“计算机组成与设计”课程设计

RISC-V CPU的流水线实现

龚奕利

yiligong@whu.edu.cn

2021年春季

实验目的

- 融会贯通计算机组成与设计课程所教授的知识，通过对知识的综合应用，加深对CPU系统各模块的工作原理及相互联系的认识。
- 学习采用EDA（Electronic Design Automation）技术设计RISC-V流水线CPU的技术与方法。
- 培养科学研究的独立工作能力，获得CPU设计、仿真、综合、实现、运行的实践和经验。
- 了解SOC系统，并在FPGA开发板上实现简单的SOC系统。

实验内容

- 用硬件描述语言（Verilog）设计RISC-V**流水线**CPU，支持RV32I中的大部分指令
- 用仿真软件Modelsim对有数据冒险和控制冒险的汇编程序进行仿真
- 用Vivado下载到N4 FPGA板上进行综合、实现和运行

RISC-V指令集

- 4个基本的ISA：
  - RV32I：32位地址空间整数指令
  - RV64I：64位地址空间整数指令
  - RV32E：RV32I的子集，更少的整数寄存器，用以支持微小控制器
  - RV128I：128位地址空间整数指令
- 还有很多extensions
  - Standard, reserved, and custom
  - E.g. the standard integer multiplication and division extension is named “M”
  - The standard atomic instruction extension, denoted by “A”
  - The standard single-precision floating-point extension, denoted by “F”
  - The standard double-precision floating-point extension, denoted by “D”
  - The standard “C” compressed instruction extension provides narrower 16-bit forms of common instructions

待实现指令

- I0={LUI, AUIPC}
- I1={JAL, JALR, BEQ, BNE, BLT, BGE, BLTU, BGEU}
- I2={LB, LH, LW, LBU, LHU, SB, SH, SW}
- I3={ADDI, SLTI, SLTIU, XORI, ANDI, ALLI, SRLI, SRALI, SRAI}
- I4={ADD, SUB, SLL, SLT, SLTU, XOR, SRL, SRA, OR, AND}
- See to RISC-V spec “Chapter 24 RV32/64G Instruction Set Listings”

考核方式

- 考勤：10%
- 课堂检查：10%
- CPU功能测试、实验报告及面试：80%

功能	内容	分值
ModelSim仿真测试通过	S1={sb, sh, sw, lb, lh, lw, lbu, lhu}	8
	S2={add， sub， xor, or, and, srl, sra, sll}	5
	S3={xori, ori, andi, srli, srai, slli}	8
	S4={slt, sltu, slti, sltiu}	8
	S5={jal, jalr}	8
	S6={beq, bne, blt, bge, bltu, bgeu}	8
	冒险检测与冲突解决	35
Vivado+N4 FPGA板测试通过	测试程序	20

提交方式

- 代码及实验报告
  - Due on 5月9日 24:00
  - 发送至coecswhu@163.com
  - 邮件标题必须为“2021SpringCOE-学号-姓名”
  - 代码压缩成zip文件(将ModelSim项目和Vivado项目（若下板子通过）)， 实验报告(pdf文件)不压缩，一起作为附件发送，文件名格式分别为“学号_姓名_modelsim实验代码.zip”， “学号_姓名_vivado实验代码.zip”和“学号_姓名_实验报告.pdf”。
  - 用你的学号和姓名替换文件名中的“学号”和“姓名”
- 面试
  - 面试内容包括但不限于代码调试、运行操作、结果展示、现场实现新增指令等
  - 时间待定

工具使用：ModelSim

- ModelSim教程
  - Modelsim详细使用教程（一看就会）.pdf
  - Modelsim精选入门教程.pdf
- 项目导入/打开
  - ModelSim使用的是绝对路径，而不是相对路径。提供的源文件如果直接放在C:盘下，可直接使用
  - 否则，要点击File->Change Directory，更改路径到你的目录位置，再打开工程文件
  - 推荐重新创建工程文件，减少不必要的麻烦
- 参见附录一和网盘文件夹中的解说视频

RISC-V编译器：Venus

- https://venus.cs61c.org/
- 汇编代码→RISC-V机器代码，仿真执行

RISC-V流水线CPU的设计与实现

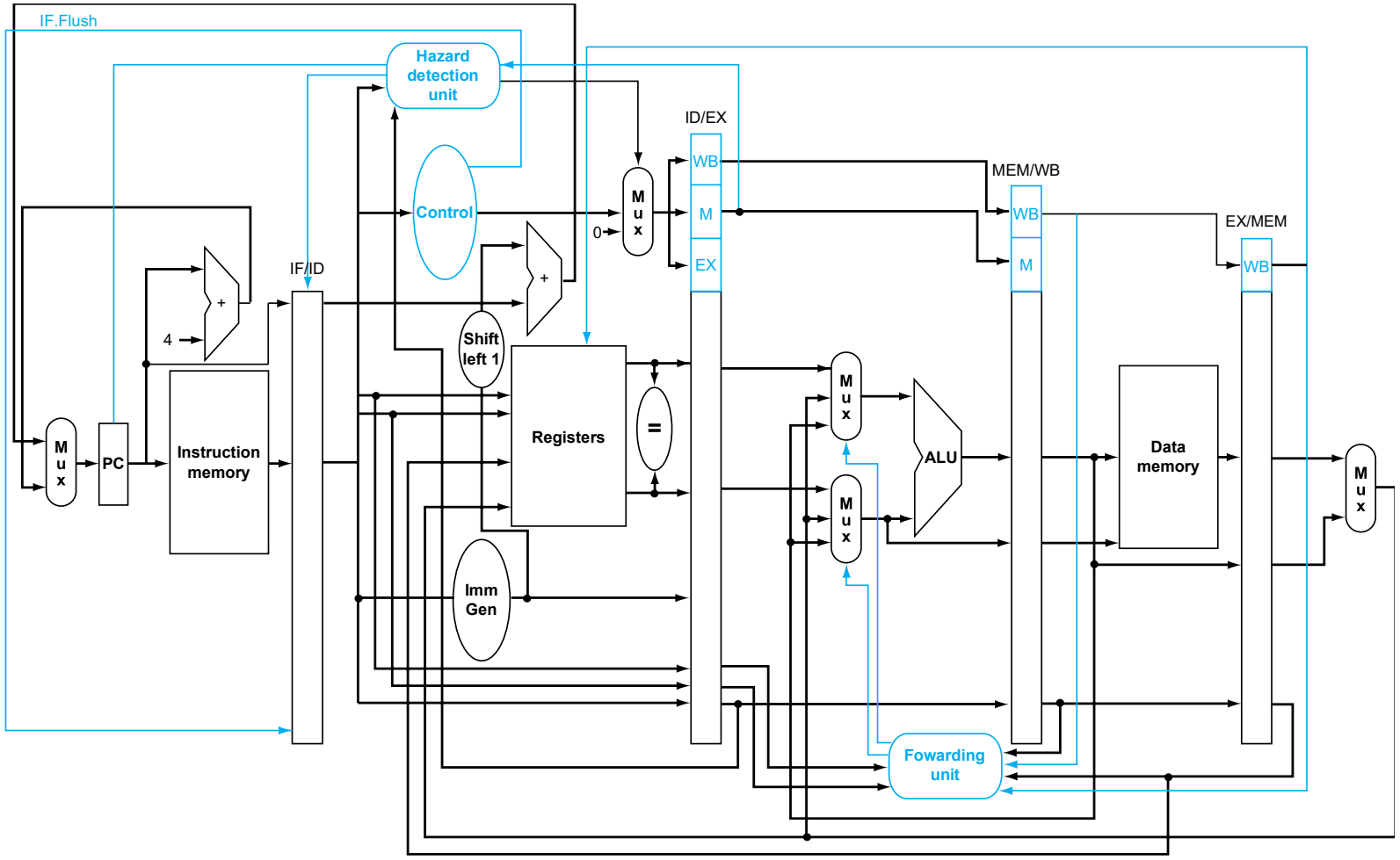


FIGURE 4.62 The final datapath and control for this chapter.
Note that this is a stylized figure rather than a detailed datapath, so it's missing the ALUsrc Mux from Figure 4.55 and the multiplexor controls from Figure 4.49.

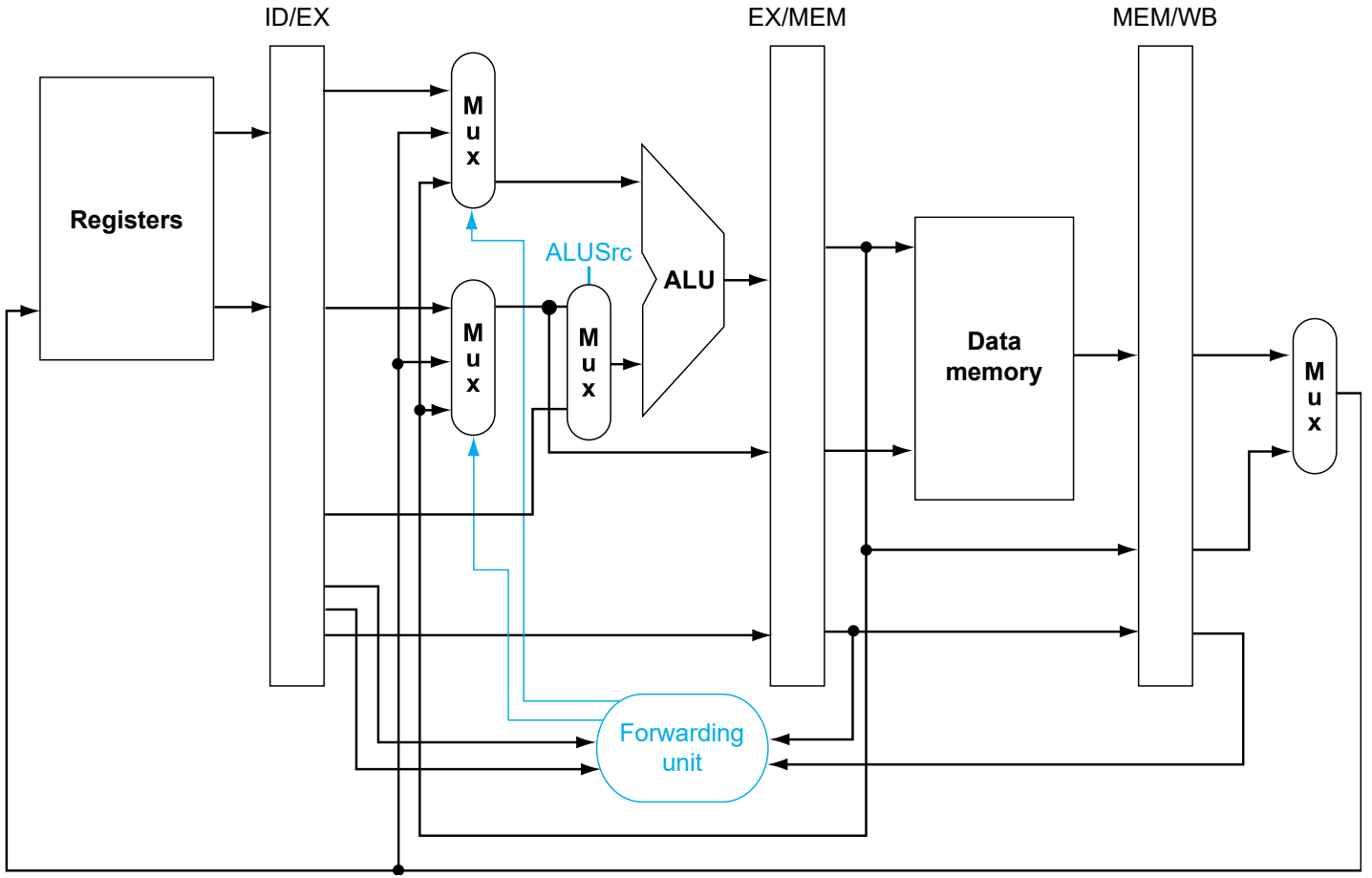


FIGURE 4.55 A close-up of the datapath in Figure 4.52 shows a 2:1 multiplexor, which has been added to select the signed immediate as an ALU input.

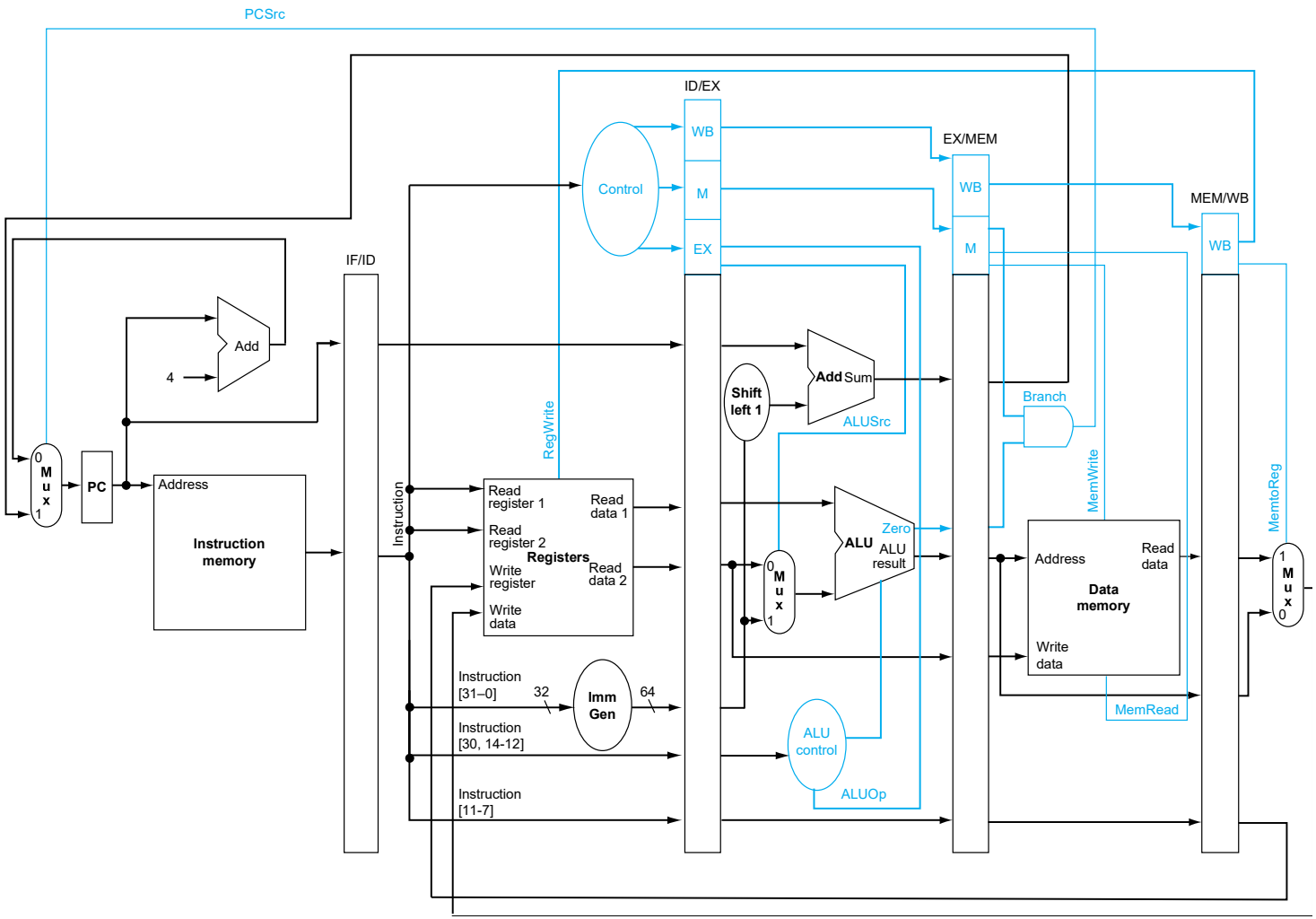
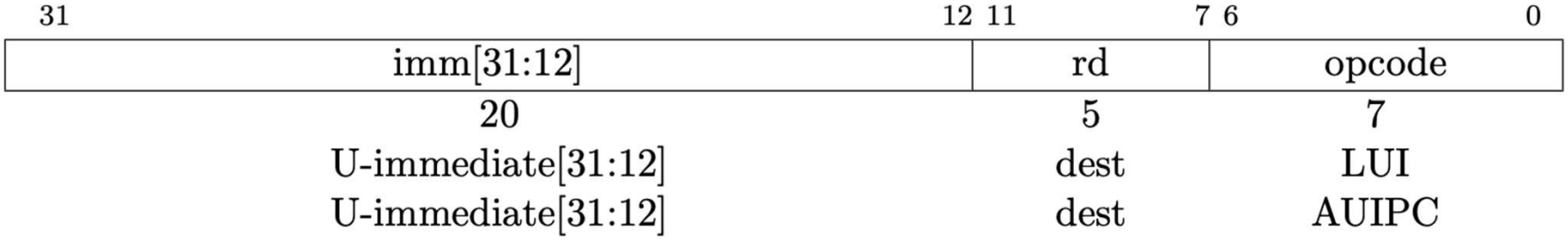


FIGURE 4.49 The pipelined datapath of Figure 4.44, with the control signals connected to the control portions of the pipeline registers.

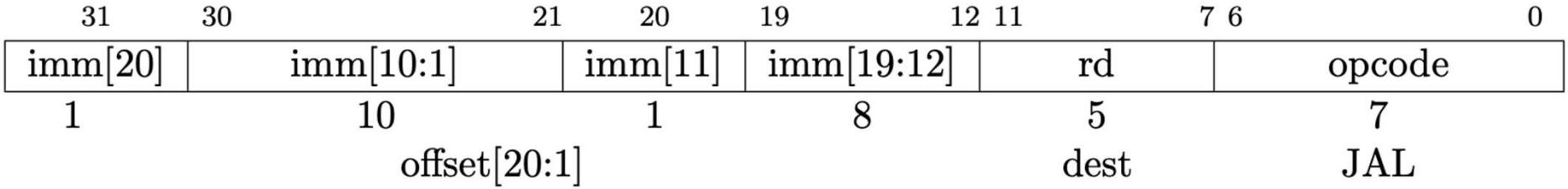
The control values for the last three stages are created during the instruction decode stage and then placed in the ID/EX pipeline register. The control lines for each pipe stage are used, and remaining control lines are then passed to the next pipeline stage.

