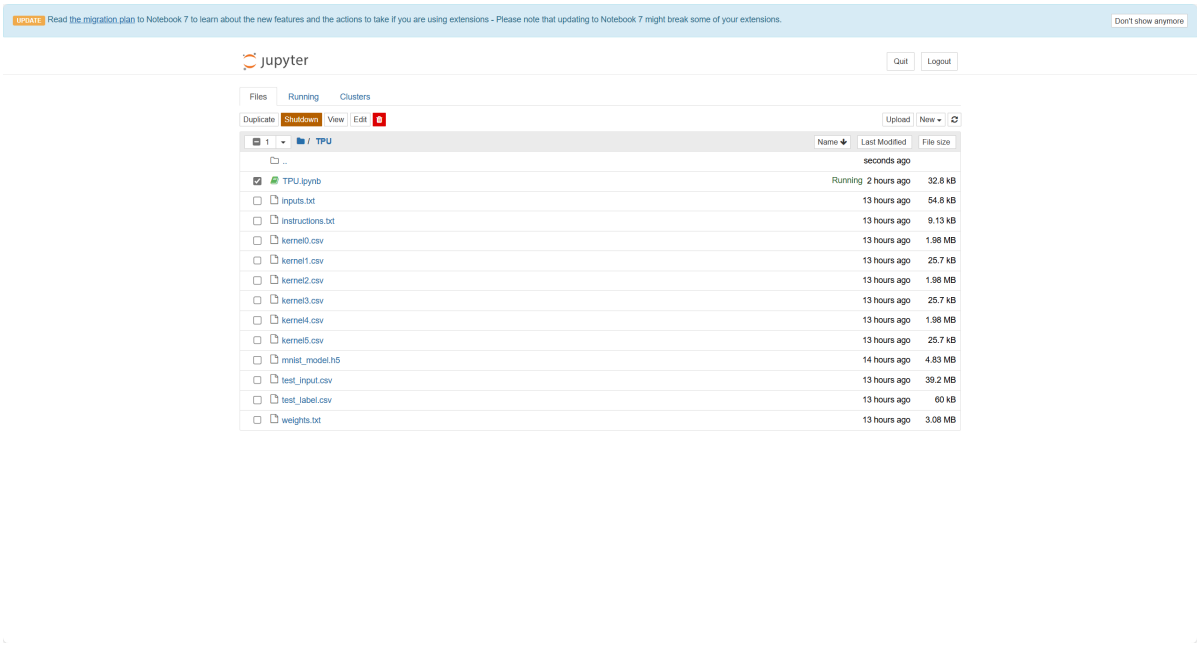
Click Create Image and copy the generated file BOOT.bin to the SD card.



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 se the follow document generated.



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.