OpenTSN Hardware Development Manual

theme	OpenTSN Hardware Development Manual
document number	
creation time	2019-12-17
Last Review	2019-12-17
version number	1.0
file name	OpenTSN Hardware Development Manual.pdf
file format	Portable Document Format

content

1. Document overview

This document is the OpenTSN hardware development and design manual. The document mainly introduces how to use the code on the cloud FPGA_OS netlist and UM code to build hardware project, and use of Vivado development tools.

2. Description of the development environment

Before building the hardware project, you need to install the Vivado2018.3 development tools, and OpenTSN Hardware netlist and UM source code.

(1) Vivado development tools installation

Since the OpenTSN hardware engineering netlist is generated on the 2018.3 version of the Vivado development tools Yes, in order to avoid compatibility issues with inconsistent versions, it is recommended to use the 2018.3 version of Vivado development tools. The download link is as follows:

https://china.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/vivado-design-tools/2018-3.html

Vivado installation requires about 34G of disk space, please reserve enough space first. Install After completion, license cracking is required, you can refer to the online cracking tutorial.

(2) Preparation of engineering documents

The hardware files required for OpenTSN hardware engineering are as follows: 1) FPGA_OS netlist file; 2) engineering 3) project top-level file; 4) UM source code. All the above files can be found on the code cloud OpenTSN open source project download, FPGA_OS is the 1in_4out version. The download address is as follows:

https://gitee.com/opentsn/openTSN FPGA_OS file directory:

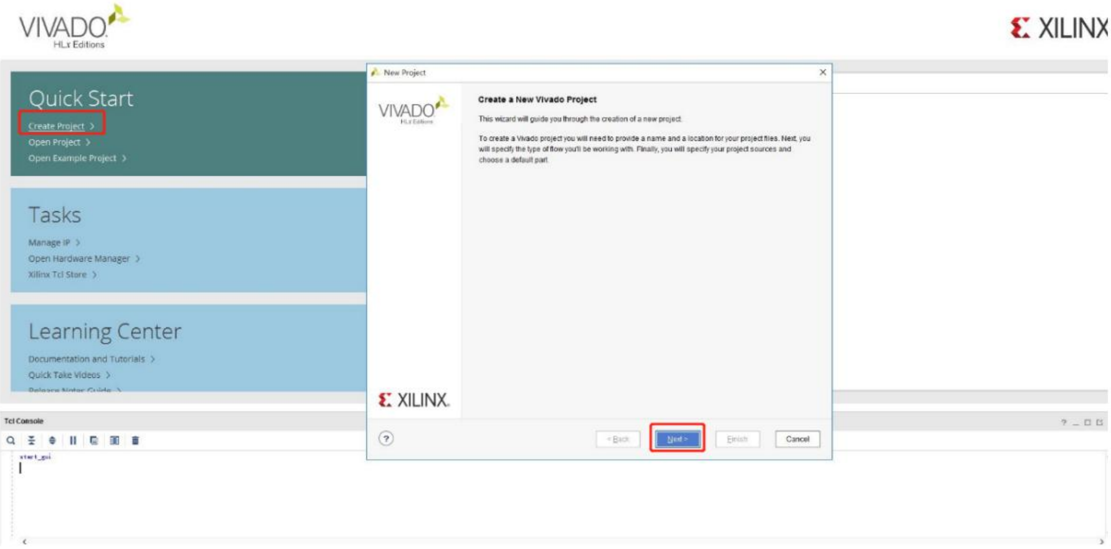
opentsn-openTSN-master/openTSN/sys/fast/fast_os/FPGA_OS_1in_4out

UM source directory:

opentsn-openTSN-master/openTSN/src/hardwarecode/tsn_switch/um_code

3. Build a netlist project

1. Open the Vivado 2018.3 development tool and select Create Project;



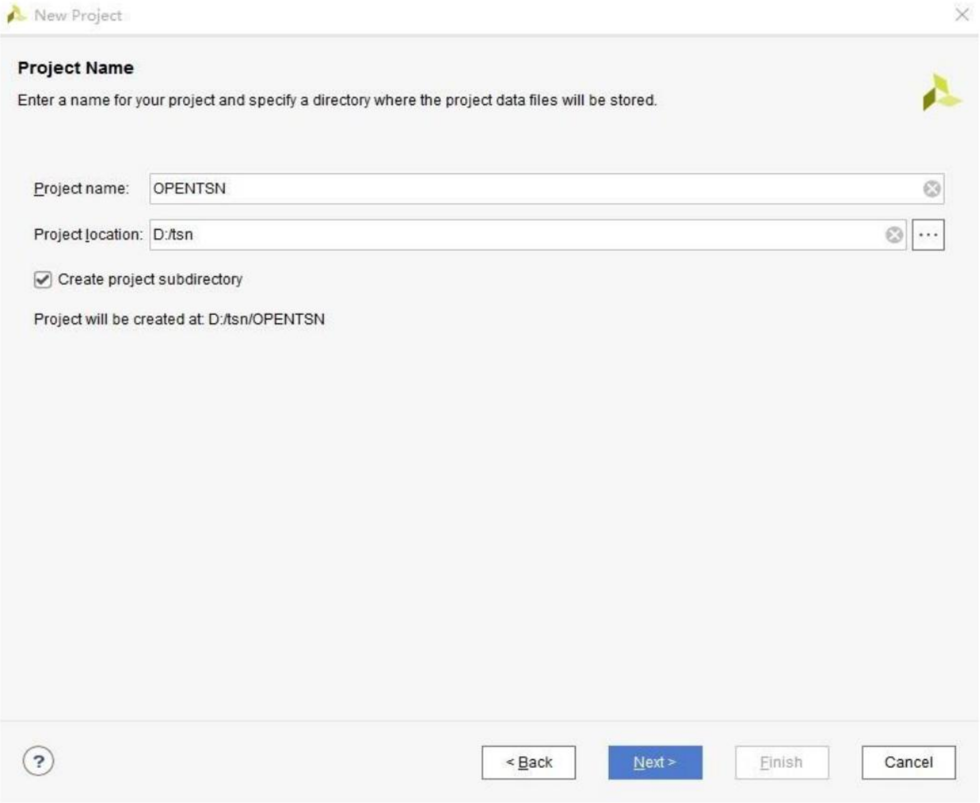
Create a New Vivado Project

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.

```
start_gui
```

2. Enter the project name and the directory where the project is located, and click Next;



Project Name

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

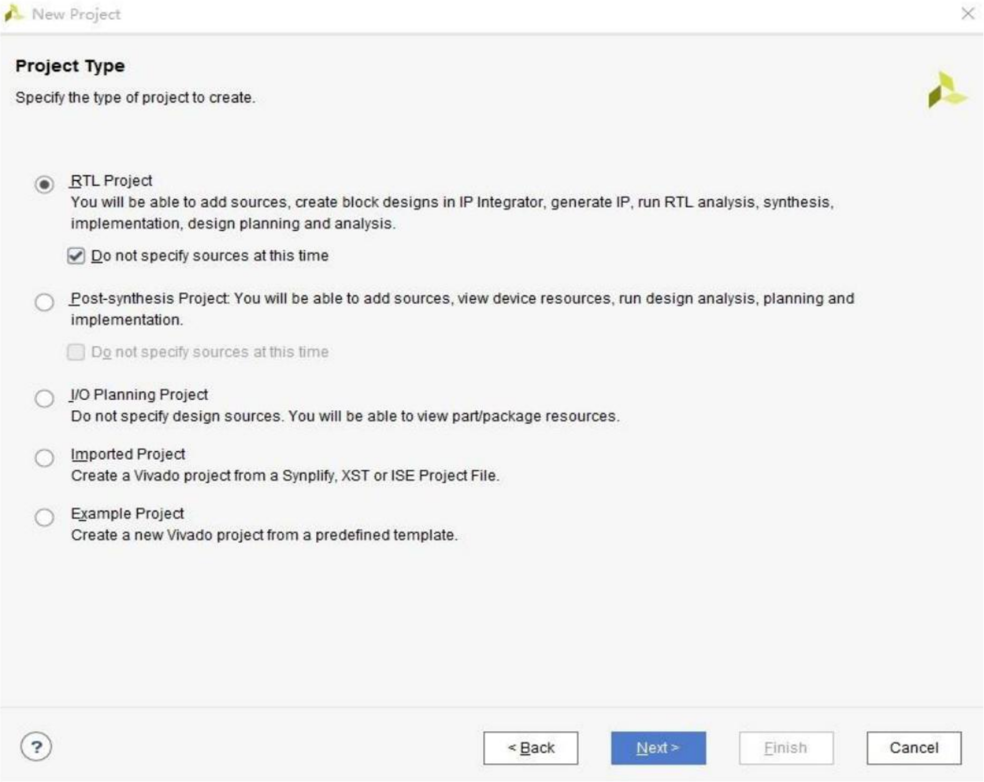
Project name: OPENTSN

Project location: D:/tsn

☑ Create project subdirectory

Project will be created at: D:/tsn/OPENTSN

3. Select Create RTL Project;



Project Type

Specify the type of project to create.