LUI&AUIPC



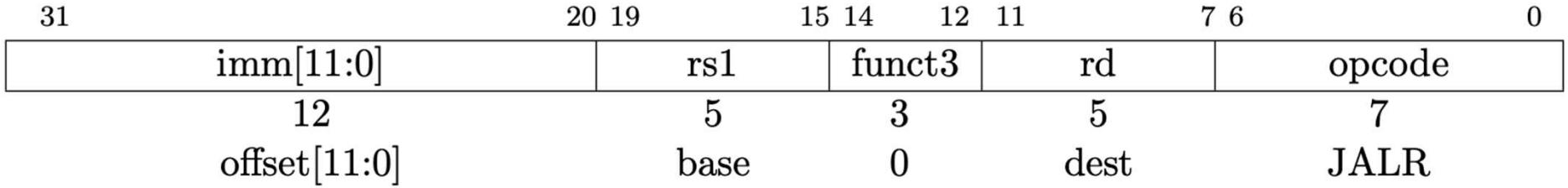
- U-type format
- LUI (load upper immediate) is used to build 32-bit constants and uses the U-type format. LUI places the 32-bit U-immediate value into the destination register rd, filling in the lowest 12 bits with zeros.
- IMM模块：20位立即数+12个0，ALU做加法，srca选0，srcb选immout
- AUIPC (add upper immediate to pc) is used to build pc-relative addresses and uses the U-type format. AUIPC forms a 32-bit offset from the U-immediate, filling in the lowest 12 bits with zeros, adds this offset to the address of the AUIPC instruction, then places the result in register rd.
- IMM模块：20位立即数+12个0，ALU做加法，srca选pc，srcb选immout

JAL



- J-type format
- The J-immediate encodes a signed offset in multiples of 2 bytes. The offset is sign-extended and added to the address of the jump instruction to form the jump target address.
- Stores the address of the instruction following the jump (pc+4) into register rd.
- IMM模块有符号扩展立即数，pcadder2做pc+immout左移1位后的结果，WB阶段pc+4被写入rd

JALR



- I-type format
- The target address is obtained by adding the sign-extended 12-bit I-immediate to the register rs1, then setting the least-significant bit of the result to zero.
- The address of the instruction following the jump (pc+4) is written to register rd.
- IMM模块有符号扩展立即数，pcadder2做rs1和immout的加法，然后结果最低位置0； WB阶段pc+4被写入rd

Conditional Branch

31	30 25	24 20	19 15	14 12	11 8	7	6 0
imm[12]	imm[10:5]	rs2	rs1	funct3	imm[4:1]	imm[11]	opcode
1	6	5	5	3	4	1	7
offset[12|10:5]		src2	src1	BEQ/BNE	offset[11|4:1]		BRANCH
offset[12|10:5]		src2	src1	BLT[U]	offset[11|4:1]		BRANCH
offset[12|10:5]		src2	src1	BGE[U]	offset[11|4:1]		BRANCH

- B-type format
- IMM模块有符号扩展立即数， pcadder2 做pc+immout左移1位后的结果
- srca选rs1，srcb选rs2，根据是否带u，cmp比较器相应地做（有/无符号数）减法，根据减法结果相应地设置zero，lt和ge信号

合并所有的指令行为

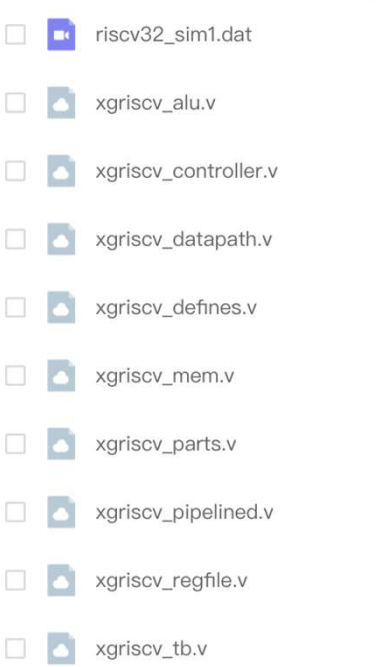
指令（类型）	IMM	srca	srcb	ALUctrl	MEM	WB	pc
load	itype	rs1	immout	+	read	memout→rd	pc+4
store	stype	rs1	immout	+	write	不写	pc+4
LUI	utype	0	immout	+	无	aluout→rd	pc+4
AUIPC	utype	pc	immout	+	无	aluout→rd	pc+4
JAL	jtype	pc+4	无关	MOVEA	无	pc+4→rd	pcadder2的计算结果
JALR	itype	pc+4	无关	MOVEA	无	pc+4→rd	pcadder2的计算结果
Branch	btype	无关	无关	无关	无	不写	根据cmp的比较结果决定选择pcadder2或者pc+4

pcadder2

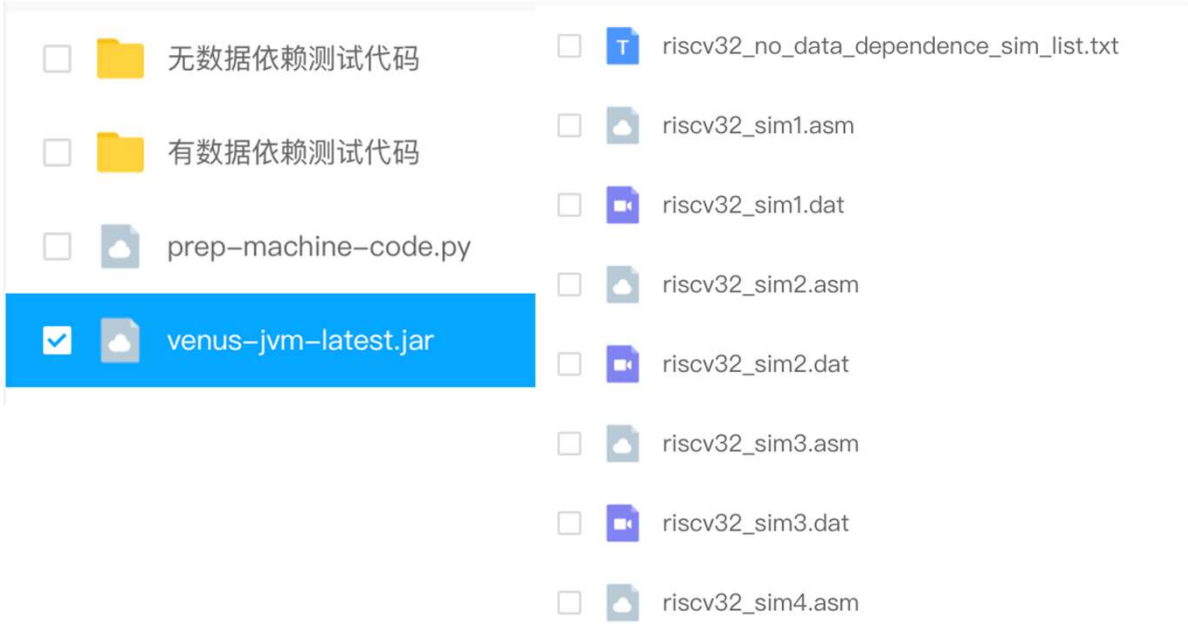
指令（类型）	srca	srcb	pcbranch（output）
JAL/Branch	pc	shifted immout	srca + srcb
JALR	rs1	immout	srca + srcb 最后一位置0

代码目录

- xgriscv-初始3指令版
- 已实现：lui，auipc，addi

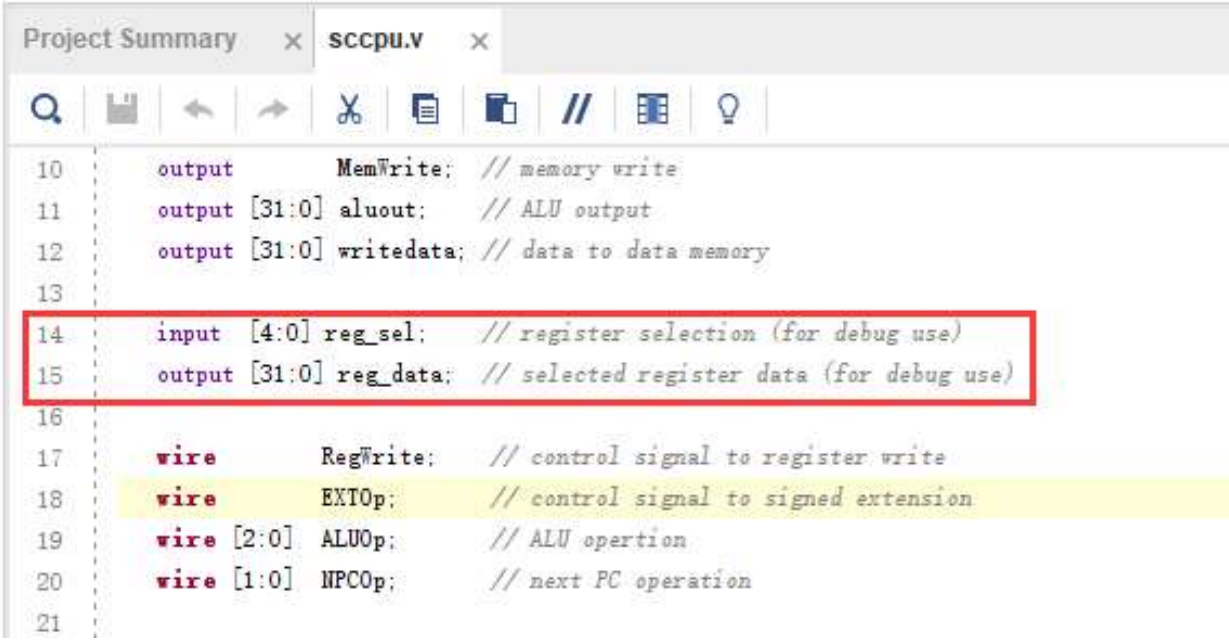


- riscv测试代码



从仿真代码到FPGA代码

1. 删除module imem的定义，后面会用系统提供的IP实现imem
2. 删除module xgriscv_pipeline的定义
3. 修改fpga_top.v和你的cpu实现：
   1. 在你的cpu文件中添加这几行：
      - Reg_sel：数码管显示寄存器编号
      - Reg_data：数码管显示寄存器数据
   2. 在fpga_top.v文件里把cpu模块名替换成你的cpu模块名



从仿真代码到FPGA代码

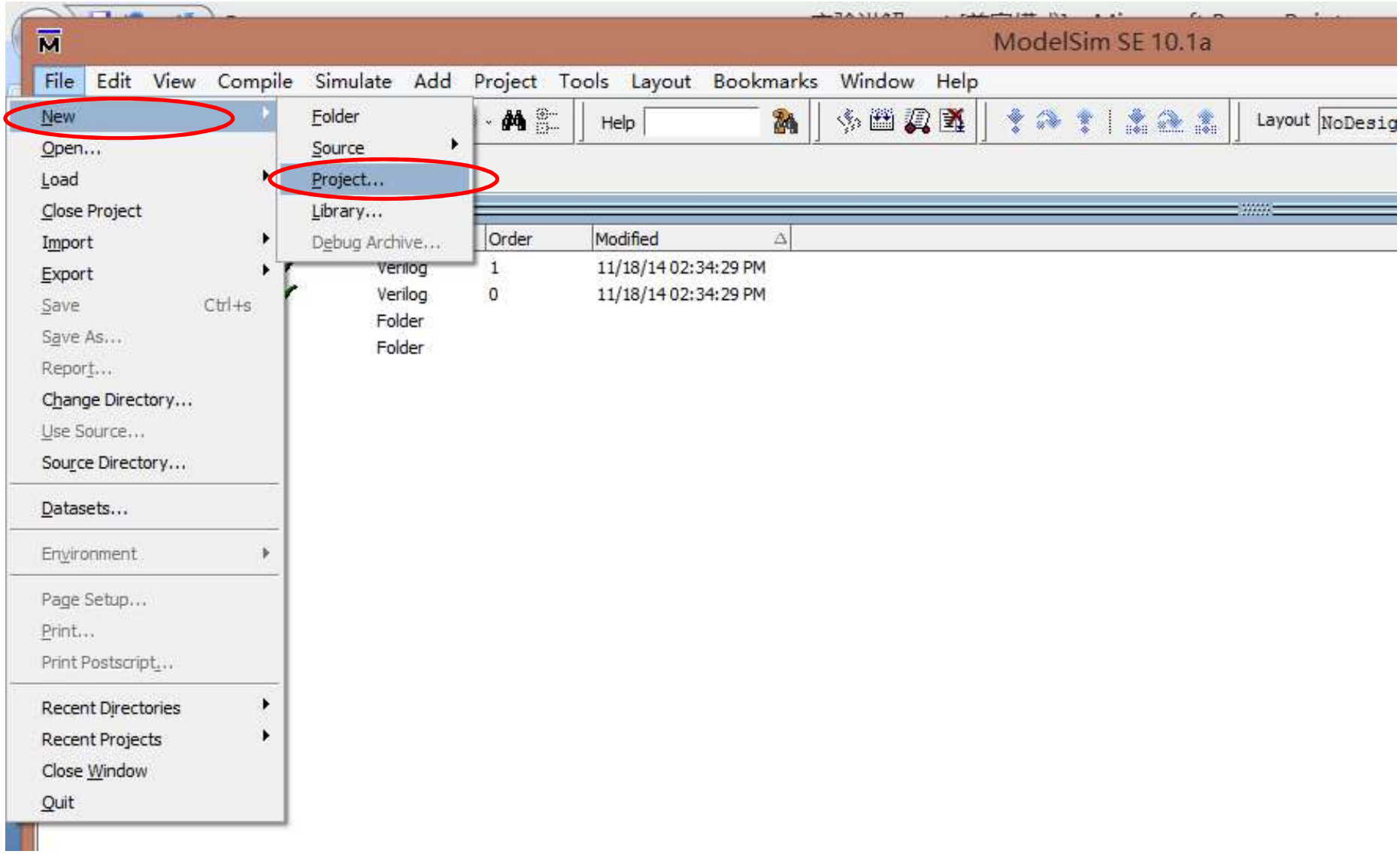
Project Summary | sccpu.v

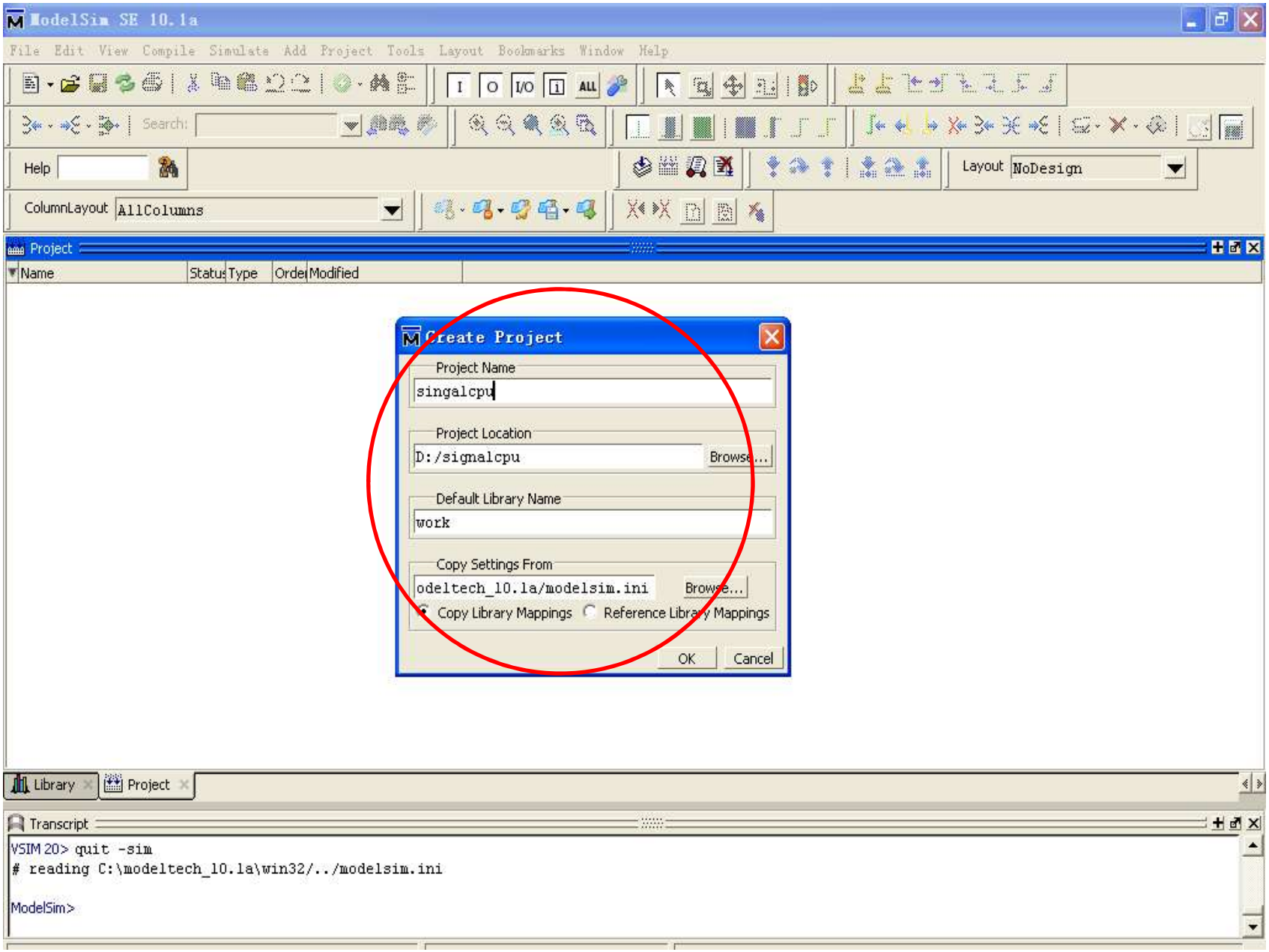
```
10    output        MemWrite;  // memory write
11    output [31:0] aluout;    // ALU output
12    output [31:0] writedata; // data to data memory
13
14    input  [4:0] reg_sel;    // register selection (for debug use)
15    output [31:0] reg_data;  // selected register data (for debug use)
16
17    wire        RegWrite;    // control signal to register write
18    wire        EXTOp;       // control signal to signed extension
19    wire [2:0]  ALUOp;       // ALU opertion
20    wire [1:0]  NPCOp;       // next PC operation
21
```

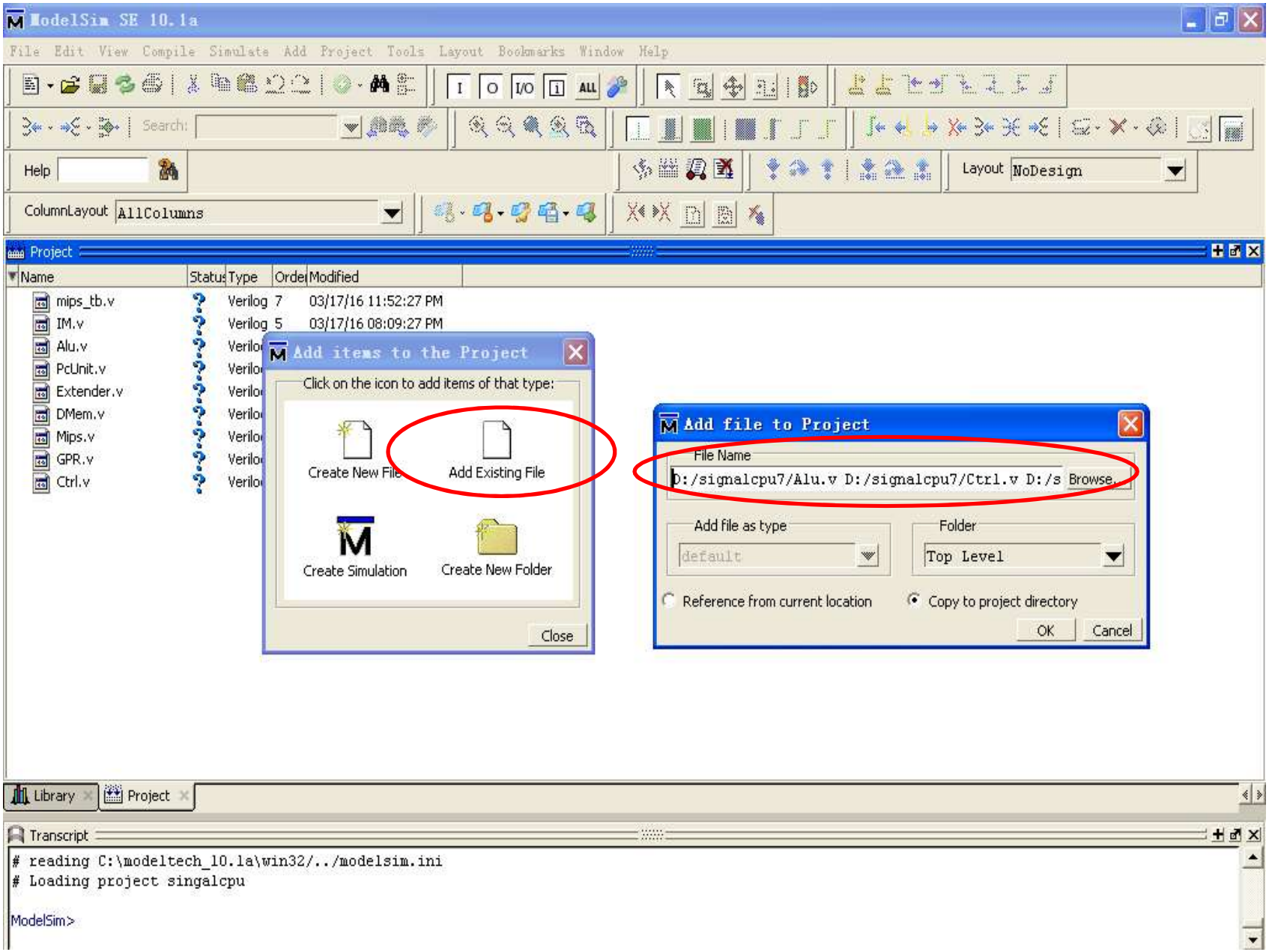
在你的cpu文件中添加这几行：
Reg_sel：数码管显示寄存器编号
Reg_data：数码管显示寄存器数据