● RTL Project
You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis.

☑ Do not specify sources at this time

○ Post-synthesis Project: You will be able to add sources, view device resources, run design analysis, planning and implementation.

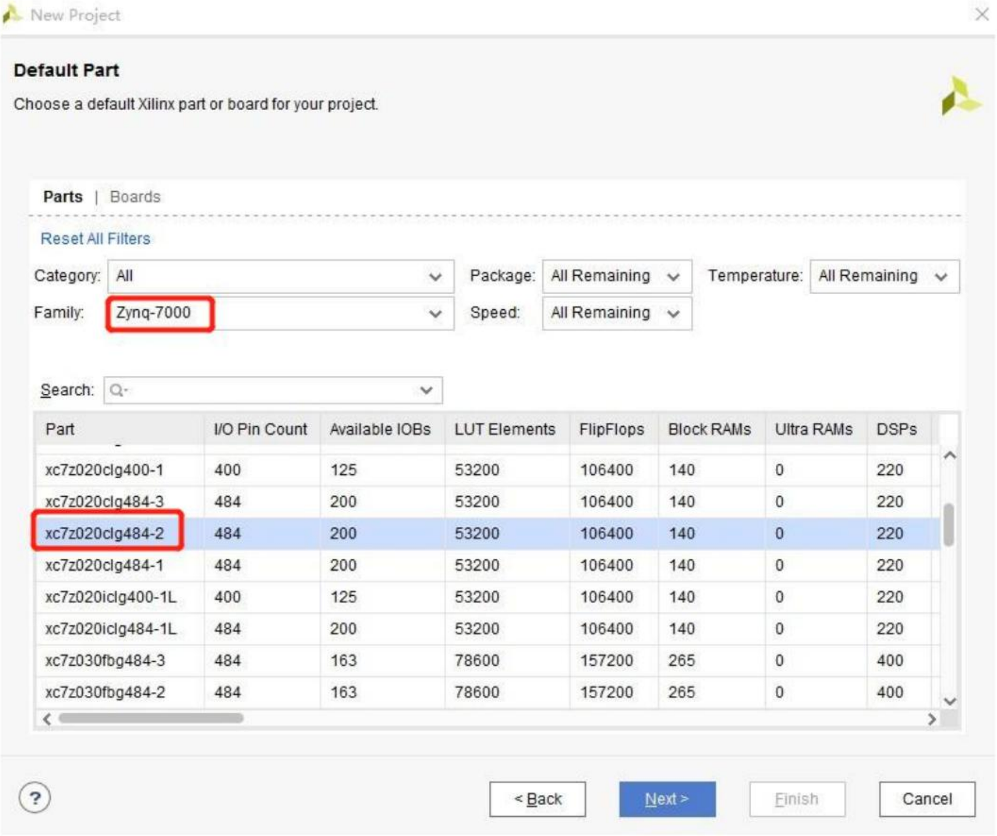
□ Do not specify sources at this time

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

○ Imported Project
Create a Vivado project from a Synplify, XST or ISE Project File.

○ Example Project
Create a new Vivado project from a predefined template.

4. Select the device model as xc7z020clg484-2 in the Zynq-7000 series, and click Next;



Default Part

Choose a default Xilinx part or board for your project.

Category: All

Package: All Remaining

Temperature: All Remaining

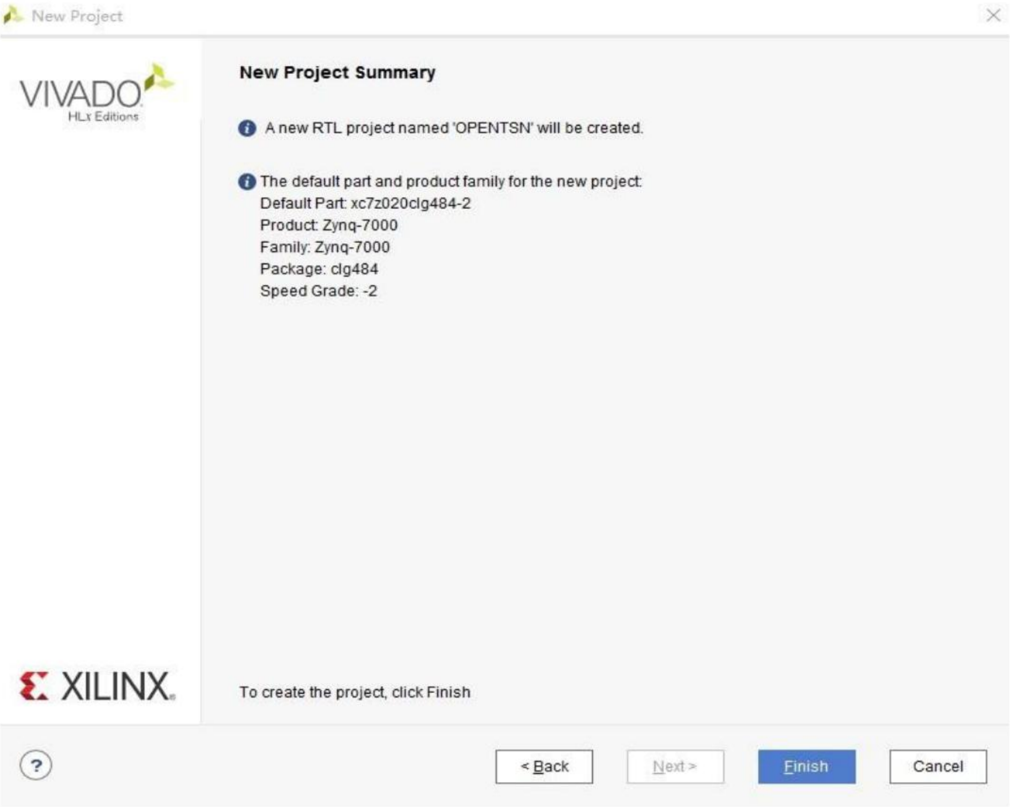
Family: Zynq-7000

Speed: All Remaining

Search: Q-

Part	I/O Pin Count	Available IOBs	LUT Elements	FlipFlops	Block RAMs	Ultra RAMs	DSPs
xc7z020clg400-1	400	125	53200	106400	140	0	220
xc7z020clg484-3	484	200	53200	106400	140	0	220
xc7z020clg484-2	484	200	53200	106400	140	0	220
xc7z020clg484-1	484	200	53200	106400	140	0	220
xc7z020iclg400-1L	400	125	53200	106400	140	0	220
xc7z020iclg484-1L	484	200	53200	106400	140	0	220
xc7z030fbg484-3	484	163	78600	157200	265	0	400
xc7z030fbg484-2	484	163	78600	157200	265	0	400

5. Click Finish to complete the project creation;



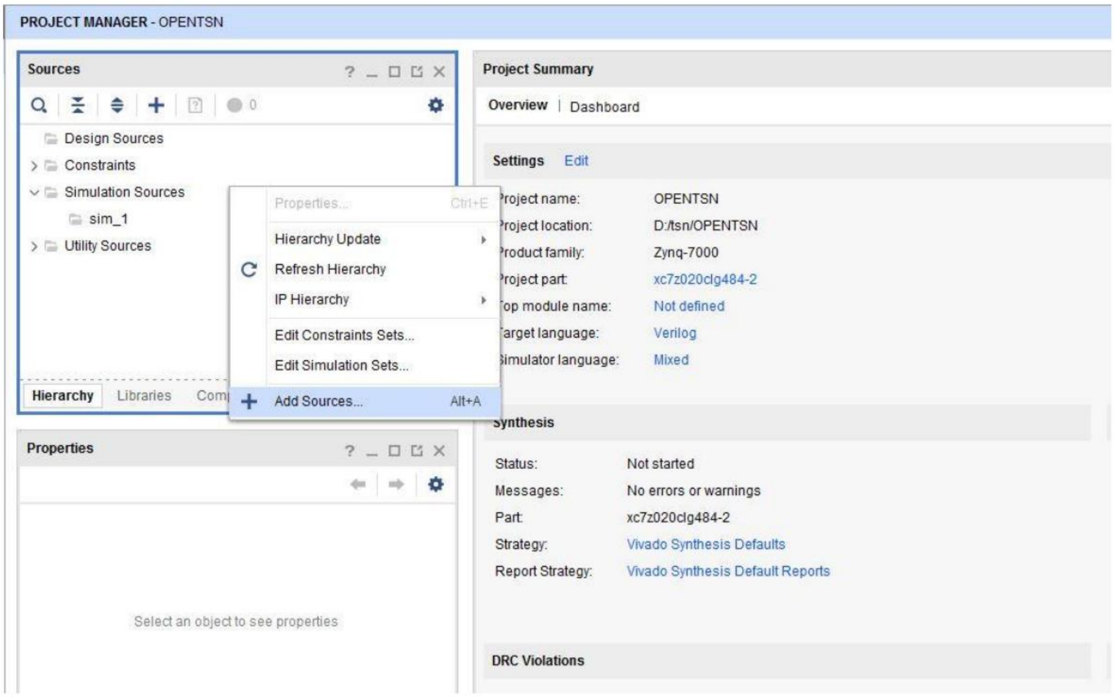
New Project Summary

A new RTL project named 'OPENTSN' will be created.