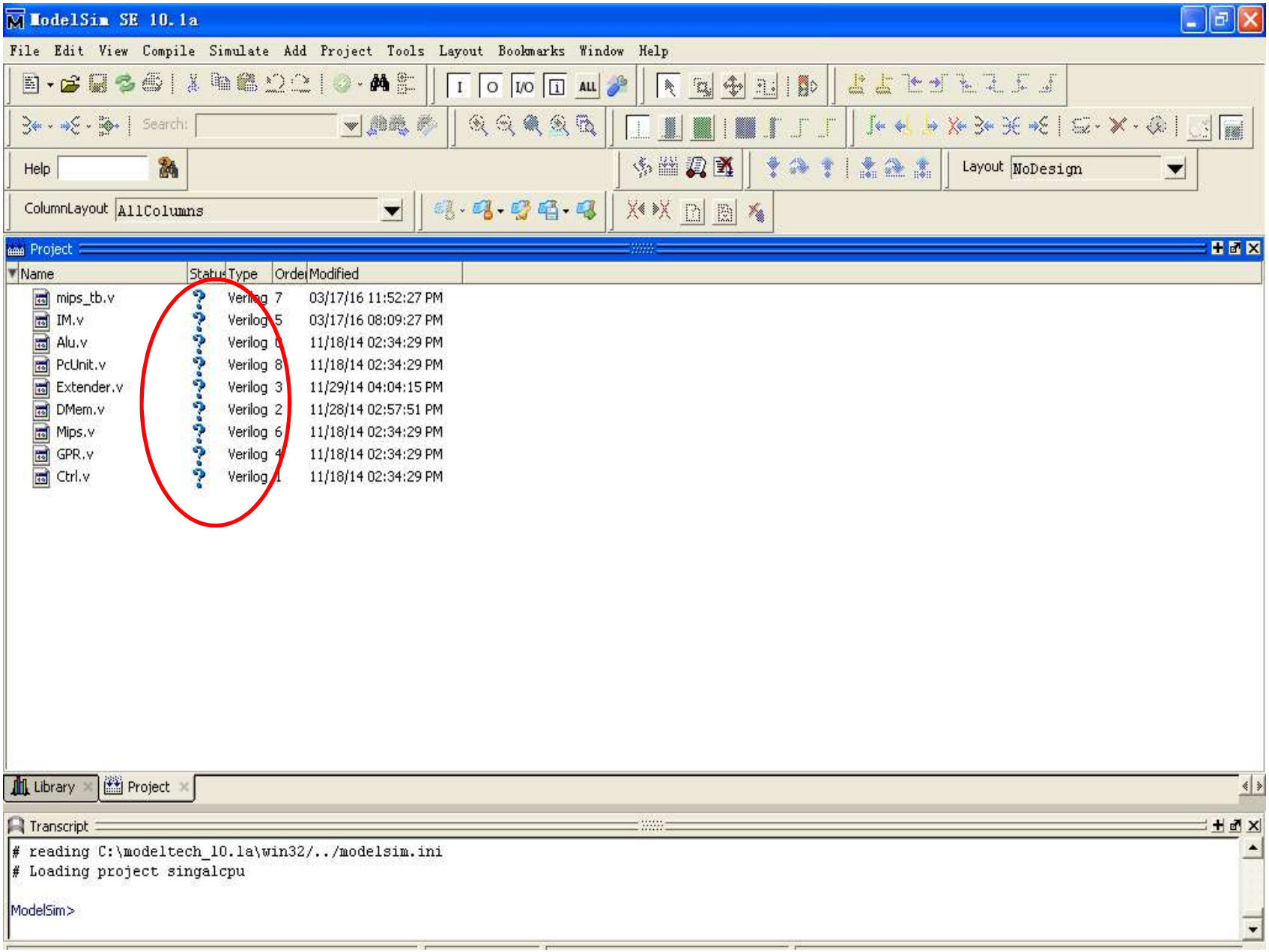
在fpga_top.v文件里把cpu模块名替换成你的cpu模块名

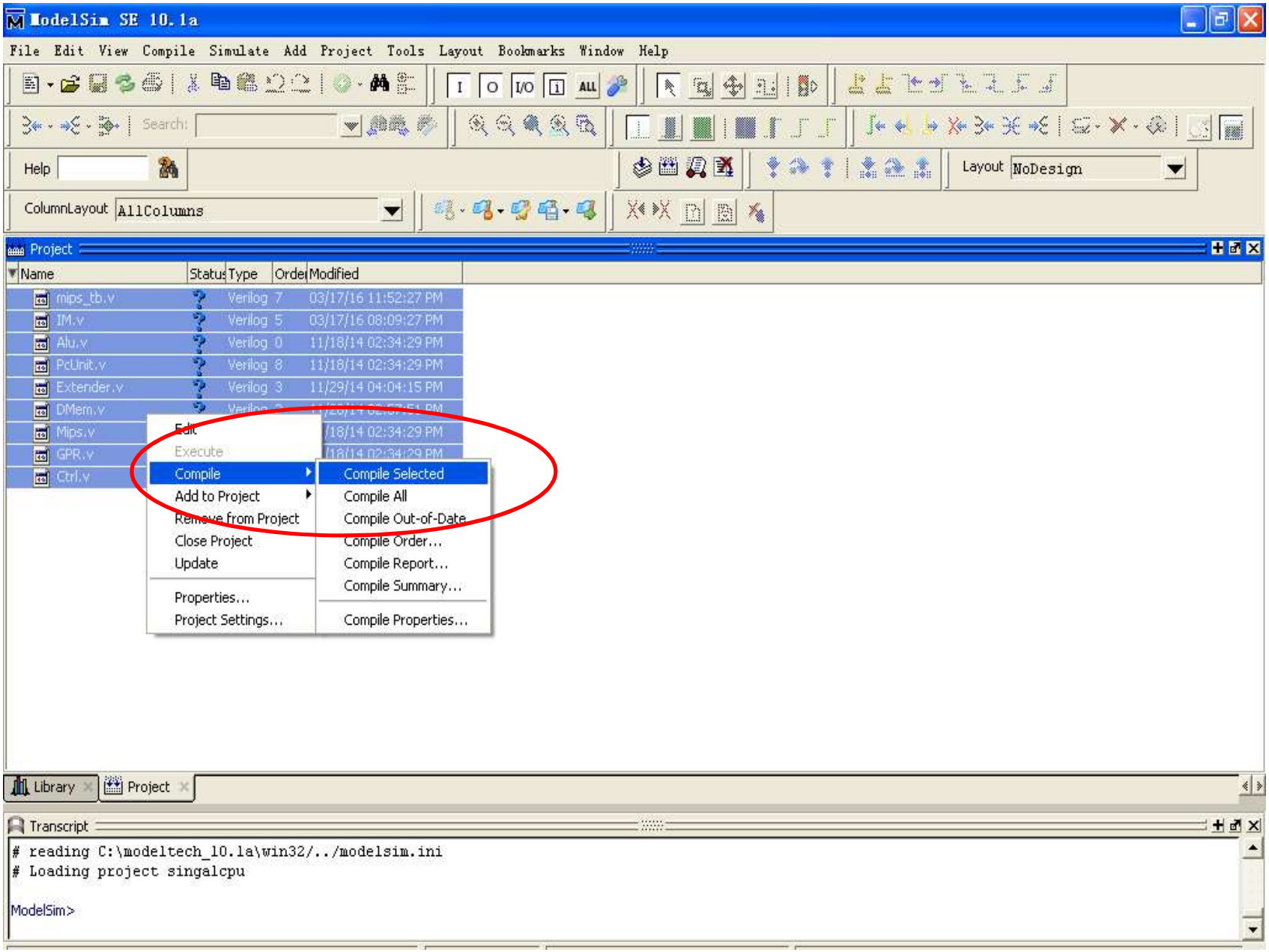
附录一：ModelSim使用

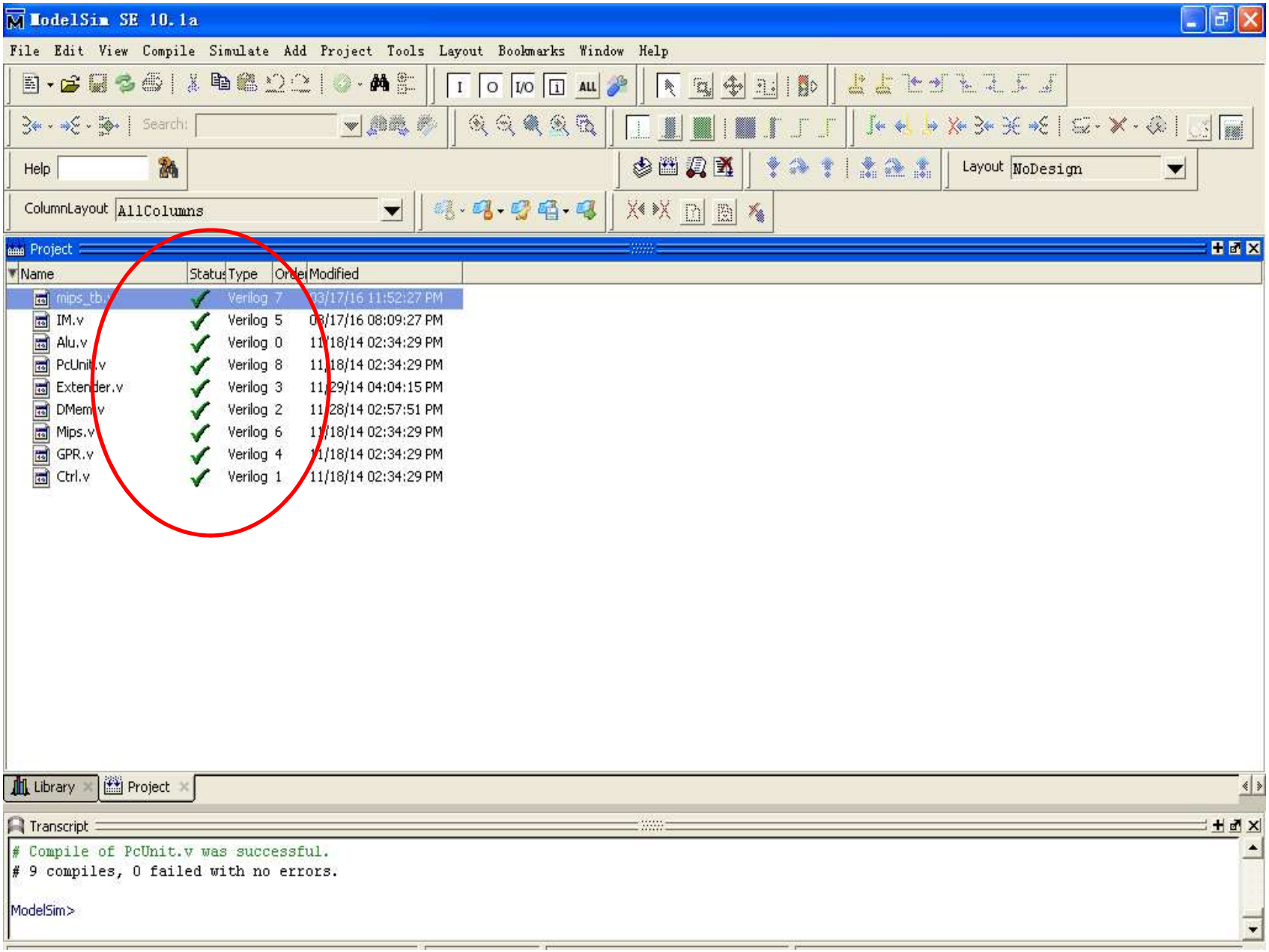

ModelSim SE 10.1a
File
Edit
View
Compile
Simulate
Add
Project
Tools
Layout
Bookmarks
Window
Help
New
Open...
Load
Close Project
Import
Export
Save Ctrl+s
Save As...
Report...
Change Directory...
Use Source...
Source Directory...
Datasets...
Environment
Page Setup...
Print...
Print Postscript...
Recent Directories
Recent Projects
Close Window
Quit
Folder
Source
Project...
Library...
Debug Archive...
Help
Layout
NoDesig
Order
Modified
Verilog 1 11/18/14 02:34:29 PM
Verilog 0 11/18/14 02:34:29 PM
Folder
Folder

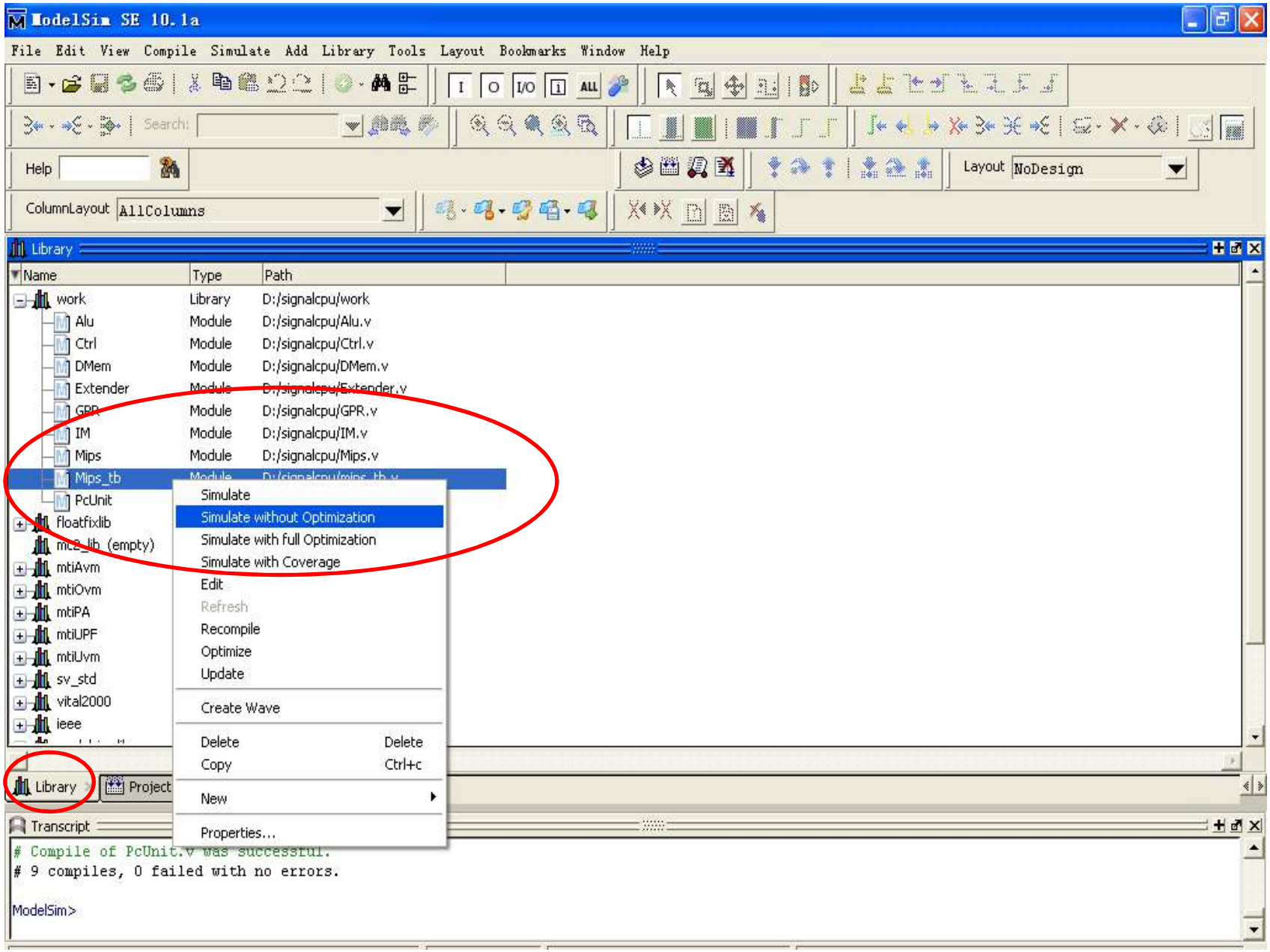

ModelSim SE 10.1a
File Edit View Compile Simulate Add Project Tools Layout Bookmarks Window Help
Search:
Help
Layout NoDesign
ColumnLayout AllColumns
Project
Name Status Type Order Modified
Create Project
Project Name
singalcpu
Project Location
D:/signalcpu
Browse...
Default Library Name
work
Copy Settings From
odeltech_10.1a/modelsim.ini
Browse...
Copy Library Mappings
Reference Library Mappings
OK
Cancel
Library
Project
Transcript
VSIM 20> quit -sim
reading C:\modeltech_10.1a\win32/../modelsim.ini
ModelSim>