The default part and product family for the new project:
Default Part: xc7z020clg484-2
Product: Zynq-7000
Family: Zynq-7000
Package: clg484
Speed Grade: -2

To create the project, click Finish

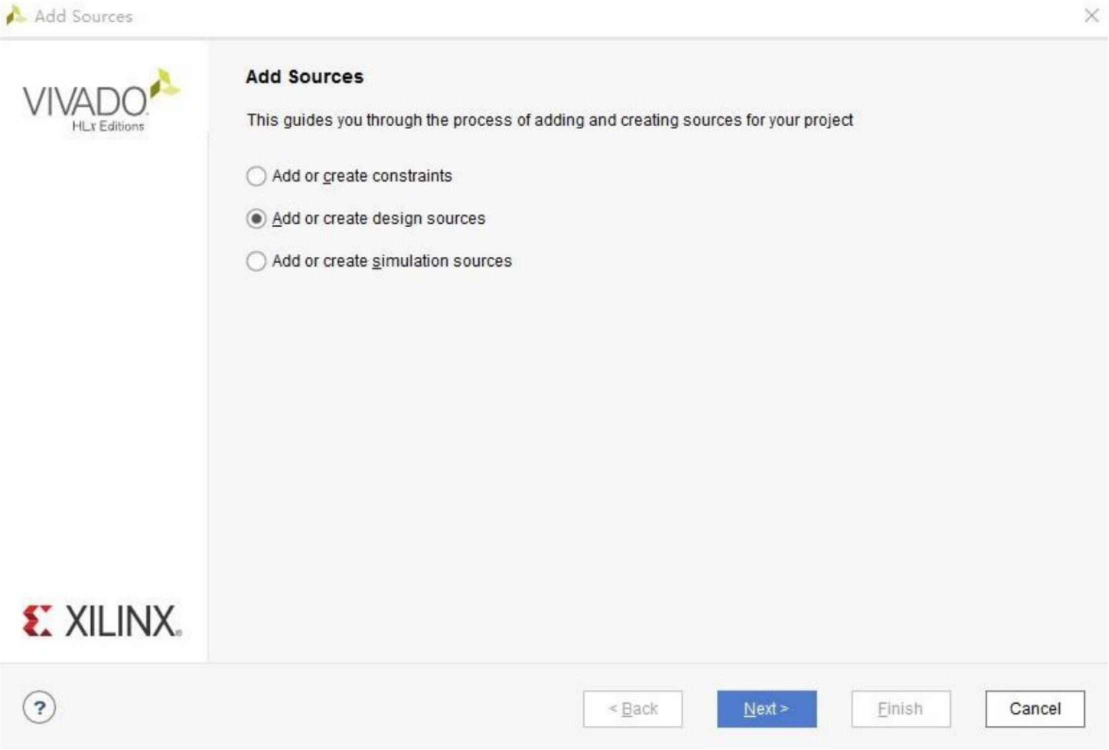
6. Right-click in the Sources window and select Add Sources to add project source files;



roject name: OPENTSN
roject location: D:/tsn/OPENTSN
roduct family: Zynq-7000
roject part: xc7z020clg484-2
op module name: Not defined
arget language: Verilog
imulator language: Mixed

Status: Not started
Messages: No errors or warnings
Part: xc7z020clg484-2
Strategy: Vivado Synthesis Defaults
Report Strategy: Vivado Synthesis Default Reports

7. Select Add or create design sources to add engineering design files;



Add Sources

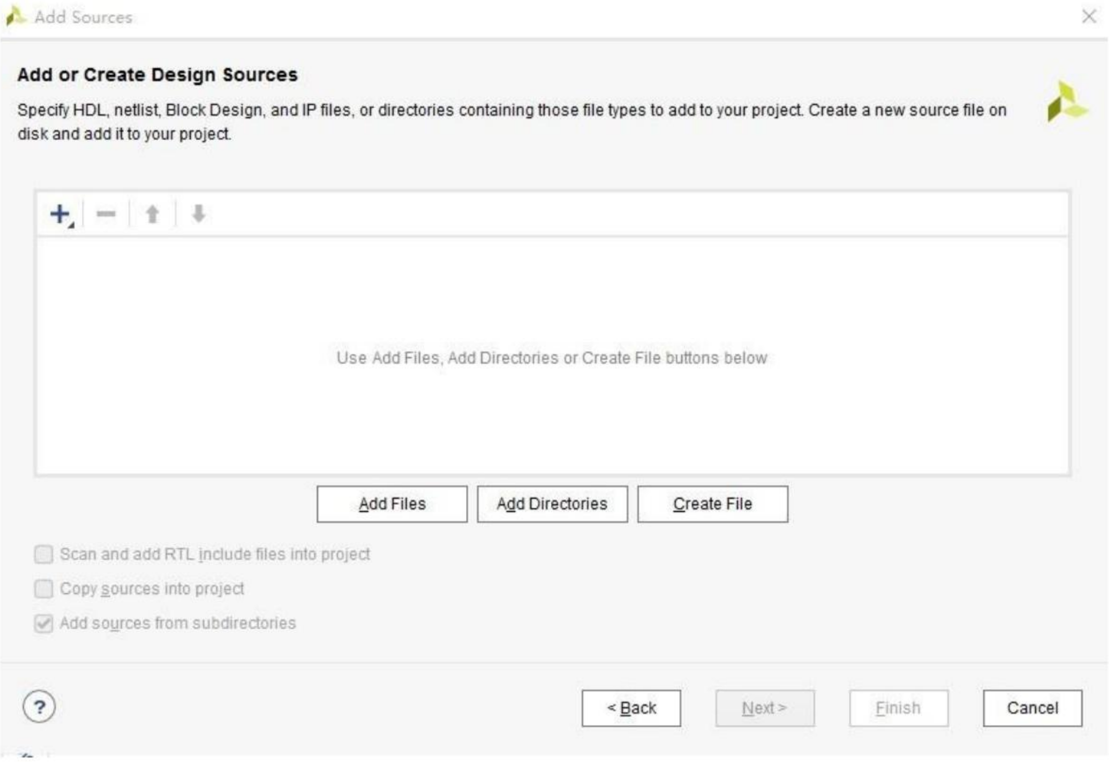
This guides you through the process of adding and creating sources for your project

○ Add or create constraints

● Add or create design sources

○ Add or create simulation sources

8. Click Add Files;



Add or Create Design Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those file types to add to your project. Create a new source file on disk and add it to your project.

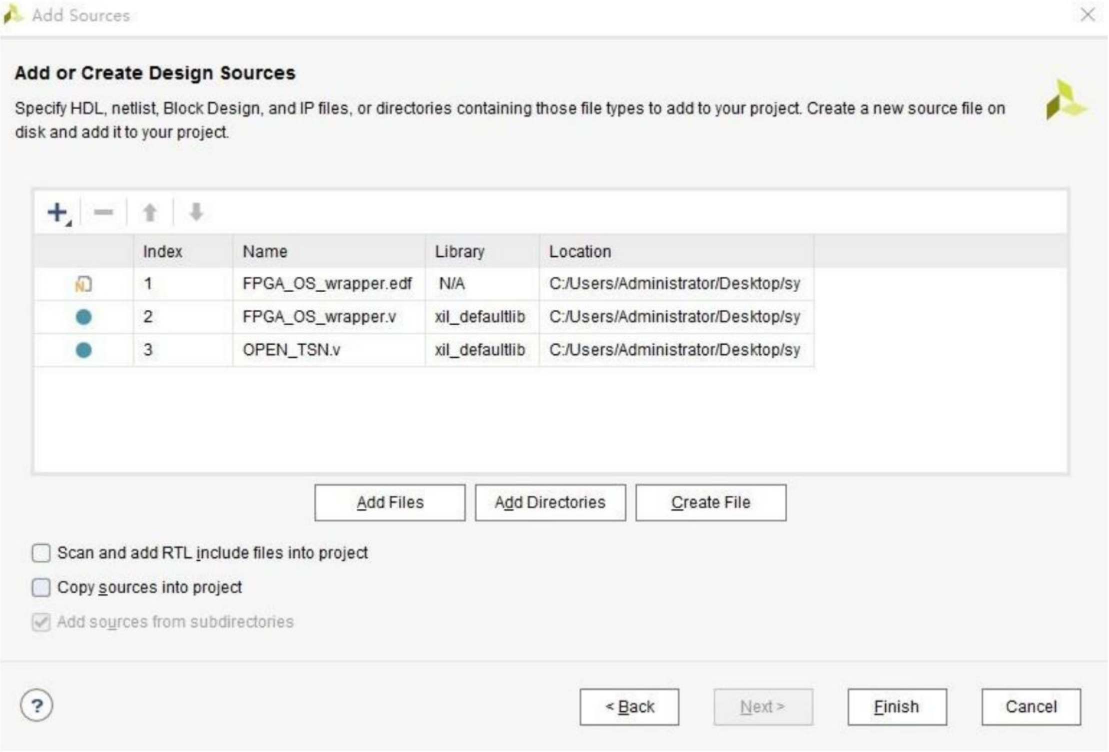
Use Add Files, Add Directories or Create File buttons below

□ Scan and add RTL include files into project

□ Copy sources into project

☑ Add sources from subdirectories

9. Add FPGA OS netlist file and project top-level file;



Add Sources

Add or Create Design Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those file types to add to your project. Create a new source file on disk and add it to your project.

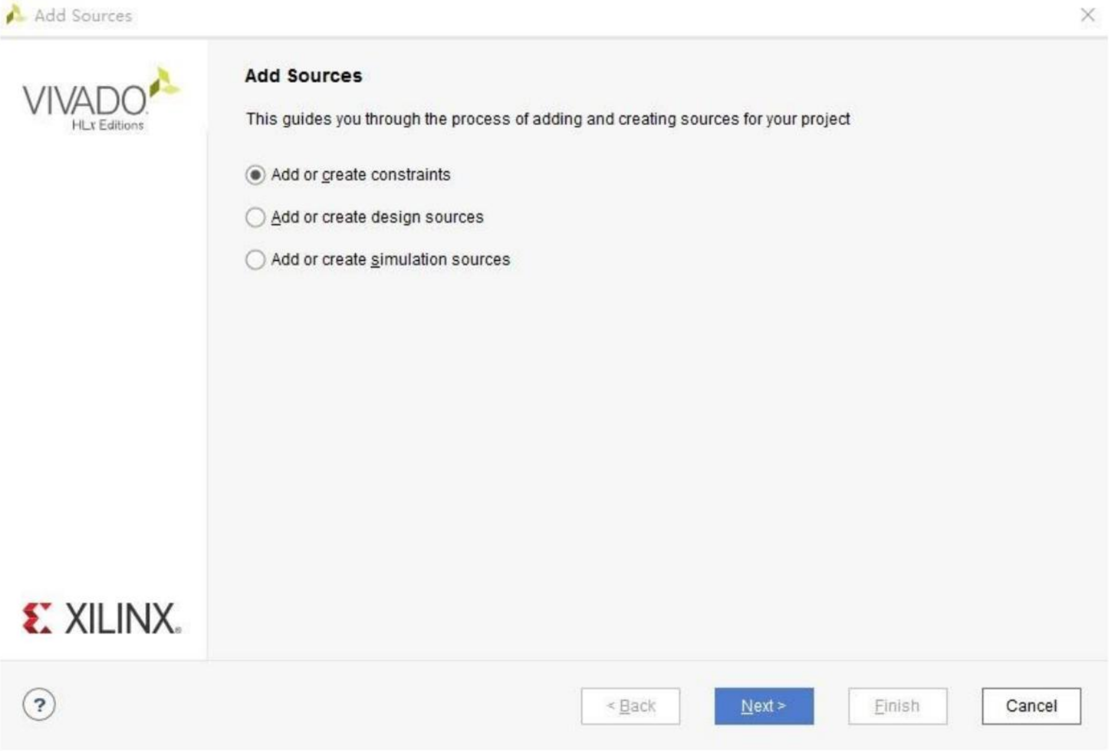
	Index	Name	Library	Location
	1	FPGA_OS_wrapper.edf	N/A	C:/Users/Administrator/Desktop/sy
●	2	FPGA_OS_wrapper.v	xil_defaultlib	C:/Users/Administrator/Desktop/sy
●	3	OPEN_TSN.v	xil_defaultlib	C:/Users/Administrator/Desktop/sy

□ Scan and add RTL include files into project

□ Copy sources into project

☑ Add sources from subdirectories

10. Add project constraint file;



Add Sources

Add Sources

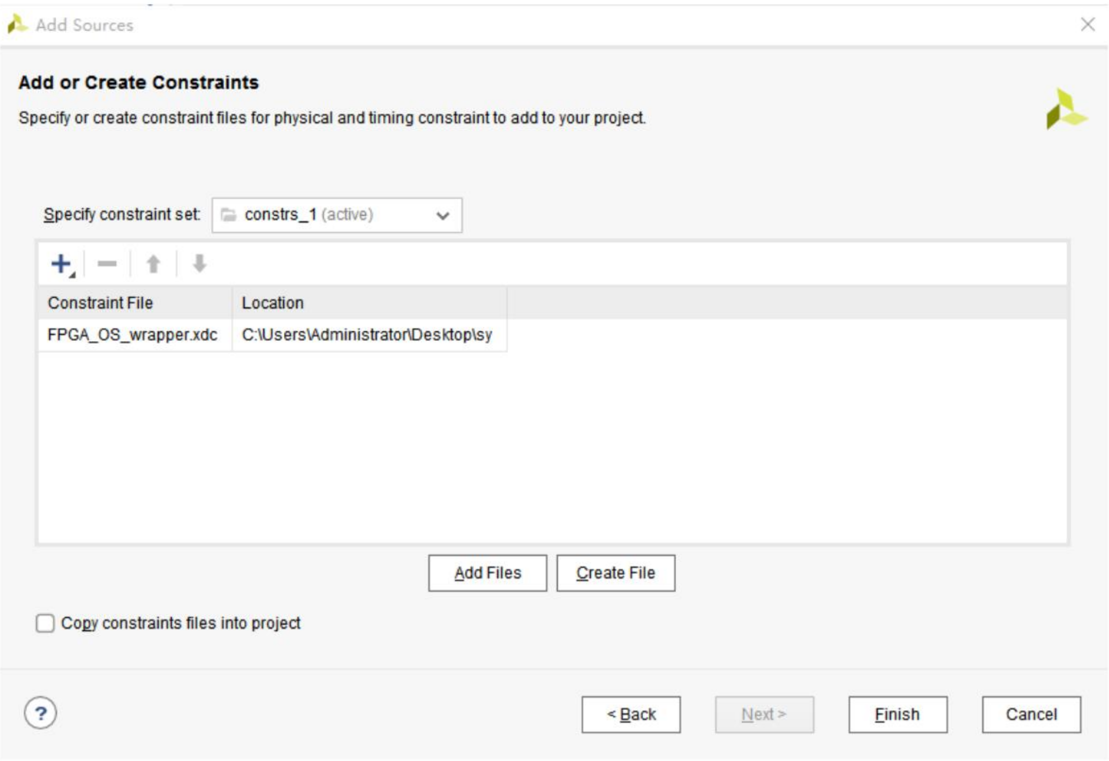
This guides you through the process of adding and creating sources for your project

● Add or create constraints

○ Add or create design sources

○ Add or create simulation sources

11. Select the FPGA_OS_wrapper.xdc file;



Add Sources

Add or Create Constraints

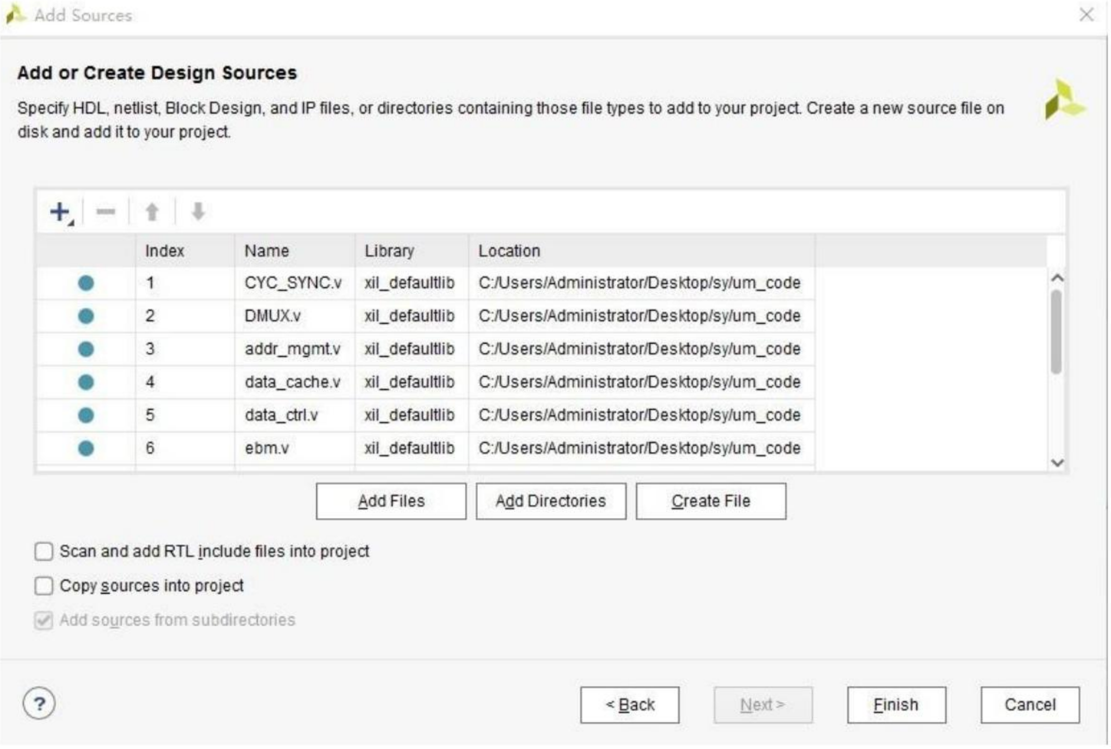
Specify or create constraint files for physical and timing constraint to add to your project.