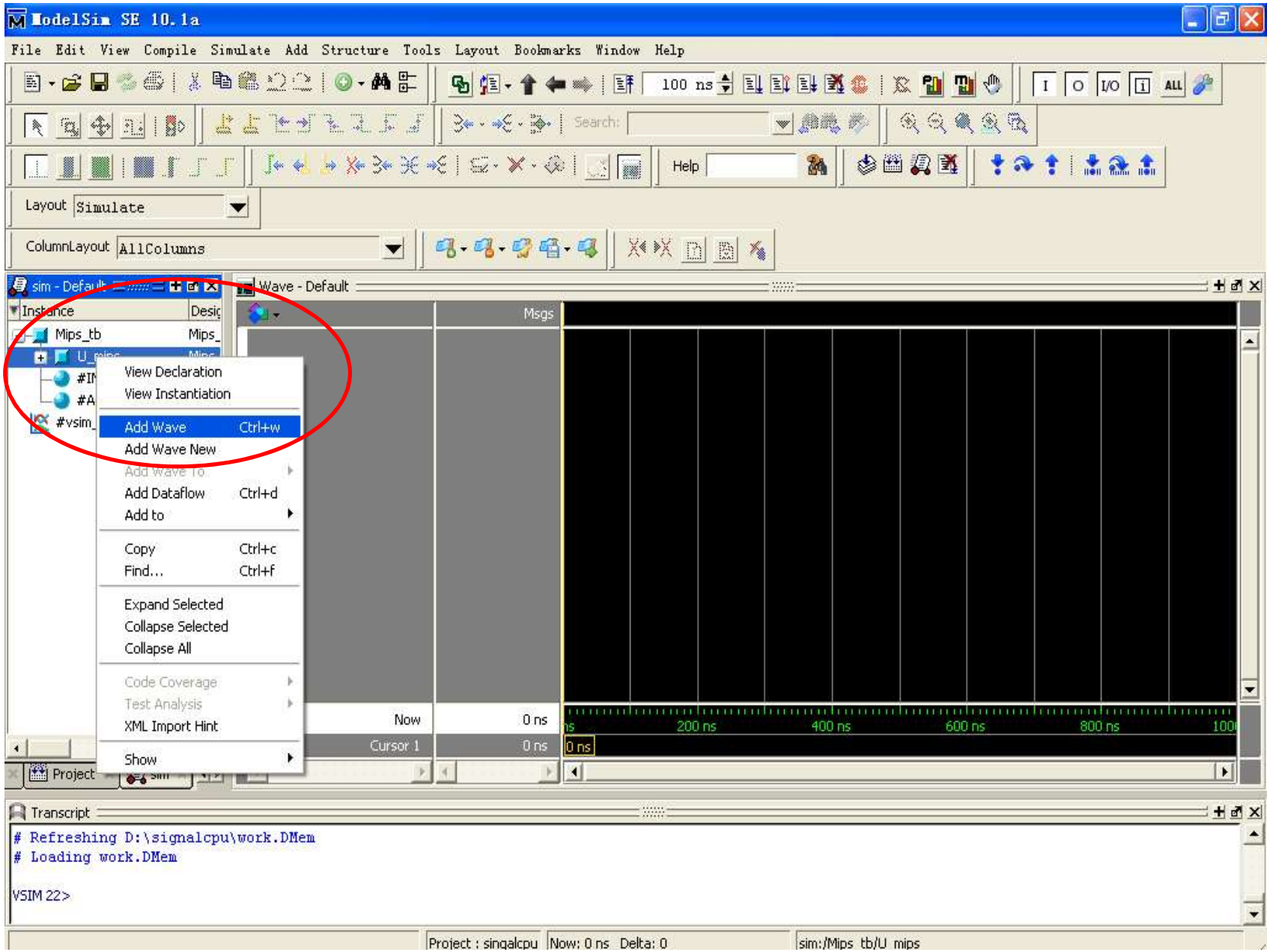

ModelSim SE 10.1a
File Edit View Compile Simulate Add Project Tools Layout Bookmarks Window Help
Search:
Help
Layout NoDesign
ColumnLayout AllColumns
Project
Name Status Type Order Modified
mips_tb.v Verilog 7 03/17/16 11:52:27 PM
IM.v Verilog 5 03/17/16 08:09:27 PM
Alu.v
PcUnit.v
Extender.v
DMem.v
Mips.v
GPR.v
Ctrl.v
Add items to the Project
Click on the icon to add items of that type:
Create New File
Add Existing File
Create Simulation
Create New Folder
Close
Add file to Project
File Name
D:/signalcpu7/Alu.v D:/signalcpu7/Ctrl.v D:/s
Browse...
Add file as type
default
Folder
Top Level
Reference from current location
Copy to project directory
OK
Cancel
Library
Project
Transcript
reading C:\modeltech_10.1a\win32/../modelsim.ini
Loading project singalcpu
ModelSim>

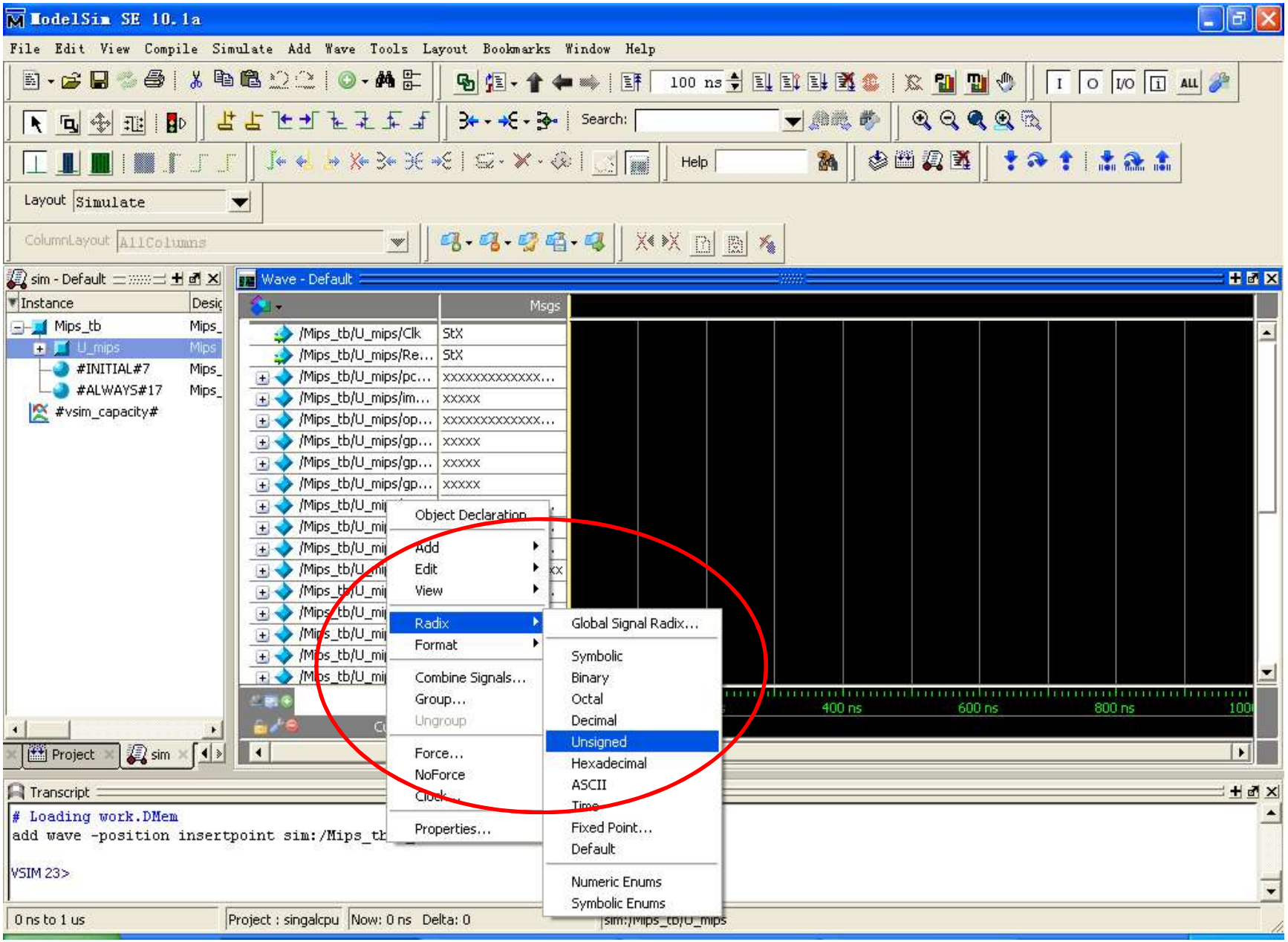

ModelSim SE 10.1a
File Edit View Compile Simulate Add Project Tools Layout Bookmarks Window Help
Search:
Help
Layout NoDesign
ColumnLayout AllColumns
Project
Name Status Type Order Modified
mips_tb.v Verilog 7 03/17/16 11:52:27 PM
IM.v Verilog 5 03/17/16 08:09:27 PM
Alu.v Verilog 11/18/14 02:34:29 PM
PcUnit.v Verilog 8 11/18/14 02:34:29 PM
Extender.v Verilog 3 11/29/14 04:04:15 PM
DMem.v Verilog 2 11/28/14 02:57:51 PM
Mips.v Verilog 6 11/18/14 02:34:29 PM
GPR.v Verilog 4 11/18/14 02:34:29 PM
Ctrl.v Verilog 1 11/18/14 02:34:29 PM
Library
Project
Transcript
reading C:\modeltech_10.1a\win32/../modelsim.ini
Loading project singalcpu
ModelSim>

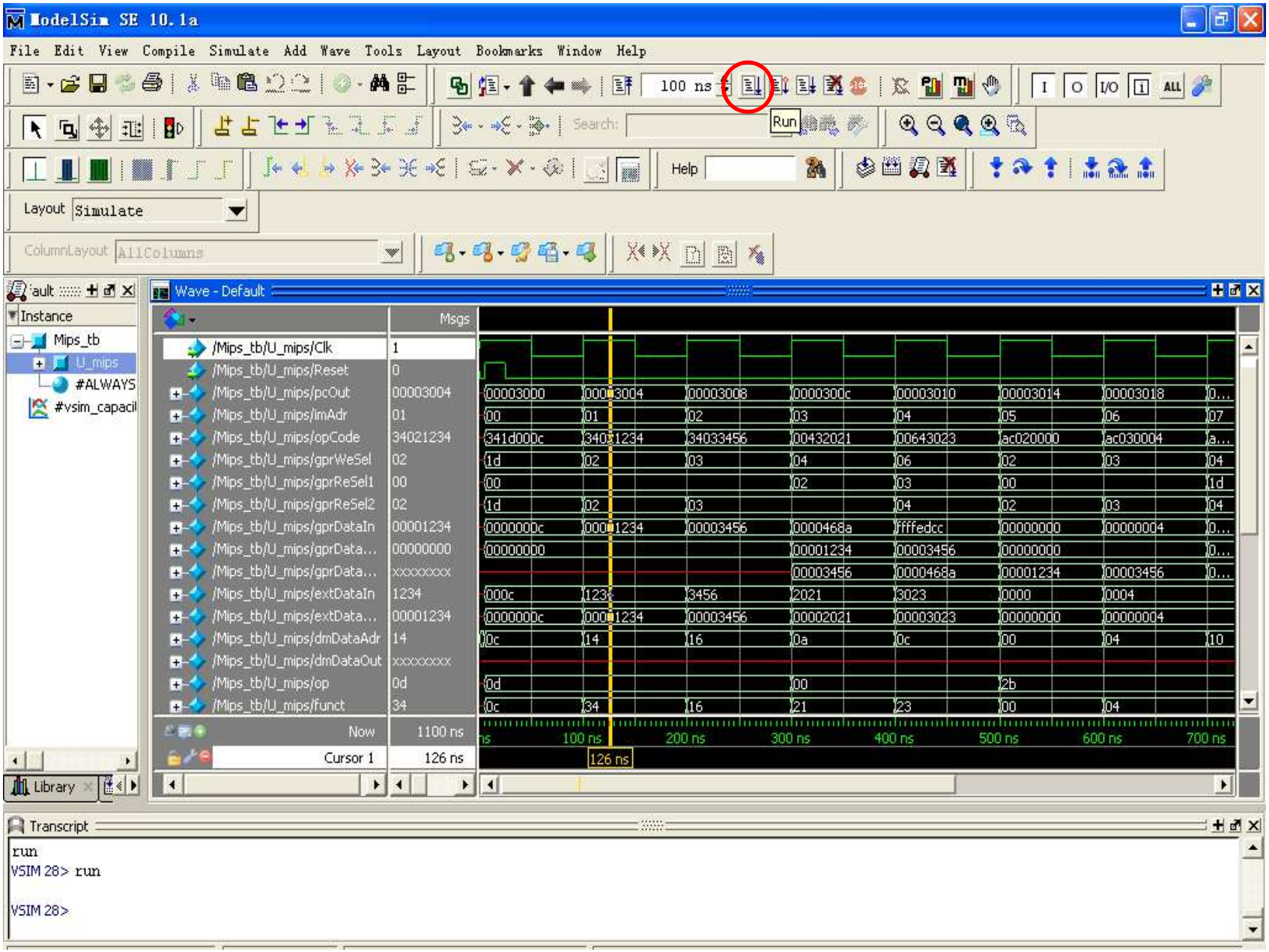

ModelSim SE 10.1a
File Edit View Compile Simulate Add Project Tools Layout Bookmarks Window Help
Search:
Help
Layout NoDesign
ColumnLayout AllColumns
Project
Name Status Type Order Modified
mips_tb.v Verilog 7 03/17/16 11:52:27 PM
IM.v Verilog 5 03/17/16 08:09:27 PM
Alu.v Verilog 0 11/18/14 02:34:29 PM
PcUnit.v Verilog 8 11/18/14 02:34:29 PM
Extender.v Verilog 3 11/29/14 04:04:15 PM
DMem.v
Mips.v
GPR.v
Ctrl.v
Edit
Execute
Compile
Add to Project
Remove from Project
Close Project
Update
Properties...
Project Settings...
Compile Selected
Compile All
Compile Out-of-Date
Compile Order...
Compile Report...
Compile Summary...
Compile Properties...
Library
Project
Transcript
reading C:\modeltech_10.1a\win32/../modelsim.ini
Loading project singalcpu
ModelSim>

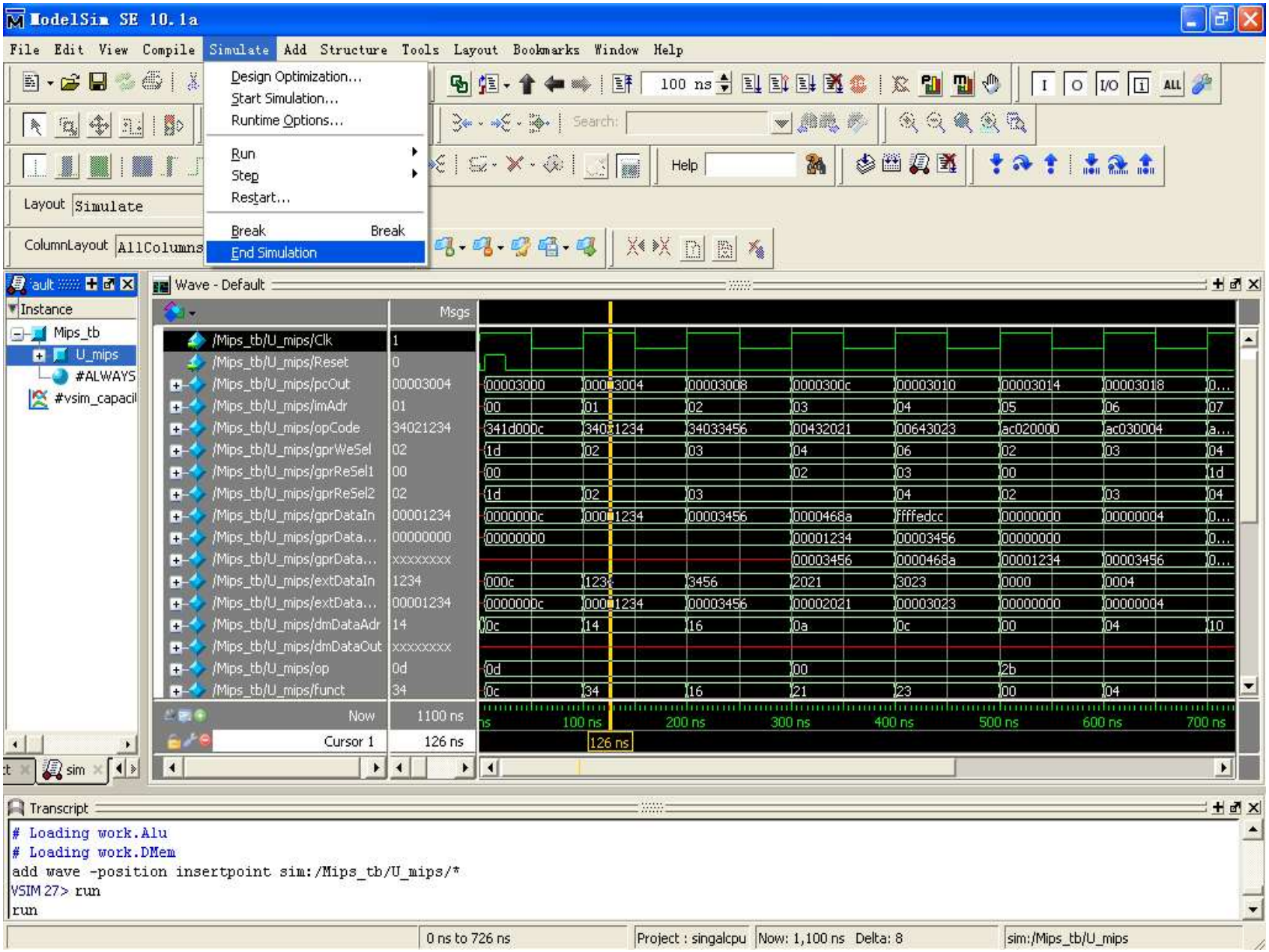

ModelSim SE 10.1a
File Edit View Compile Simulate Add Project Tools Layout Bookmarks Window Help
Search:
Help
Layout NoDesign
ColumnLayout AllColumns
Project
Name Status Type Order Modified
mips_tb.v Verilog 7 03/17/16 11:52:27 PM
IM.v Verilog 5 03/17/16 08:09:27 PM
Alu.v Verilog 0 11/18/14 02:34:29 PM
PcUnit.v Verilog 8 11/18/14 02:34:29 PM
Extender.v Verilog 3 11/29/14 04:04:15 PM
DMem.v Verilog 2 11/28/14 02:57:51 PM
Mips.v Verilog 6 11/18/14 02:34:29 PM
GPR.v Verilog 4 11/18/14 02:34:29 PM
Ctrl.v Verilog 1 11/18/14 02:34:29 PM
Library
Project
Transcript
Compile of PcUnit.v was successful.
9 compiles, 0 failed with no errors.
ModelSim>


ModelSim SE 10.1a
File Edit View Compile Simulate Add Library Tools Layout Bookmarks Window Help
Search:
Help
Layout NoDesign
ColumnLayout AllColumns
Library
Name Type Path
work Library D:/signalcpu/work
Alu Module D:/signalcpu/Alu.v
Ctrl Module D:/signalcpu/Ctrl.v
DMem Module D:/signalcpu/DMem.v
Extender Module D:/signalcpu/Extender.v
GPR Module D:/signalcpu/GPR.v
IM Module D:/signalcpu/IM.v
Mips Module D:/signalcpu/Mips.v
Mips_tb Module
PcUnit
floatfixlib
mc2_lib (empty)
mtiAvm
mtiOvm
mtiPA
mtiUPF
mtiUvm
sv_std
vital2000
ieee
Simulate
Simulate without Optimization
Simulate with full Optimization
Simulate with Coverage
Edit
Refresh
Recompile
Optimize
Update
Create Wave
Delete Delete
Copy Ctrl+c
New
Properties...
Library
Project
Transcript
Compile of PcUnit.v was successful.
9 compiles, 0 failed with no errors.
ModelSim>


ModelSim SE 10.1a
File Edit View Compile Simulate Add Structure Tools Layout Bookmarks Window Help
Layout Simulate
ColumnLayout AllColumns
sim - Default
Instance
Mips_tb
View Declaration
View Instantiation
Add Wave Ctrl+w
Add Wave New
Add Dataflow Ctrl+d
Add to
Copy Ctrl+c
Find... Ctrl+f
Expand Selected
Collapse Selected
Collapse All
Code Coverage
Test Analysis
XML Import Hint
Show
Wave - Default
Msgs
Now 0 ns
Cursor 1 0 ns
200 ns
400 ns
600 ns
800 ns
Transcript
Refreshing D:\signalcpu\work.DMem
Loading work.DMem
VSIM 22>
Project : singalcpu
Now: 0 ns Delta: 0
sim:/Mips_tb/U_mips