Specify constraint set: constrs_1 (active)

Constraint File	Location
FPGA_OS_wrapper.xdc	C:\Users\Administrator\Desktop\sy

☐ Copy constraints files into project

12. Add the UM source file of OpenTSN;



Add Sources

Add or Create Design Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those file types to add to your project. Create a new source file on disk and add it to your project.

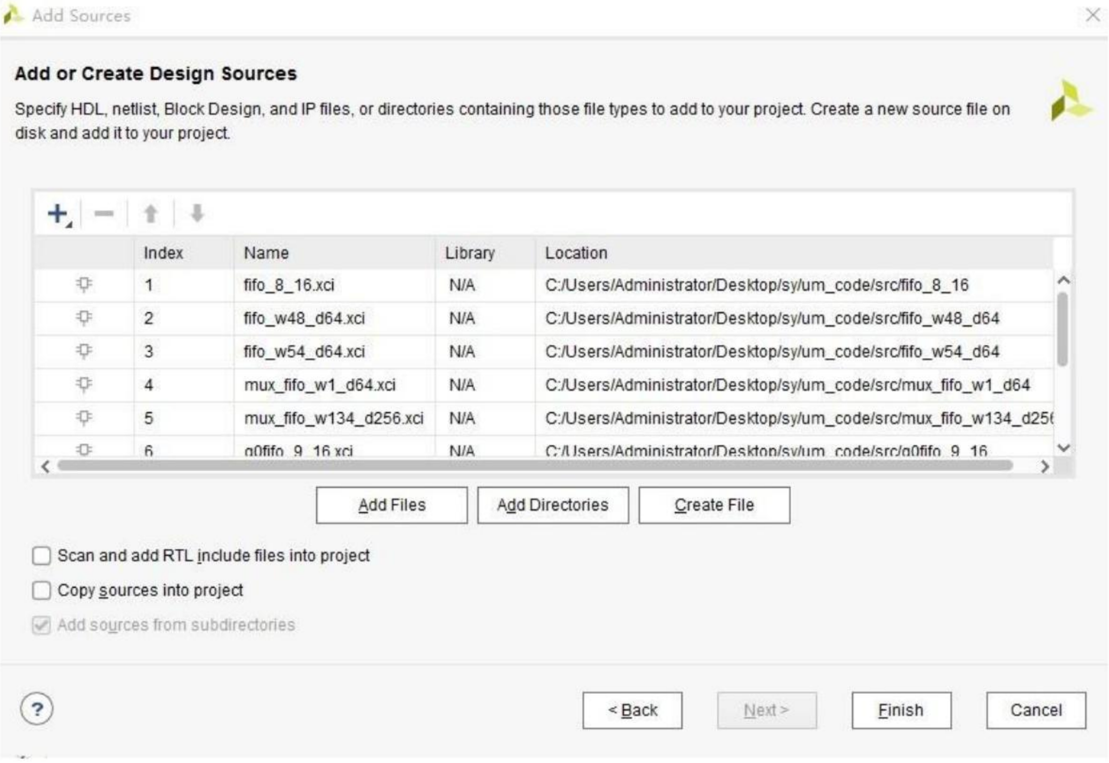
	Index	Name	Library	Location
●	1	CYC_SYNC.v	xil_defaultlib	C:/Users/Administrator/Desktop/sy/um_code
●	2	DMUX.v	xil_defaultlib	C:/Users/Administrator/Desktop/sy/um_code
●	3	addr_mgmt.v	xil_defaultlib	C:/Users/Administrator/Desktop/sy/um_code
●	4	data_cache.v	xil_defaultlib	C:/Users/Administrator/Desktop/sy/um_code
●	5	data_ctrl.v	xil_defaultlib	C:/Users/Administrator/Desktop/sy/um_code
●	6	ebm.v	xil_defaultlib	C:/Users/Administrator/Desktop/sy/um_code

☐ Scan and add RTL include files into project

☐ Copy sources into project

☑ Add sources from subdirectories

13. Add the FIFO and RAM source files needed in the module;



Add Sources

Add or Create Design Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those file types to add to your project. Create a new source file on disk and add it to your project.

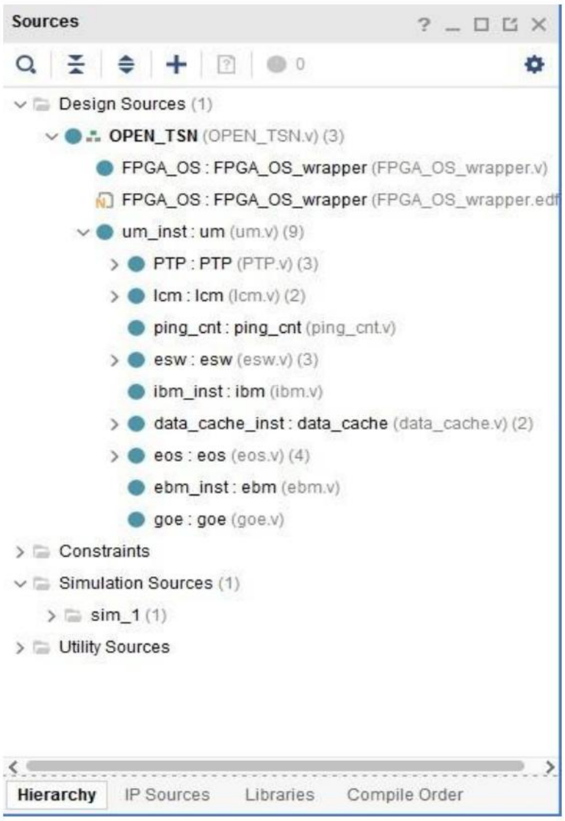
Index	Name	Library	Location
1	fifo_8_16.xci	N/A	C:/Users/Administrator/Desktop/sy/um_code/src/fifo_8_16
2	fifo_w48_d64.xci	N/A	C:/Users/Administrator/Desktop/sy/um_code/src/fifo_w48_d64
3	fifo_w54_d64.xci	N/A	C:/Users/Administrator/Desktop/sy/um_code/src/fifo_w54_d64
4	mux_fifo_w1_d64.xci	N/A	C:/Users/Administrator/Desktop/sy/um_code/src/mux_fifo_w1_d64
5	mux_fifo_w134_d256.xci	N/A	C:/Users/Administrator/Desktop/sy/um_code/src/mux_fifo_w134_d25
6	q0fifo_9_16.xci	N/A	C:/Users/Administrator/Desktop/sy/um_code/src/q0fifo_9_16

□ Scan and add RTL include files into project

□ Copy sources into project

☑ Add sources from subdirectories

14. So far, the whole project is completed, and the structure of the whole project is as follows.

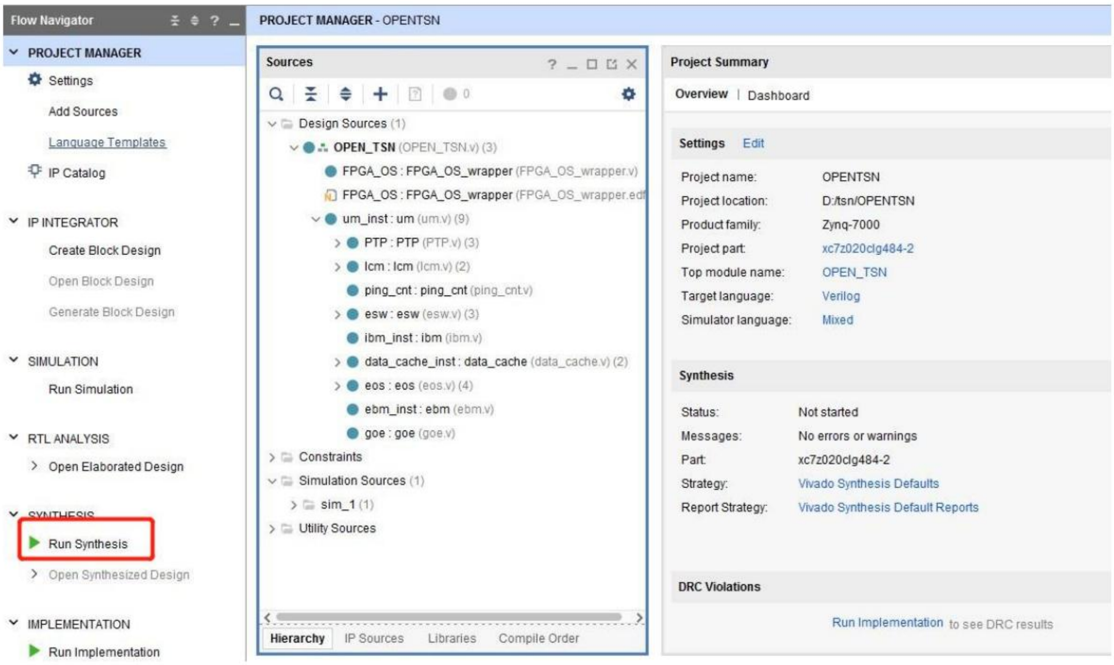


Sources

```
Design Sources (1)
  OPEN_TSN (OPEN_TSN.v) (3)
      FPGA_OS : FPGA_OS_wrapper (FPGA_OS_wrapper.v)
      FPGA_OS : FPGA_OS_wrapper (FPGA_OS_wrapper.edf
    um_inst : um (um.v) (9)
      > PTP : PTP (PTP.v) (3)
      > lcm : lcm (lcm.v) (2)
        ping_cnt : ping_cnt (ping_cnt.v)
      > esw : esw (esw.v) (3)
        ibm_inst : ibm (ibm.v)
      > data_cache_inst : data_cache (data_cache.v) (2)
      > eos : eos (eos.v) (4)
        ebm_inst : ebm (ebm.v)
        goe : goe (goe.v)
> Constraints
Simulation Sources (1)
  > sim_1 (1)
> Utility Sources
```

4. Compile and generate downloadable files

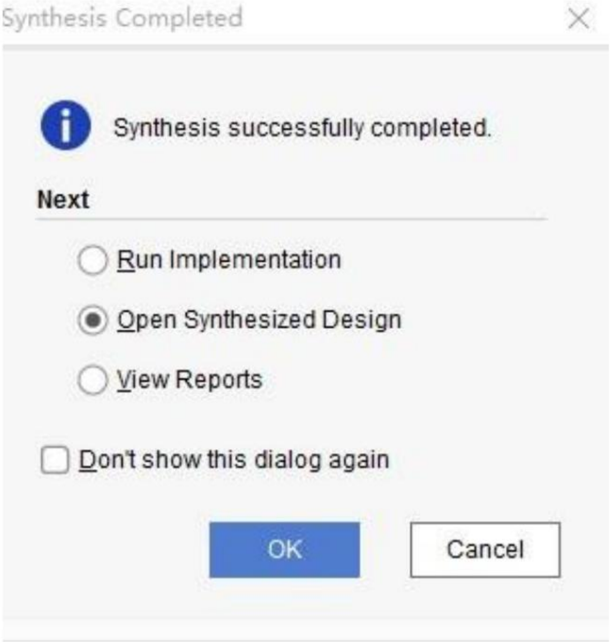
1. Start the comprehensive compilation of the project, and click Run Synthesis in the PROJECT MANAGER window on the left;



Project name: OPENTSN
Project location: D:/tsn/OPENTSN
Product family: Zynq-7000
Project part: xc7z020clg484-2
Top module name: OPEN_TSN
Target language: Verilog
Simulator language: Mixed

Status: Not started
Messages: No errors or warnings
Part: xc7z020clg484-2
Strategy: Vivado Synthesis Defaults
Report Strategy: Vivado Synthesis Default Reports

2. After the synthesis is completed, select Open Synthesizd Design;


Synthesis Completed

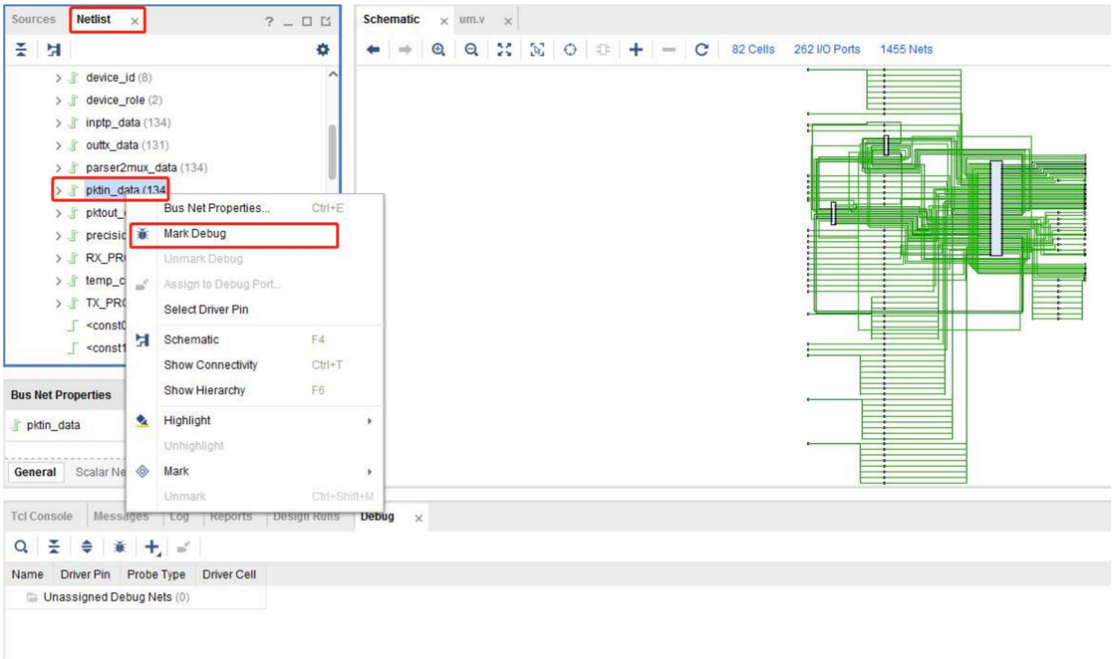
Synthesis successfully completed.

Next

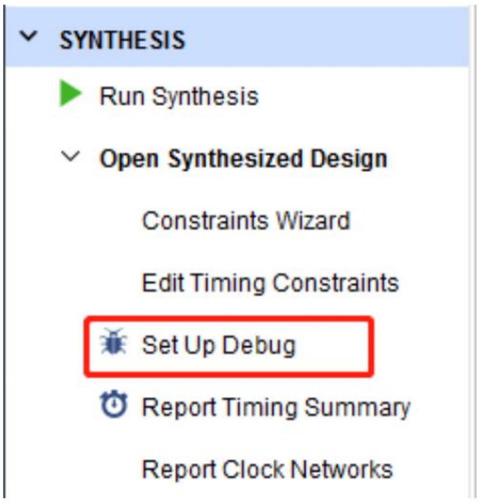
○ Run Implementation
● Open Synthesized Design
○ View Reports

□ Don't show this dialog again

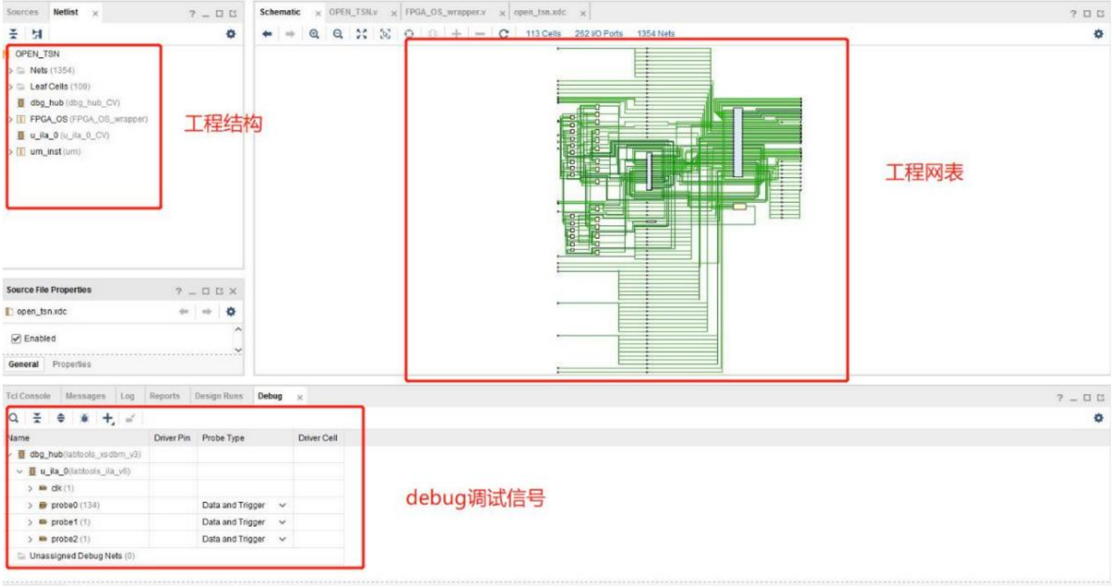
3. Pull the Debug signal you want to observe (optional). Switch to the Netlist tab in the left window, right-click the signal you want to observe, and click Mark Debug to select; if you want to pull more signals, repeat this step;