ModelSim SE 10.1a
File Edit View Compile Simulate Add Wave Tools Layout Bookmarks Window Help
Search:
Help
Layout Simulate
ColumnLayout AllColumns
sim - Default
Instance
Mips_tb
U_mips
#INITIAL#7
#ALWAYS#17
#vsim_capacity#
Wave - Default
Msgs
/Mips_tb/U_mips/Clk
/Mips_tb/U_mips/Re...
/Mips_tb/U_mips/pc...
/Mips_tb/U_mips/im...
/Mips_tb/U_mips/op...
/Mips_tb/U_mips/gp...
Object Declaration...
Add
Edit
View
Radix
Format
Combine Signals...
Group...
Ungroup
Force...
NoForce
Properties...
Global Signal Radix...
Symbolic
Binary
Octal
Decimal
Unsigned
Hexadecimal
ASCII
Time
Fixed Point...
Default
Numeric Enums
Symbolic Enums
400 ns
600 ns
800 ns
Project
sim
Transcript
Loading work.DMem
add wave -position insertpoint sim:/Mips_tb
VSIM 23>
0 ns to 1 us
Project : singalcpu
Now: 0 ns Delta: 0
sim:/Mips_tb/U_mips


ModelSim SE 10.1a
File Edit View Compile Simulate Add Wave Tools Layout Bookmarks Window Help
100 ns
Run
Search:
Help
Layout Simulate
ColumnLayout AllColumns
Wave - Default
Instance
Mips_tb
U_mips
#ALWAYS
#vsim_capacil
Msgs
/Mips_tb/U_mips/Clk 1
/Mips_tb/U_mips/Reset 0
/Mips_tb/U_mips/pcOut 00003004
/Mips_tb/U_mips/imAdr 01
/Mips_tb/U_mips/opCode 34021234
/Mips_tb/U_mips/gprWeSel 02
/Mips_tb/U_mips/gprReSel1 00
/Mips_tb/U_mips/gprReSel2 02
/Mips_tb/U_mips/gprDataIn 00001234
/Mips_tb/U_mips/gprData... 00000000
/Mips_tb/U_mips/gprData... xxxxxxxx
/Mips_tb/U_mips/extDataIn 1234
/Mips_tb/U_mips/extData... 00001234
/Mips_tb/U_mips/dmDataAdr 14
/Mips_tb/U_mips/dmDataOut xxxxxxxx
/Mips_tb/U_mips/op 0d
/Mips_tb/U_mips/funct 34
Now 1100 ns
Cursor 1 126 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns
126 ns
Library
Transcript
run
VSIM 28> run
VSIM 28>


ModelSim SE 10.1a
File Edit View Compile Simulate Add Structure Tools Layout Bookmarks Window Help
Design Optimization...
Start Simulation...
Runtime Options...
Run
Step
Restart...
Break Break
End Simulation
Layout Simulate
ColumnLayout AllColumns
Wave - Default
Instance
Mips_tb
U_mips
#ALWAYS
#vsim_capacil
Msgs
/Mips_tb/U_mips/Clk
/Mips_tb/U_mips/Reset
/Mips_tb/U_mips/pcOut
/Mips_tb/U_mips/imAdr
/Mips_tb/U_mips/opCode
/Mips_tb/U_mips/gprWeSel
/Mips_tb/U_mips/gprReSel1
/Mips_tb/U_mips/gprReSel2
/Mips_tb/U_mips/gprDataIn
/Mips_tb/U_mips/gprData...
/Mips_tb/U_mips/gprData...
/Mips_tb/U_mips/extDataIn
/Mips_tb/U_mips/extData...
/Mips_tb/U_mips/dmDataAdr
/Mips_tb/U_mips/dmDataOut
/Mips_tb/U_mips/op
/Mips_tb/U_mips/funct
Now 1100 ns
Cursor 1 126 ns
100 ns 200 ns 300 ns 400 ns 500 ns 600 ns 700 ns
126 ns
Transcript
Loading work.Alu
Loading work.DMem
add wave -position insertpoint sim:/Mips_tb/U_mips/*
VSIM 27> run
run
0 ns to 726 ns
Project : singalcpu
Now: 1,100 ns Delta: 8
sim:/Mips_tb/U_mips

附录二：vivado使用

FPGA基本概念

- 全称：Filed Programmable Gate Device，现场可编程逻辑器件
- 架构：基于内嵌SRAM工艺的查找表加触发器架构
- 组成：可编程I/O、可编程逻辑单元、内嵌RAM单元、时钟管理单元、DSP单元、内嵌硬核等

主要产品线：Virtex系列、Zynq系列、Kintex系列、Artix系列和Spartan系列。

主要产品线：Stratix系列、Arria系列、Cyclone系列和MAX系列。已被Intel收购。

FPGA常用术语

- IP核（Intellectual Property）

  即知识产权，是一段具有特定电路功能的硬件描述语言程序

- 软核（Soft Core）

  用HDL语言建立的数字系统模型

- 硬核（Hard Core）

  在芯片内部已完成特定布局布线、具备特定功能的数字系统模型

- 网表（netlist）

  是一个电路的雏形、电路之间硬件的连接形式

- 综合（Synthesis）

  是在所给的标准单元库和设计约束的前提下，将对电路的HDL高级语言描述，转化成优化过的门级网表的处理过程

FPGA开发工具

- 编程语言

Verilog或VHDL硬件逻辑描述语言

- 开发工具

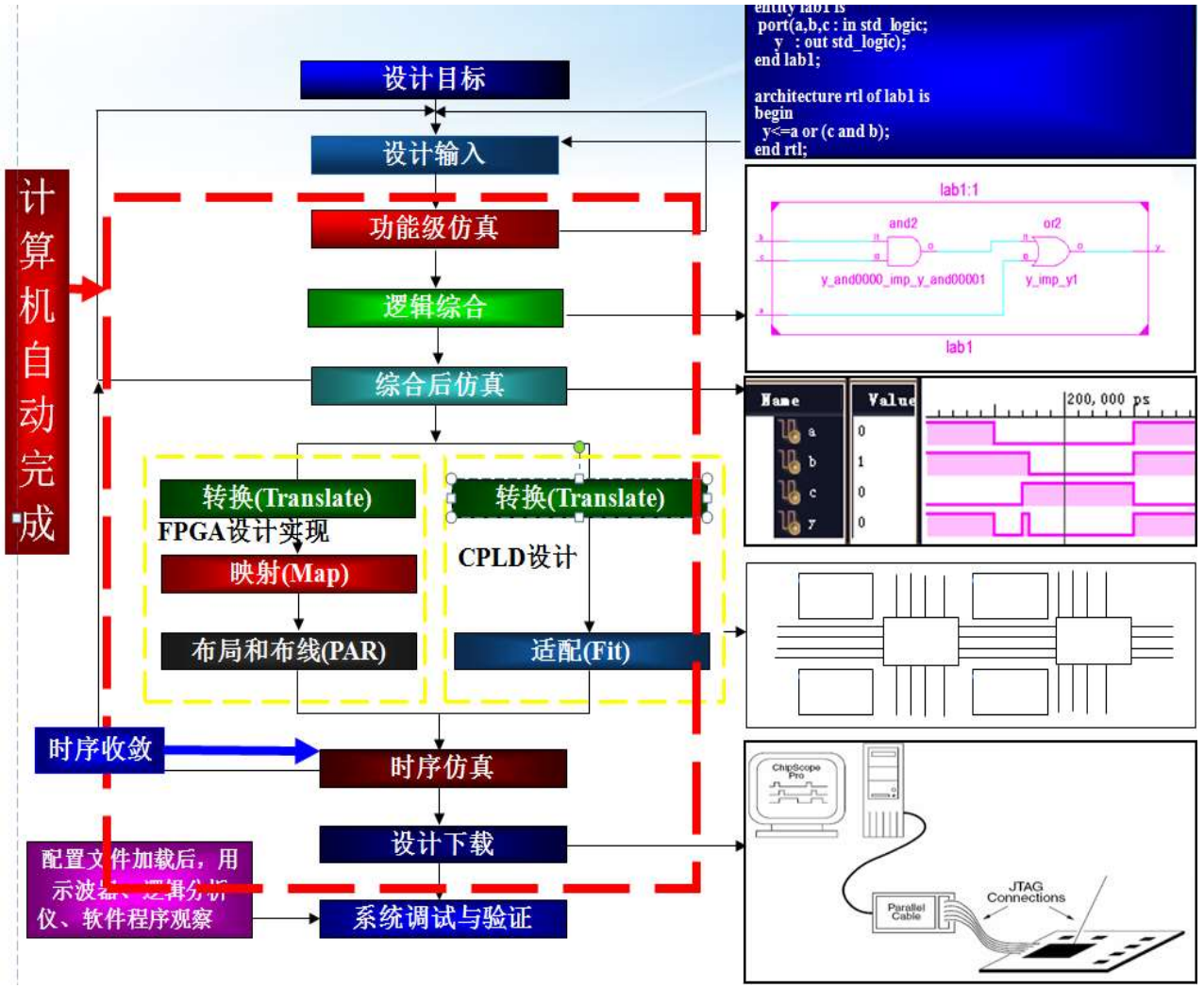
VIVADO 2017.1

- 开发板卡

NEXYS4 DDR

- 芯片型号

XC7A100TCSG324-1



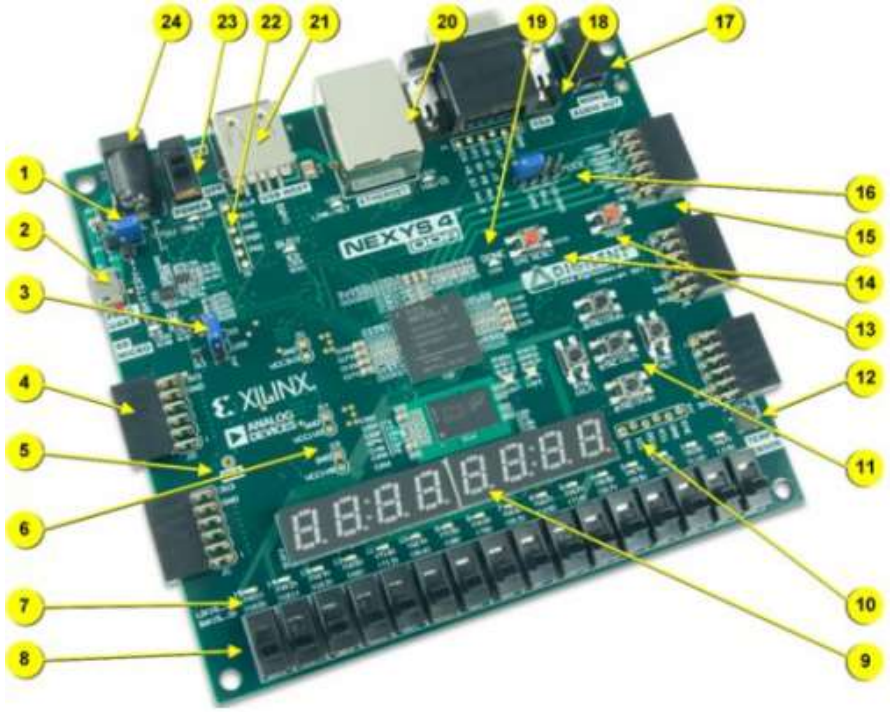
Nexys 4 DDR 硬件

- Nexys4-DDR采用了Xilinx Artix-7 FPGA芯片，它是一款简单易用的数字电路开发平台，可以支持在课堂环境中来设计一些行业应用。

- 大规模、高容量的FPGA，大容量外部存储，各种USB、以太网、以及其它接口，让Nexys4-DDR 能够满足从入门级组合逻辑电路到强大的嵌入式系统的设计。

- 板上集成的加速度、温度传感器、MEMs数字麦克风、扬声器放大器以及大量的I/O设备，让Nexys4-DDR不需要增添额外组件而用于各种各样的设计。

Nexys 4 DDR 示意图



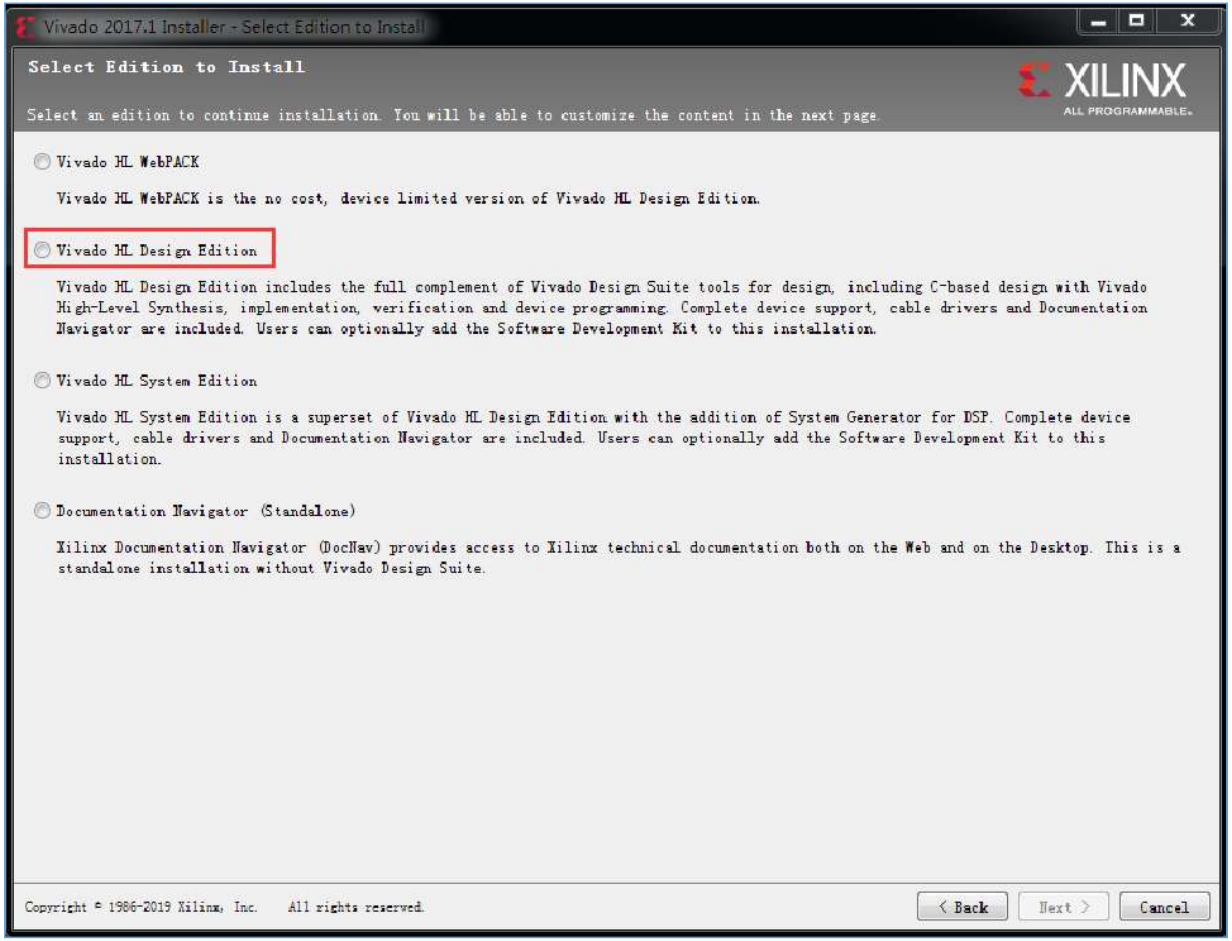
序号	描述	序号	描述
1	选择供电跳线	13	FPGA 配置复位按键
2	UART/ JTAG 共用 USB 接口	14	CPU 复位按键 (用于软核)
3	外部配置跳线柱(SD / USB)	15	模拟信号 Pmod 端口(XADC)
4	Pmod 端口	16	编程模式跳线柱
5	扩音器	17	音频连接口
6	电源测试点	18	VGA 连接口
7	16 个 LED	19	FPGA 编程完成 LED
8	16 个拨键开关	20	以太网连接口
9	8 位 7 段数码管	21	USB 连接口
10	可选用于外部接线的 JTAG 端口	22	(工业用) PIC24 编程端口
11	5 个按键开关	23	电源开关
12	板载温度传感器	24	电源接口

vivado软件

- vivado设计套件，是FPGA厂商赛灵思（xilinx）公司2012年发布的集成设计环境。
- 包括高度集成的设计环境和新一代从系统到IC级的工具，这些均建立在共享的可扩展数据模型和通用调试环境基础上。
- 基于AMBA AXI4 互联规范、IP-XACT IP封装元数据、工具命令语言(TCL)、Synopsys 系统约束(SDC) 以及其它有助于根据客户需求量身定制设计流程并符合业界标准的开放式环境。
- 赛灵思构建的vivado 工具把各类可编程技术结合在一起，能够扩展多达1 亿个等效ASIC 门的设计。