4. After the pull is completed, click Set Up Debug to form the Debug core;



5. At this point, you can see the structure of the entire project, the netlist structure and the Debug debugging signal;

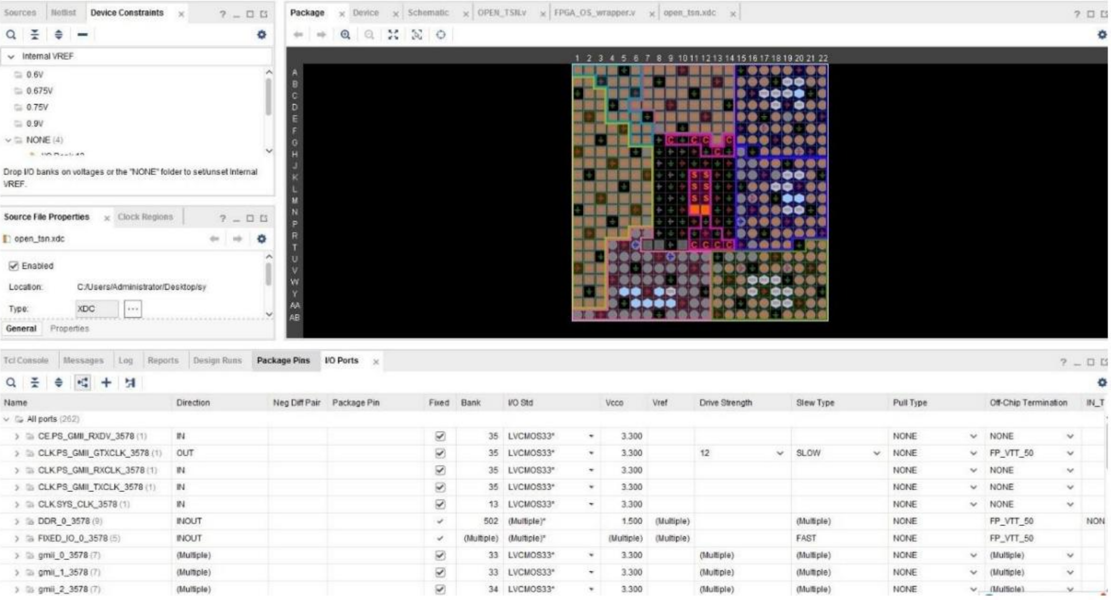


OPEN_TSN
- Nets (1354)
- Leaf Cells (109)
- dbg_hub (dbg_hub_CV)
- FPGA_OS (FPGA_OS_wrapper)
- u_ila_0 (u_ila_0_CV)
- um_inst (um)

Source File Properties: open_tsn.xdc ☑ Enabled

Name	Driver Pin	Probe Type	Driver Cell
dbg_hub (labtools_xsdbm_v3)			
u_ila_0 (labtools_ila_v6)			
clk (1)			
probe0 (134)		Data and Trigger	
probe1 (1)		Data and Trigger	
probe2 (1)		Data and Trigger	
Unassigned Debug Nets (0)			

6. View the pin assignment;



Internal VREF: 0.6V, 0.675V, 0.75V, 0.9V, NONE (4)

Drop I/O banks on voltages or the "NONE" folder to set/unset Internal VREF.

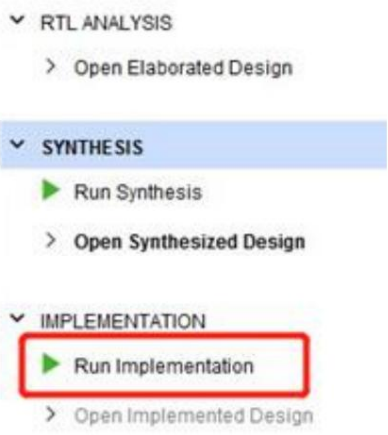
Source File Properties: open_tsn.xdc ☑ Enabled

Location: C:/Users/Administrator/Desktop/sy

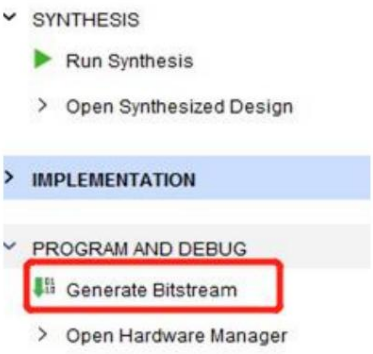
Type: XDC

Name	Direction	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std	Vcco	Vref	Drive Strength	Slew Type	Pull Type	Off-Chip Termination	IN_T
All ports (262)													
CE.PS_GMII_RXDV_3578 (1)	IN			☑	35	LVCMOS33*	3.300				NONE	NONE	
CLK.PS_GMII_GTXCLK_3578 (1)	OUT			☑	35	LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
CLK.PS_GMII_RXCLK_3578 (1)	IN			☑	35	LVCMOS33*	3.300				NONE	NONE	
CLK.PS_GMII_TXCLK_3578 (1)	IN			☑	35	LVCMOS33*	3.300				NONE	NONE	
CLK.SYS_CLK_3578 (1)	IN			☑	13	LVCMOS33*	3.300				NONE	NONE	
DDR_0_3578 (9)	INOUT			✓	502	(Multiple)*	1.500	(Multiple)		(Multiple)	NONE	FP_VTT_50	NON
FIXED_IO_0_3578 (5)	INOUT			✓	(Multiple)	(Multiple)*	(Multiple)	(Multiple)		FAST	NONE	FP_VTT_50	
gmii_0_3578 (7)	(Multiple)			☑	33	LVCMOS33*	3.300		(Multiple)	(Multiple)	NONE	(Multiple)	
gmii_1_3578 (7)	(Multiple)			☑	33	LVCMOS33*	3.300		(Multiple)	(Multiple)	NONE	(Multiple)	
gmii_2_3578 (7)	(Multiple)			☑	34	LVCMOS33*	3.300		(Multiple)	(Multiple)	NONE	(Multiple)	

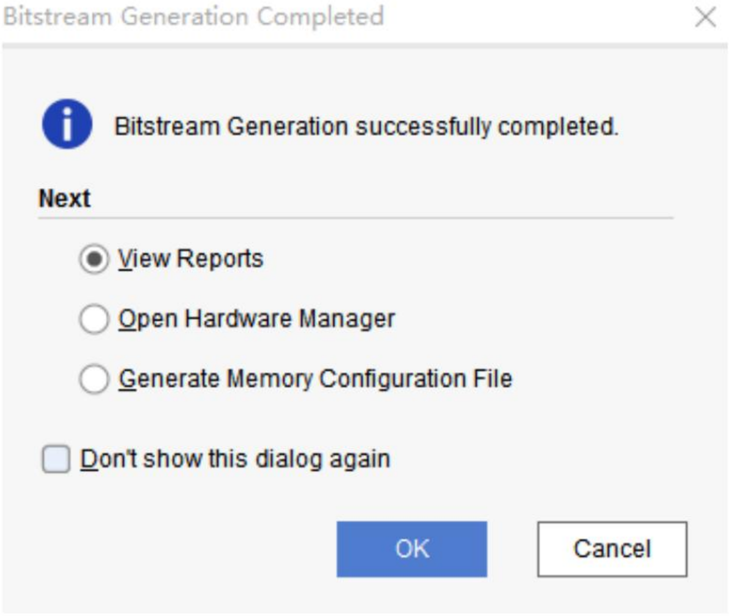
7. After saving, click Run Implementation to place and route, and check the timing;



8. After the compilation is complete, click Generate Bitstream to generate a downloadable bit stream file;



9. The hardware downloadable logic file is generated.


Bitstream Generation Completed

Bitstream Generation successfully completed.