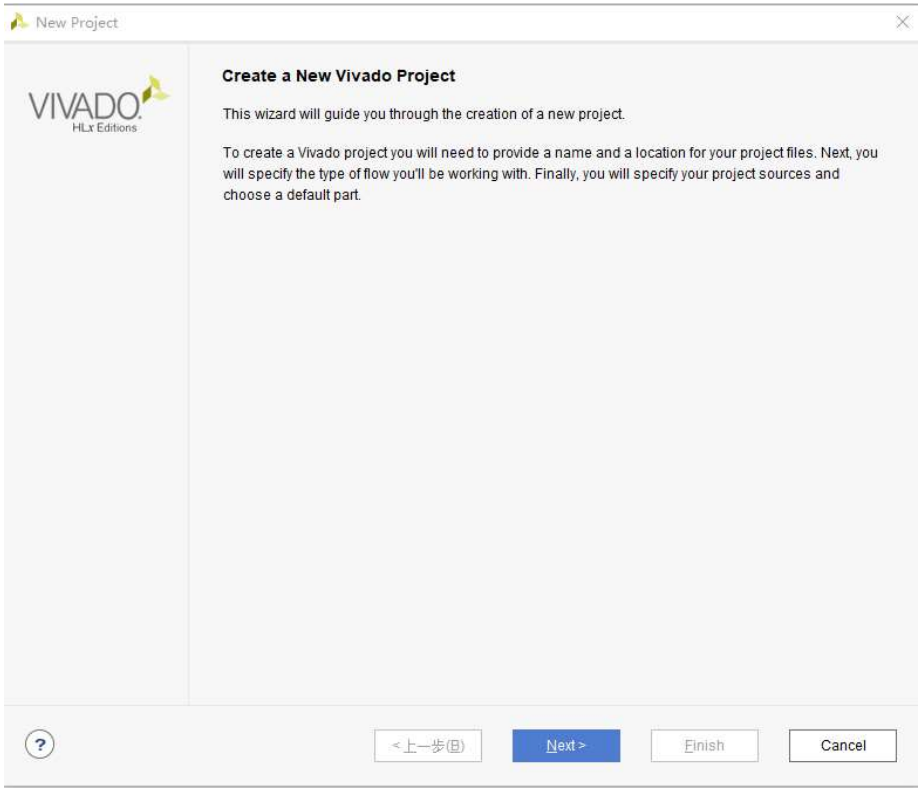
Vivado的安装

- 安装路径不能包含中文
- 选择第二项Vivado HL Design Edition



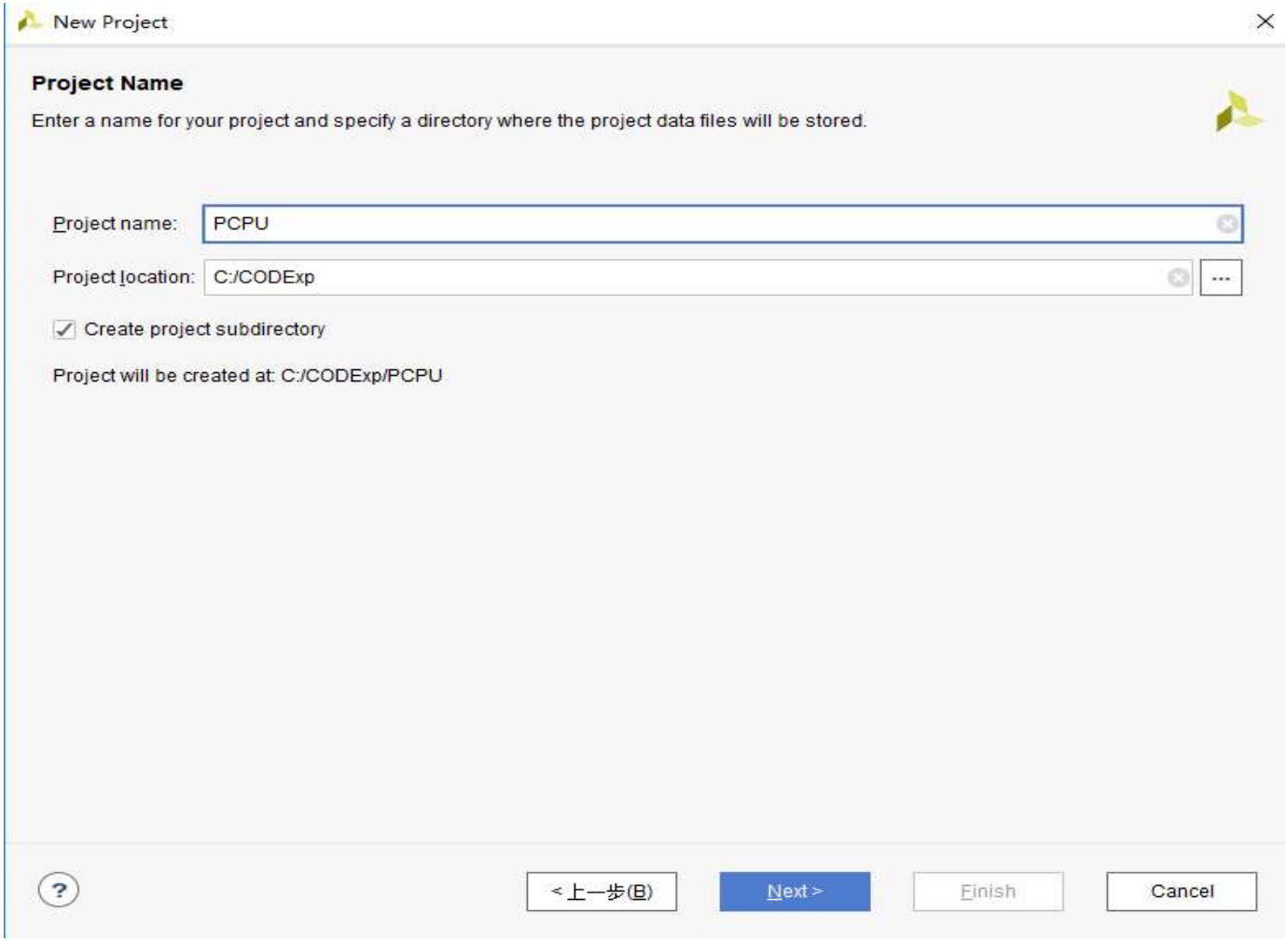
新建工程

- 在Vivado主界面中，File->New Project

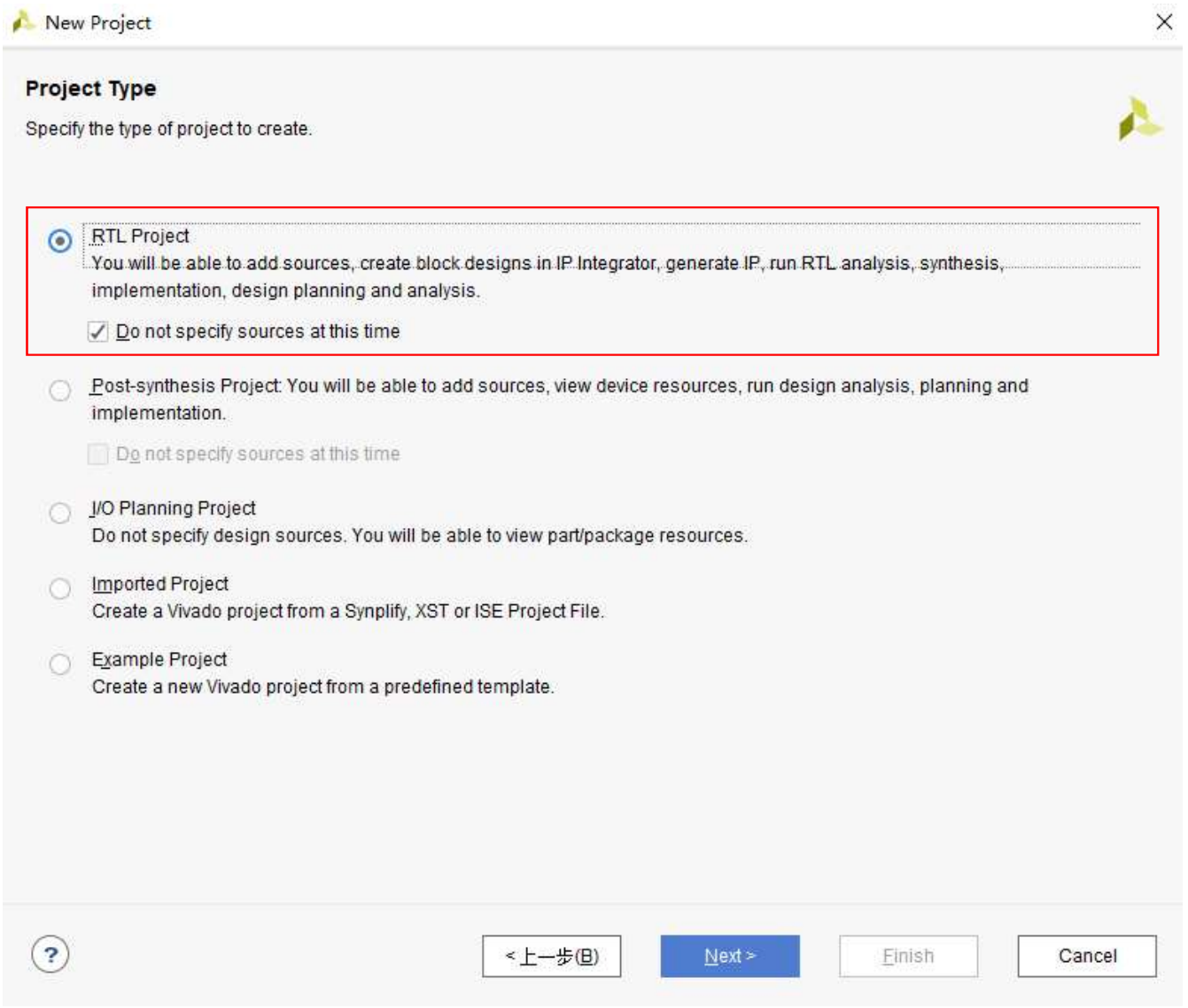


输入项目名称

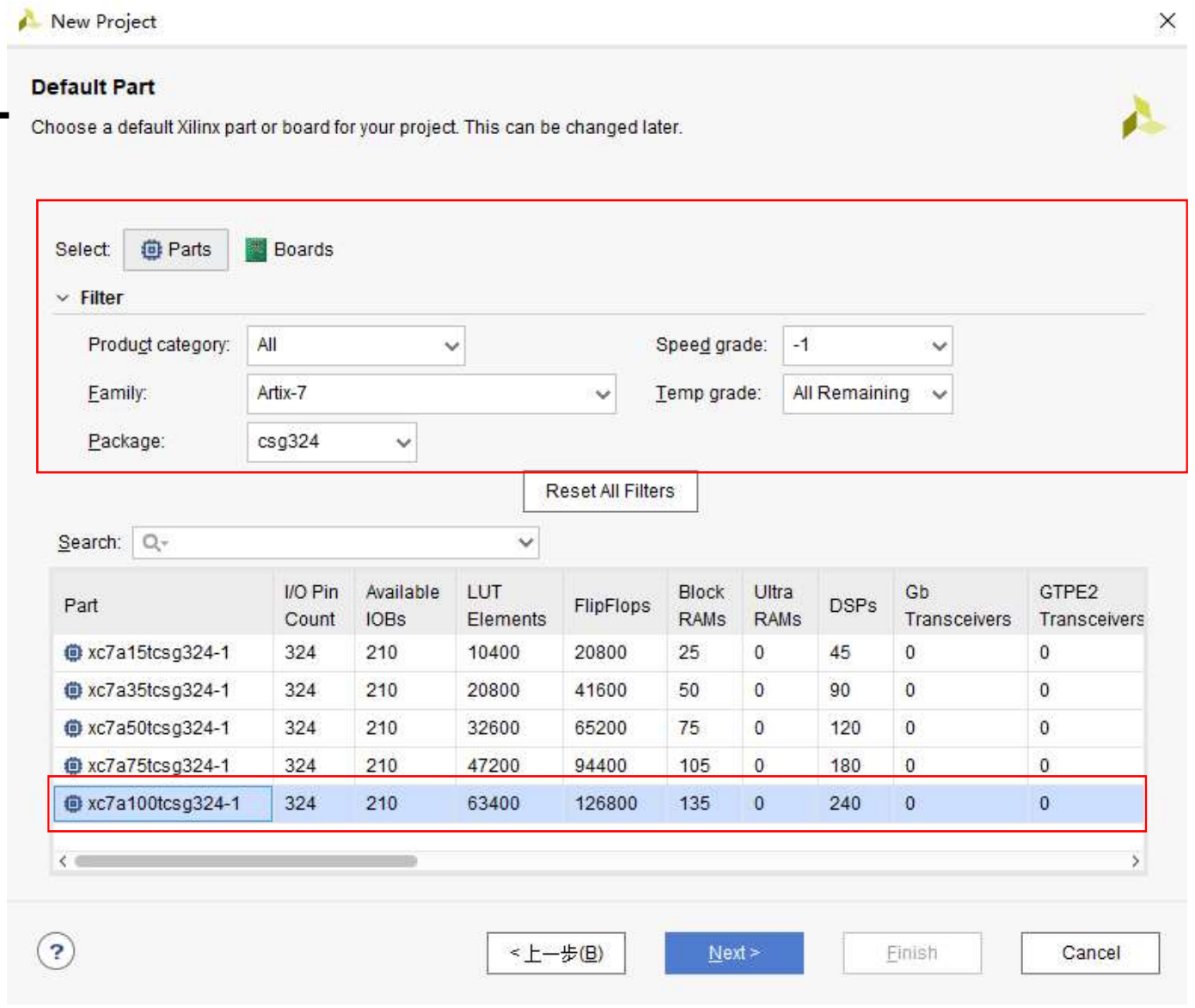
注意工程路径及工程名不要有中文字符

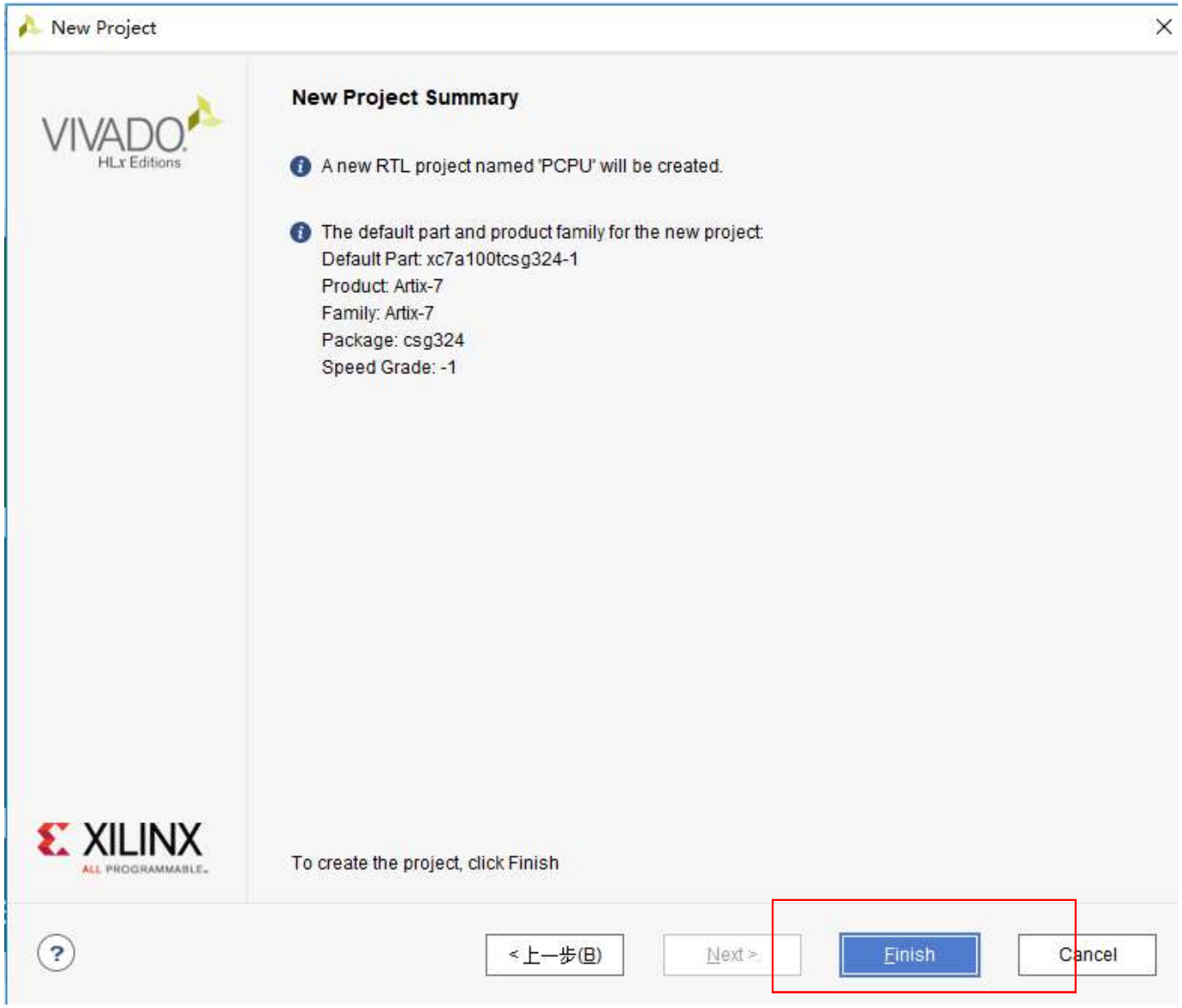


选择项目类型

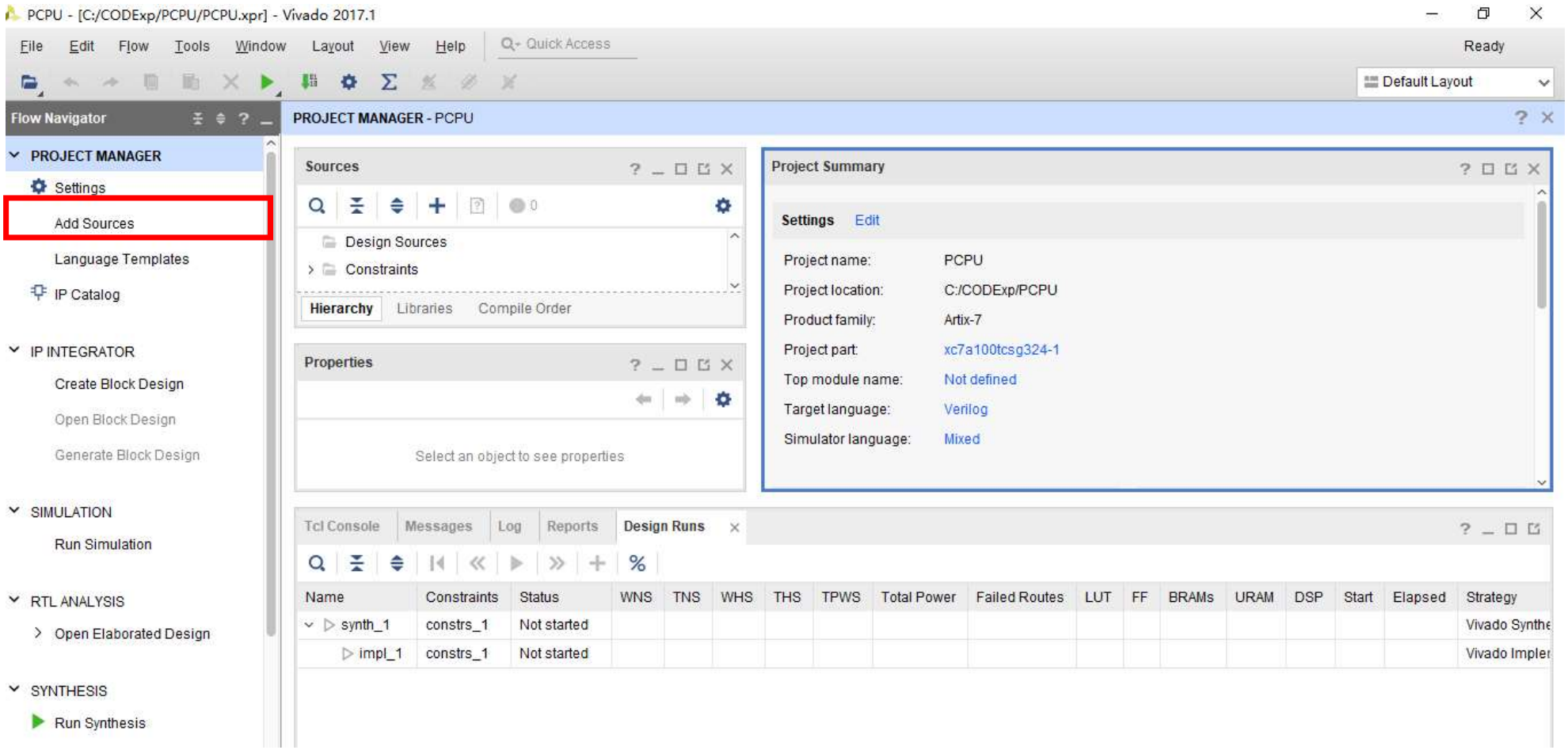

New Project
Project Type
Specify the type of project to create.
RTL Project
You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis.
Do not specify sources at this time
Post-synthesis Project: You will be able to add sources, view device resources, run design analysis, planning and implementation.
Do not specify sources at this time
I/O Planning Project
Do not specify design sources. You will be able to view part/package resources.
Imported Project
Create a Vivado project from a Synplify, XST or ISE Project File.
Example Project
Create a new Vivado project from a predefined template.
< 上一步(B)
Next >
Finish
Cancel

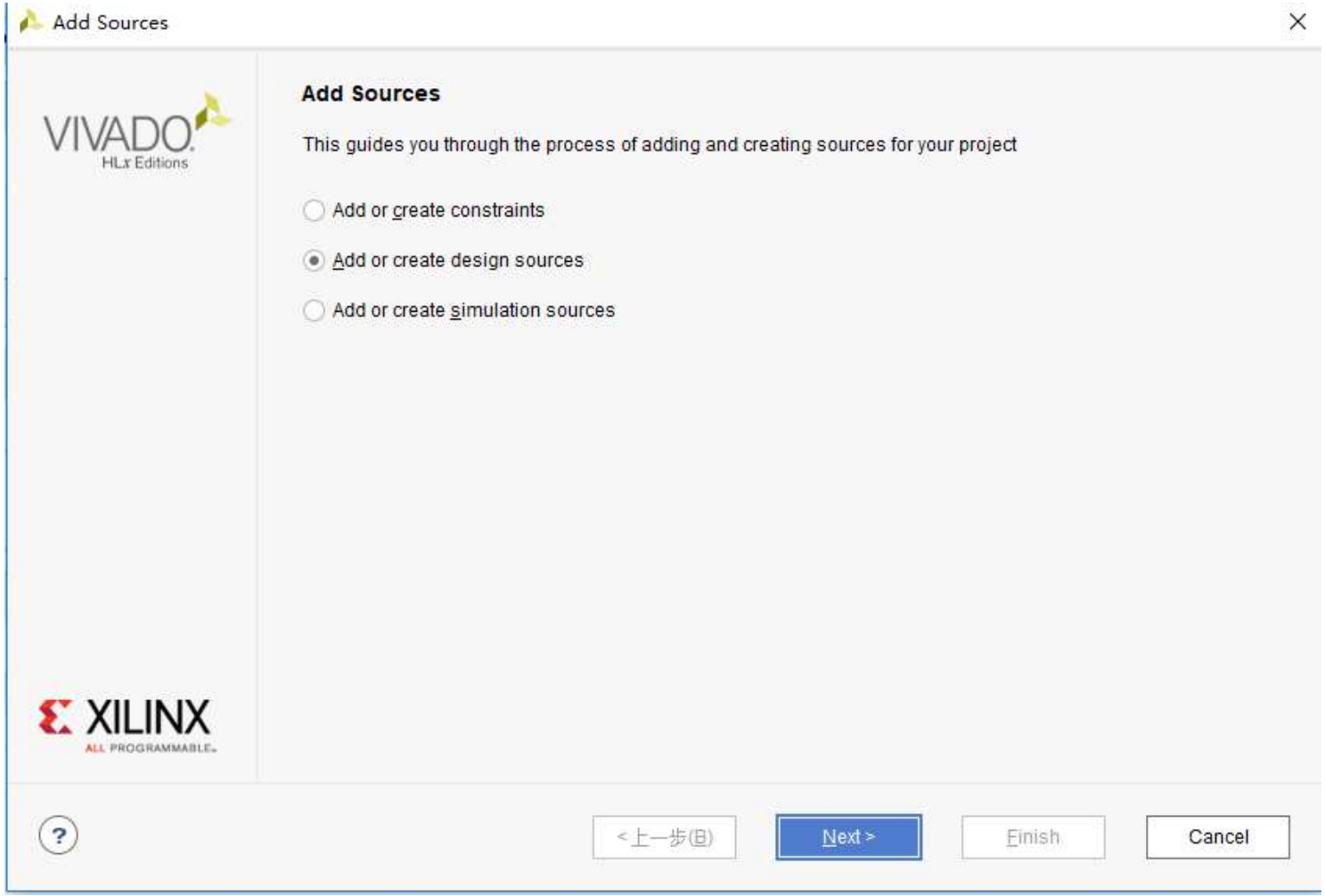
选择FPGA型号

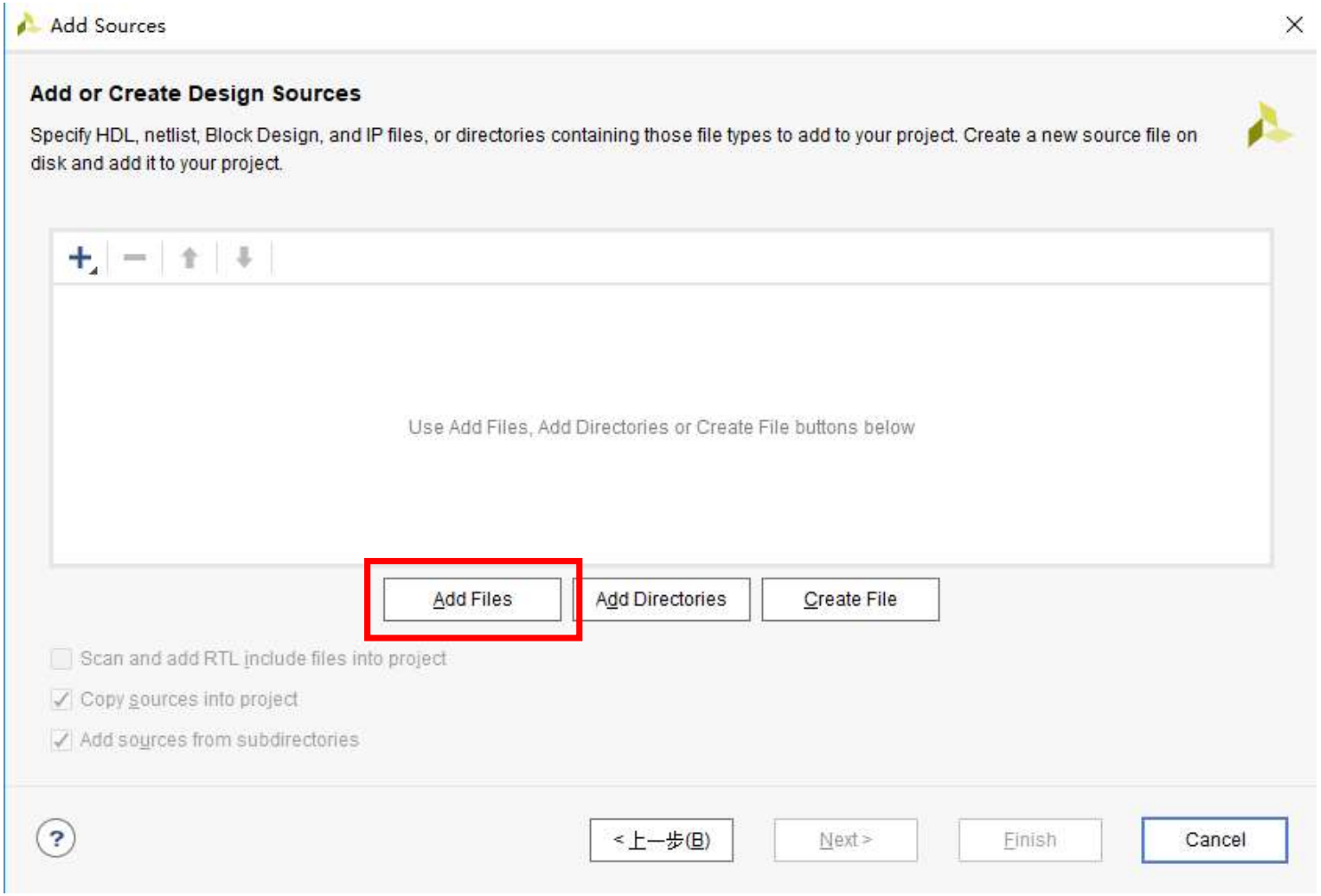

New Project
Default Part
Choose a default Xilinx part or board for your project. This can be changed later.
Select: Parts Boards
Filter
Product category: All
Speed grade: -1
Family: Artix-7
Temp grade: All Remaining
Package: csg324
Reset All Filters
Search:
Part | I/O Pin Count | Available IOBs | LUT Elements | FlipFlops | Block RAMs | Ultra RAMs | DSPs | Gb Transceivers | GTPE2 Transceivers
xc7a15tcsg324-1 | 324 | 210 | 10400 | 20800 | 25 | 0 | 45 | 0 | 0
xc7a35tcsg324-1 | 324 | 210 | 20800 | 41600 | 50 | 0 | 90 | 0 | 0
xc7a50tcsg324-1 | 324 | 210 | 32600 | 65200 | 75 | 0 | 120 | 0 | 0
xc7a75tcsg324-1 | 324 | 210 | 47200 | 94400 | 105 | 0 | 180 | 0 | 0
xc7a100tcsg324-1 | 324 | 210 | 63400 | 126800 | 135 | 0 | 240 | 0 | 0
< 上一步(B)
Next >
Finish
Cancel


New Project
VIVADO
HLx Editions
New Project Summary
A new RTL project named 'PCPU' will be created.
The default part and product family for the new project:
Default Part: xc7a100tcsg324-1
Product: Artix-7
Family: Artix-7
Package: csg324
Speed Grade: -1
XILINX
ALL PROGRAMMABLE
To create the project, click Finish
?
< 上一步(B)
Next >
Finish
Cancel

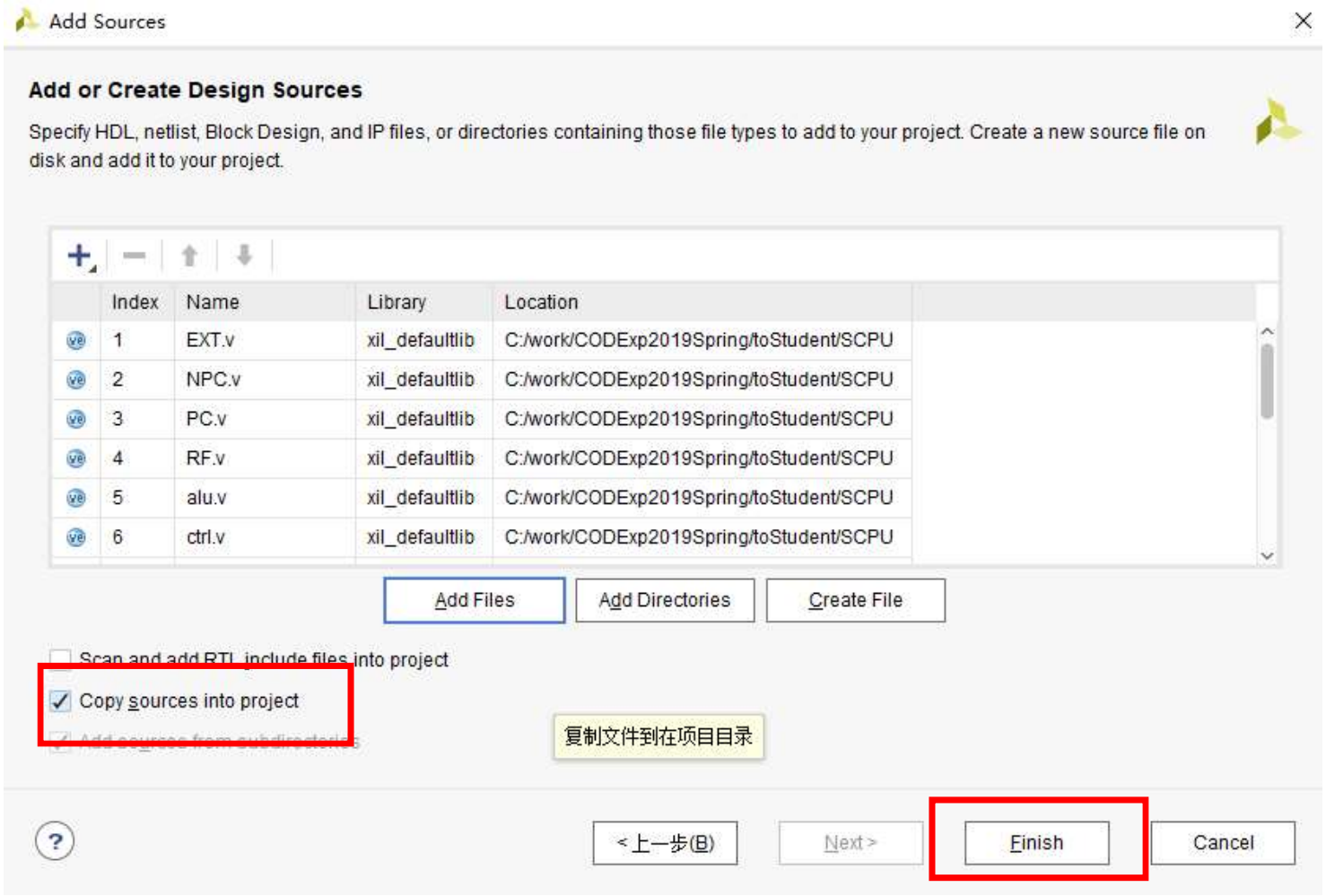
项目管理器--添加文件


PCPU - [C:/CODExp/PCPU/PCPU.xpr] - Vivado 2017.1
File Edit Flow Tools Window Layout View Help
Quick Access
Ready
Default Layout
Flow Navigator
PROJECT MANAGER
Settings
Add Sources
Language Templates
IP Catalog
IP INTEGRATOR
Create Block Design
Open Block Design
Generate Block Design
SIMULATION
Run Simulation
RTL ANALYSIS
Open Elaborated Design
SYNTHESIS
Run Synthesis
PROJECT MANAGER - PCPU
Sources
Design Sources
Constraints
Hierarchy Libraries Compile Order
Properties
Select an object to see properties
Project Summary
Settings Edit
Project name: PCPU
Project location: C:/CODExp/PCPU
Product family: Artix-7
Project part: xc7a100tcsg324-1
Top module name: Not defined
Target language: Verilog
Simulator language: Mixed
Tcl Console Messages Log Reports Design Runs
Name Constraints Status WNS TNS WHS THS TPWS Total Power Failed Routes LUT FF BRAMs URAM DSP Start Elapsed Strategy
synth_1 constrs_1 Not started Vivado Synthe
impl_1 constrs_1 Not started Vivado Impler

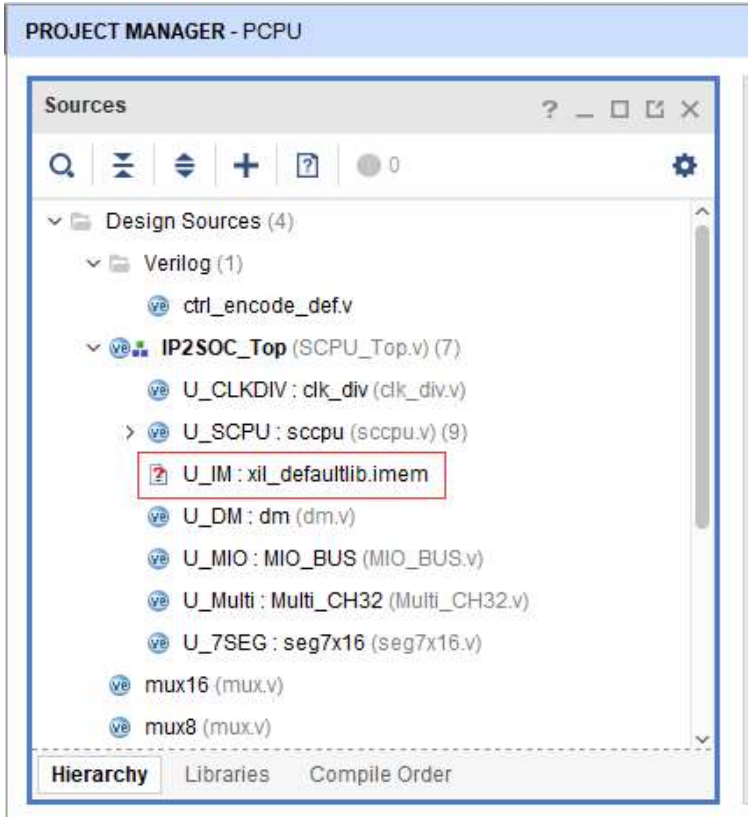

Add Sources
VIVADO
HLx Editions
Add Sources
This guides you through the process of adding and creating sources for your project
Add or create constraints
Add or create design sources
Add or create simulation sources
XILINX
ALL PROGRAMMABLE
?
< 上一步(B)
Next >
Finish
Cancel