Next

● View Reports

○ Open Hardware Manager

○ Generate Memory Configuration File

□ Don't show this dialog again

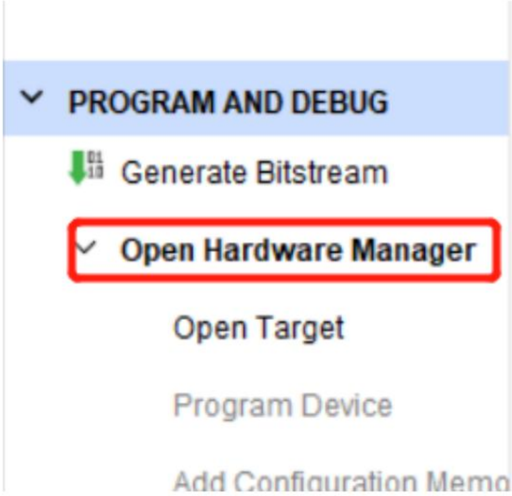
5. Hardware logic download

1. Use the download cable provided with the device to connect the Jtag port of the device and the host computer, pay attention to connect first and then power on, do not bring

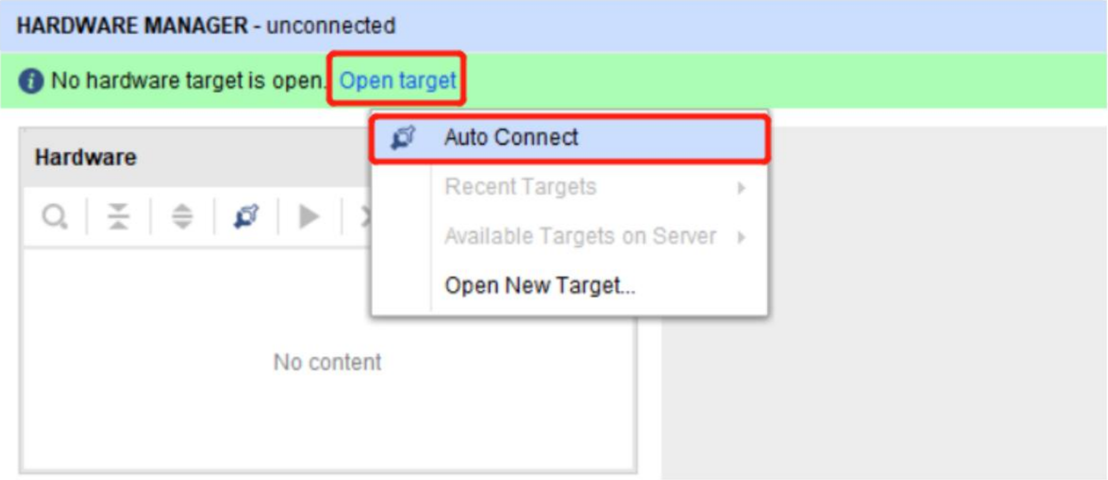
Electric plugging, easy to burn out the Jtag port of the device;

2. Click Open Hardware Manager in the left window of the Vivado main interface to open Vivado to download and debug

tool;



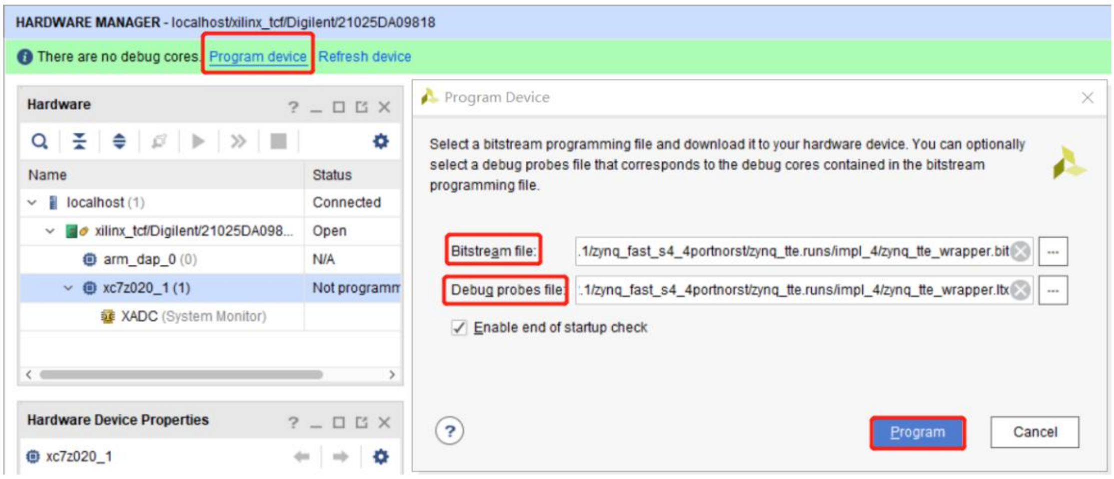
3. Click Open Target >> Auto Connect to automatically search for devices;



HARDWARE MANAGER - unconnected

No hardware target is open. Open target

4. Click Program device, Vivado will automatically select the generated logic file and Debug debug file, click Program to start the download;



HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/21025DA09818

There are no debug cores. Program device Refresh device

Name	Status
localhost (1)	Connected
xilinx_tcf/Digilent/21025DA098...	Open
arm_dap_0 (0)	N/A
xc7z020_1 (1)	Not programm
XADC (System Monitor)	

Program Device

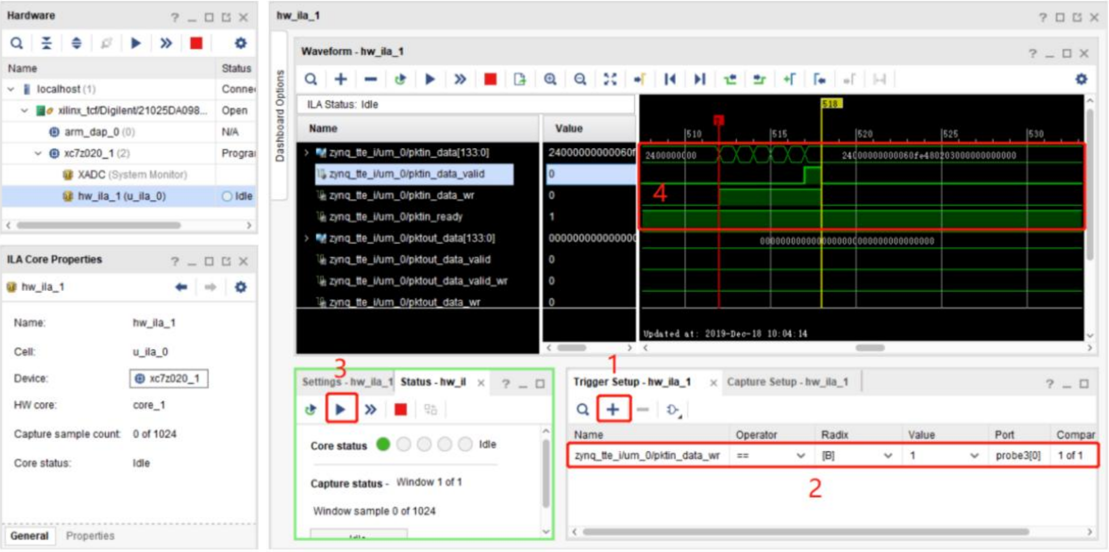
Select a bitstream programming file and download it to your hardware device. You can optionally select a debug probes file that corresponds to the debug cores contained in the bitstream programming file.

Bitstream file: .1/zynq_fast_s4_4portnorst/zynq_tte.runs/impl_4/zynq_tte_wrapper.bit

Debug probes file: .1/zynq_fast_s4_4portnorst/zynq_tte.runs/impl_4/zynq_tte_wrapper.ltx

☑ Enable end of startup check

5. After the download is complete, start to grab the Debug signal. Taking the capture interface input message as an example, first click 1 in the figure below to set the trigger signal, then click 2 to set the trigger condition, after the setting is complete, click 3 to start triggering, and then the captured signal will appear at 4.



ILA Core Properties

hw_ila_1

Name: hw_ila_1

Cell: u_ila_0

Device: xc7z020_1

HW core: core_1

Capture sample count: 0 of 1024

Core status: Idle

General Properties

ILA Status: Idle

Name	Value
zynq_tte_i/um_0/pktin_data[133:0]	24000000000060f
zynq_tte_i/um_0/pktin_data_valid	0
zynq_tte_i/um_0/pktin_data_wr	0
zynq_tte_i/um_0/pktin_ready	1
zynq_tte_i/um_0/pktout_data[133:0]	000000000000000
zynq_tte_i/um_0/pktout_data_valid	0
zynq_tte_i/um_0/pktout_data_valid_wr	0
zynq_tte_i/um_0/pktout_data_wr	0

Updated at: 2019-Dec-18 10:04:14

Settings - hw_ila_1 Status - hw_il

Core status ● ○ ○ ○ ○ Idle

Capture status - Window 1 of 1

Window sample 0 of 1024

Trigger Setup - hw_ila_1 Capture Setup - hw_ila_1

Name	Operator	Radix	Value	Port	Compar
zynq_tte_i/um_0/pktin_data_wr	==	[B]	1	probe3[0]	1 of 1

Version management

version number	Modified by	date	Remark
1.0	Wang Yaoxiang	2019-12-17	initial version