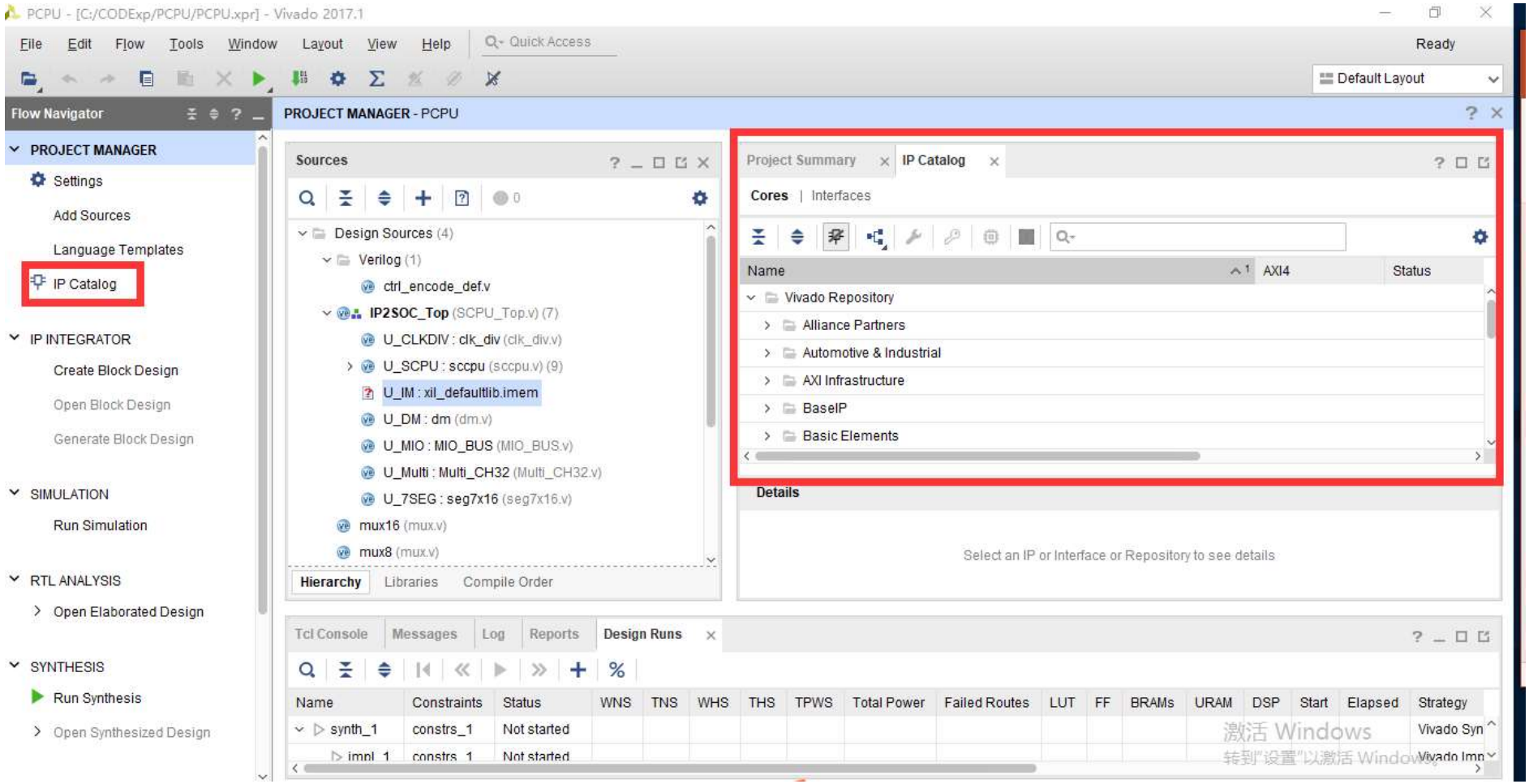
添加给你的代码CPU实现文件
再添加给你的FPGA相关的所有文件

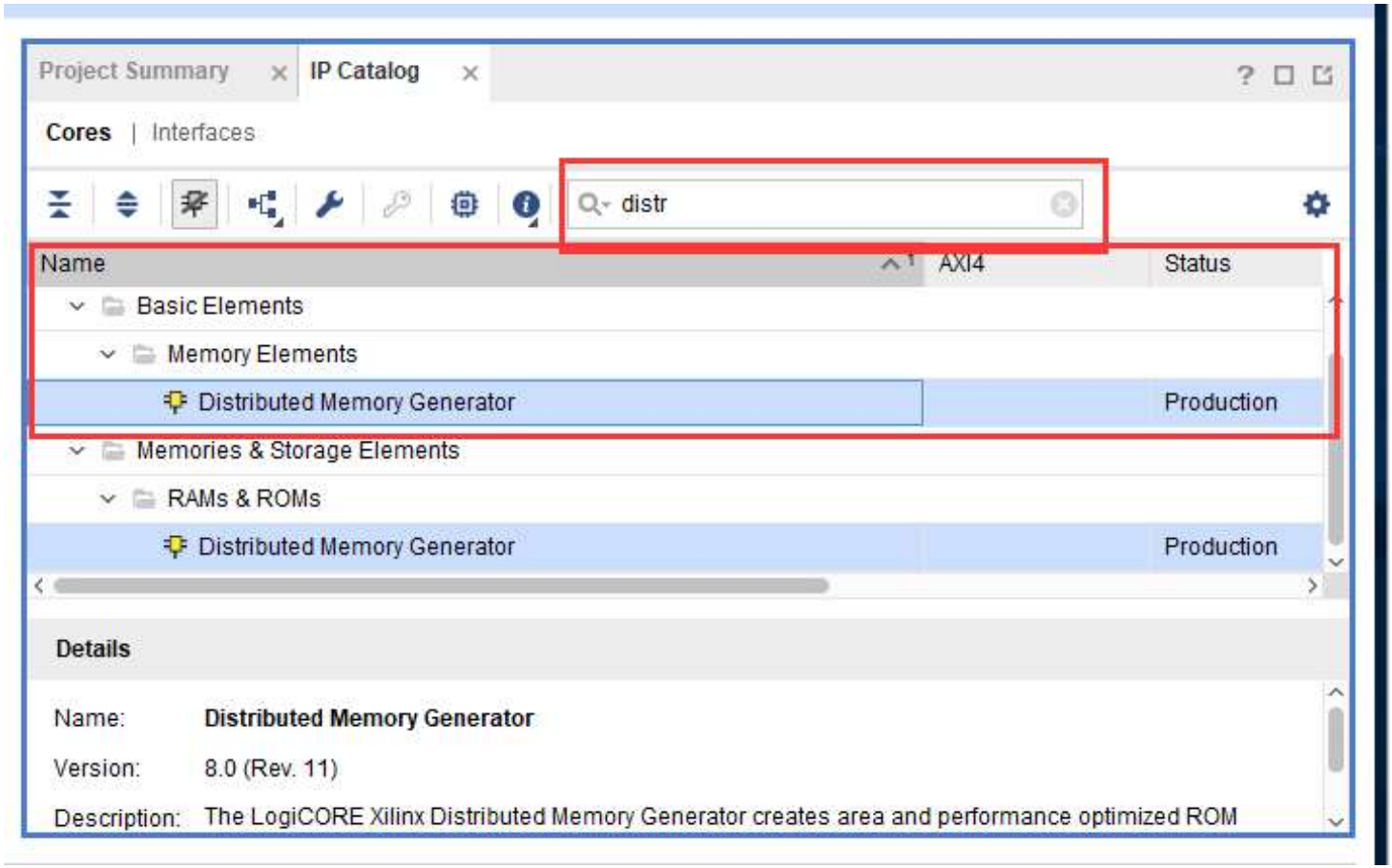

Add Sources
Add or Create Design Sources
Specify HDL, netlist, Block Design, and IP files, or directories containing those file types to add to your project. Create a new source file on disk and add it to your project.
Index
Name
Library
Location
1 EXT.v xil_defaultlib C:/work/CODExp2019Spring/toStudent/SCPU
2 NPC.v xil_defaultlib C:/work/CODExp2019Spring/toStudent/SCPU
3 PC.v xil_defaultlib C:/work/CODExp2019Spring/toStudent/SCPU
4 RF.v xil_defaultlib C:/work/CODExp2019Spring/toStudent/SCPU
5 alu.v xil_defaultlib C:/work/CODExp2019Spring/toStudent/SCPU
6 ctrl.v xil_defaultlib C:/work/CODExp2019Spring/toStudent/SCPU
Add Files
Add Directories
Create File
Scan and add RTL include files into project
Copy sources into project
复制文件到在项目目录
< 上一步(B)
Next >
Finish
Cancel

添加设计文件后的项目管理器



ROM IP核生成


PCPU - [C:/CODExp/PCPU/PCPU.xpr] - Vivado 2017.1
File Edit Flow Tools Window Layout View Help
Quick Access
Ready
Default Layout
Flow Navigator
PROJECT MANAGER
Settings
Add Sources
Language Templates
IP Catalog
IP INTEGRATOR
Create Block Design
Open Block Design
Generate Block Design
SIMULATION
Run Simulation
RTL ANALYSIS
Open Elaborated Design
SYNTHESIS
Run Synthesis
Open Synthesized Design
PROJECT MANAGER - PCPU
Sources
Design Sources (4)
Verilog (1)
ctrl_encode_def.v
IP2SOC_Top (SCPU_Top.v) (7)
U_CLKDIV : clk_div (clk_div.v)
U_SCPU : sccpu (sccpu.v) (9)
U_IM : xil_defaultlib.imem
U_DM : dm (dm.v)
U_MIO : MIO_BUS (MIO_BUS.v)
U_Multi : Multi_CH32 (Multi_CH32.v)
U_7SEG : seg7x16 (seg7x16.v)
mux16 (mux.v)
mux8 (mux.v)
Hierarchy Libraries Compile Order
Project Summary
IP Catalog
Cores | Interfaces
Name
AXI4
Status
Vivado Repository
Alliance Partners
Automotive & Industrial
AXI Infrastructure
BaseIP
Basic Elements
Details
Select an IP or Interface or Repository to see details
Tcl Console Messages Log Reports Design Runs
Name Constraints Status WNS TNS WHS THS TPWS Total Power Failed Routes LUT FF BRAMs URAM DSP Start Elapsed Strategy
synth_1 constrs_1 Not started Vivado Syn
impl_1 constrs_1 Not started Vivado Imp
激活 Windows
转到"设置"以激活 Windows。

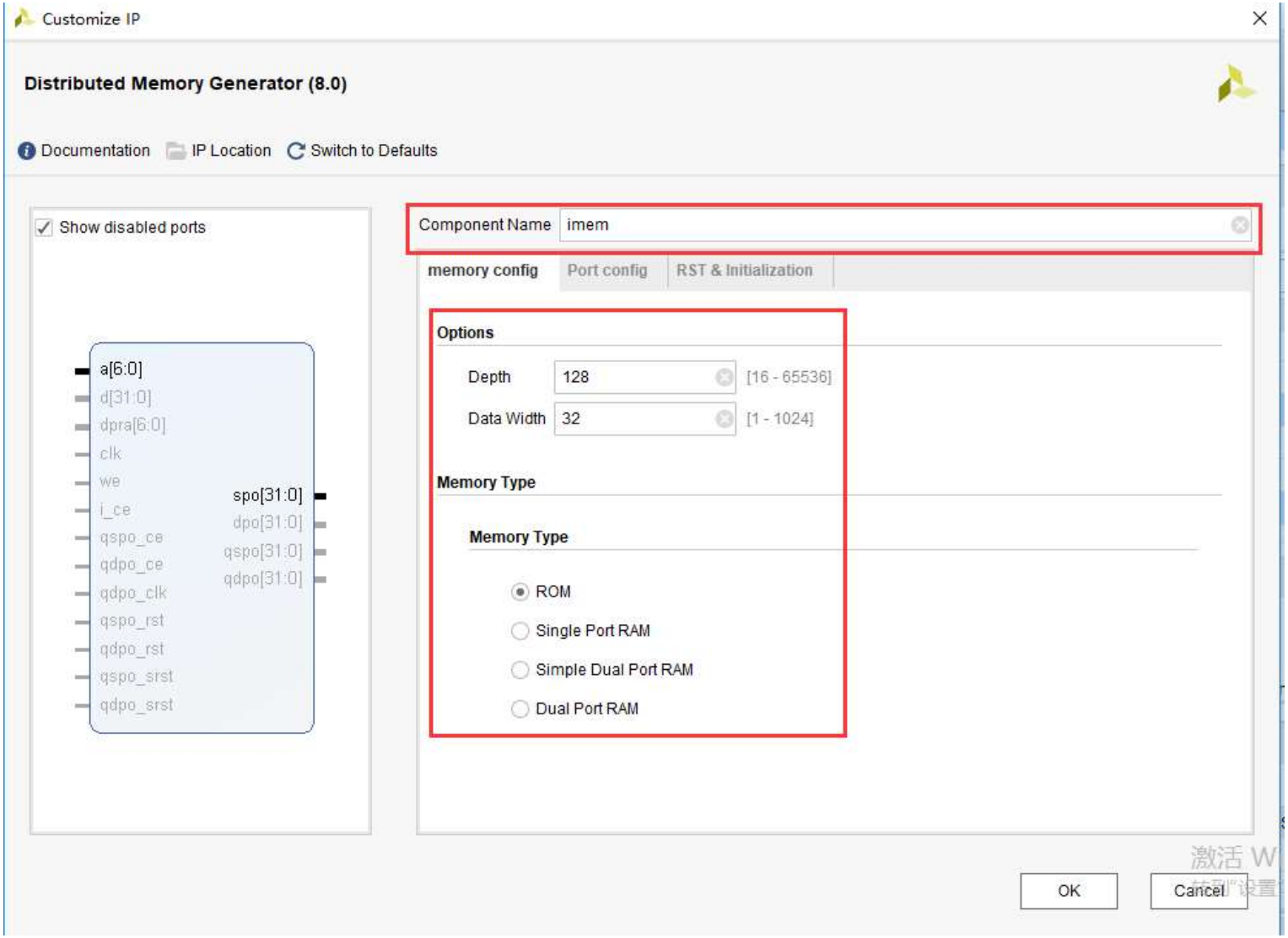


双击“Distributed Memory Generator”

配置ROM IP核 – memory config

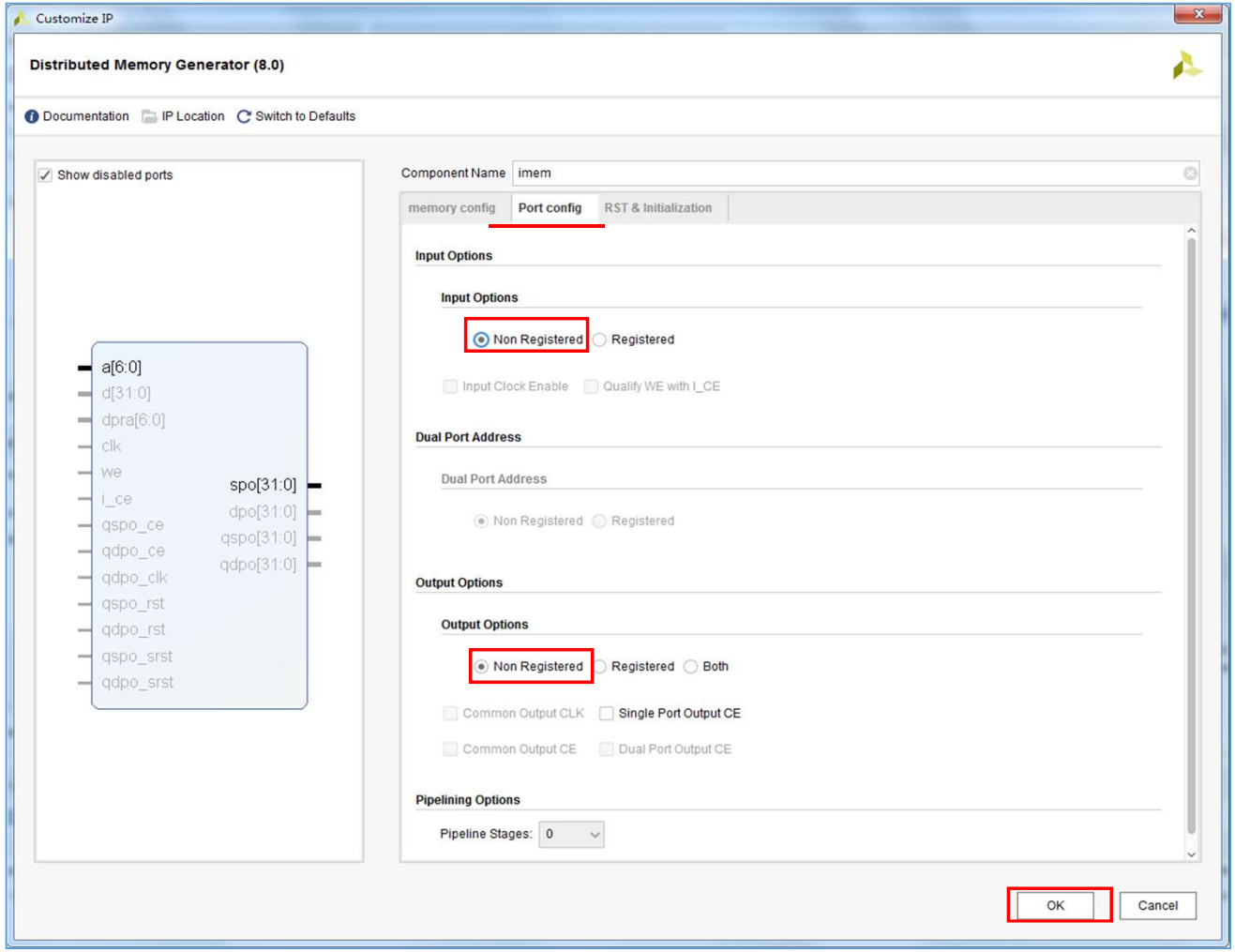
Component Name必须为imem

其他选项也按照此处的配置进行选择



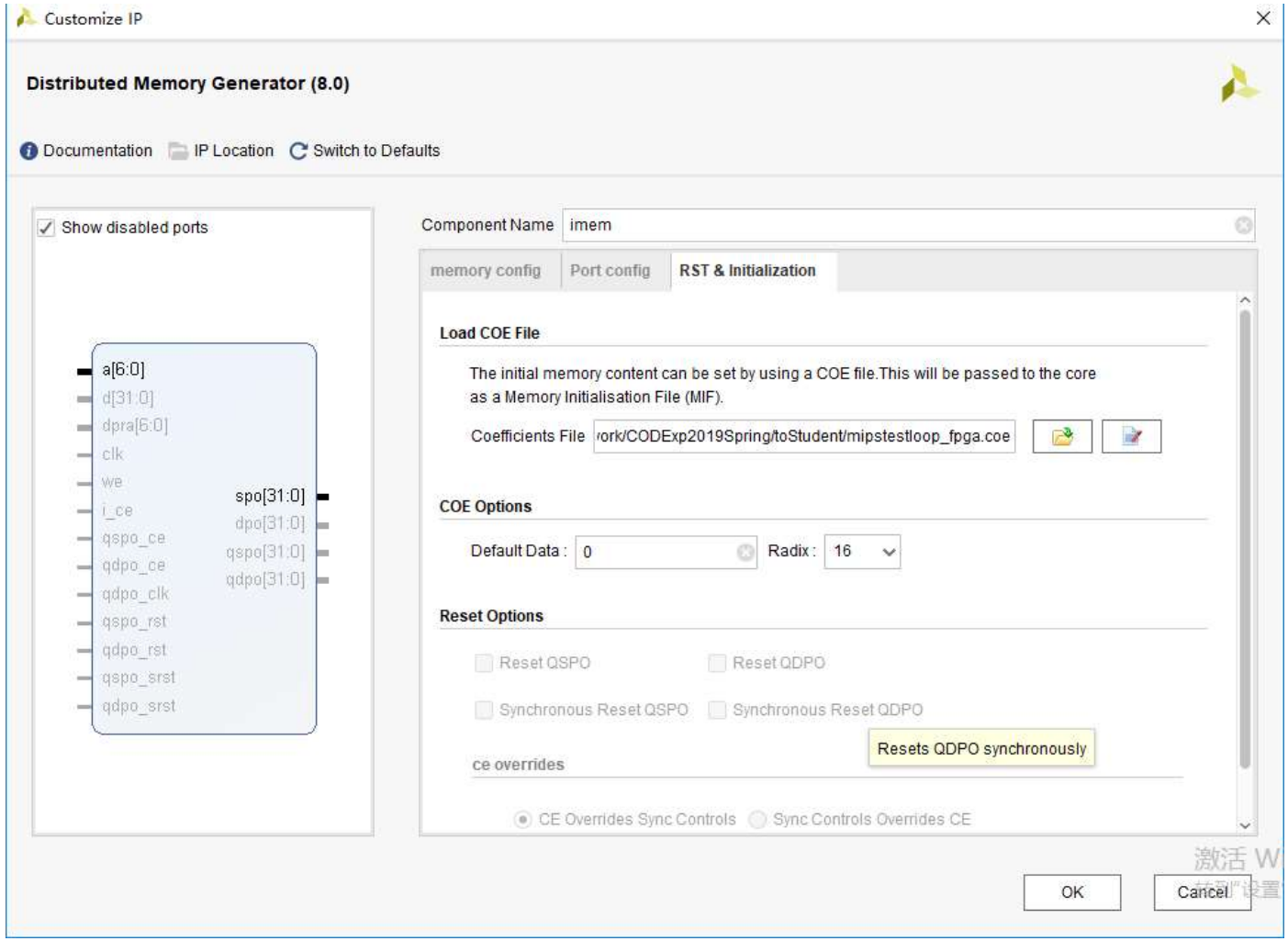
配置ROM IP核 – Port config

选择“Port config”标签



配置ROM IP核 – RST & Initialization

- 选择“RST&Initialization”标签
- “Load COE File”处选择给你的coe目录下的riscv-studentnosorting.coe
- COE(Coefficient)文件是ROM的初始化需要使用来传递参数

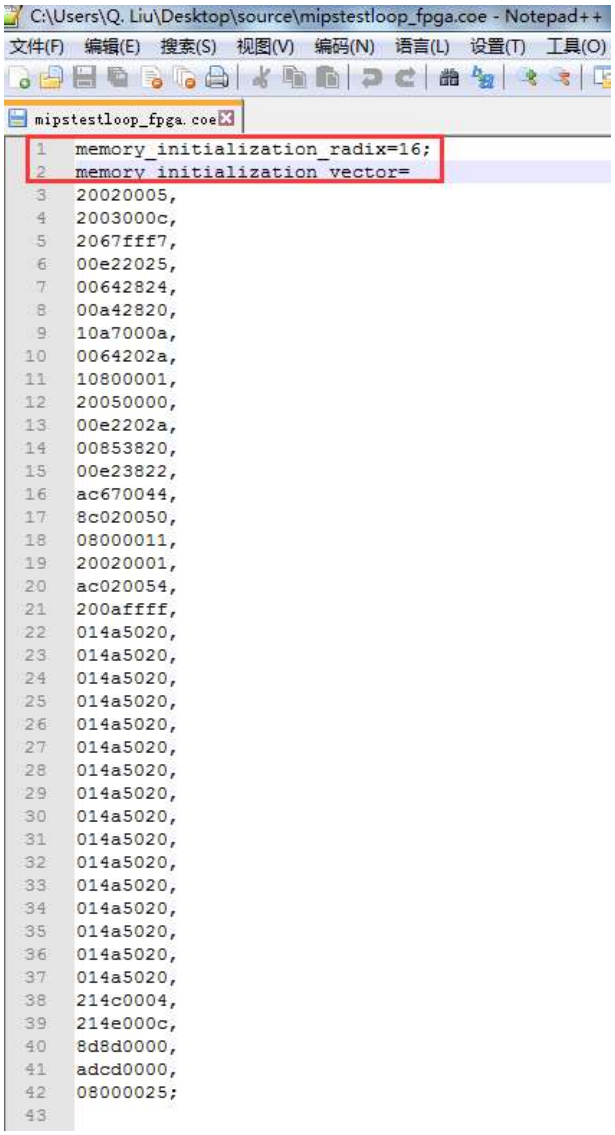


COE文件

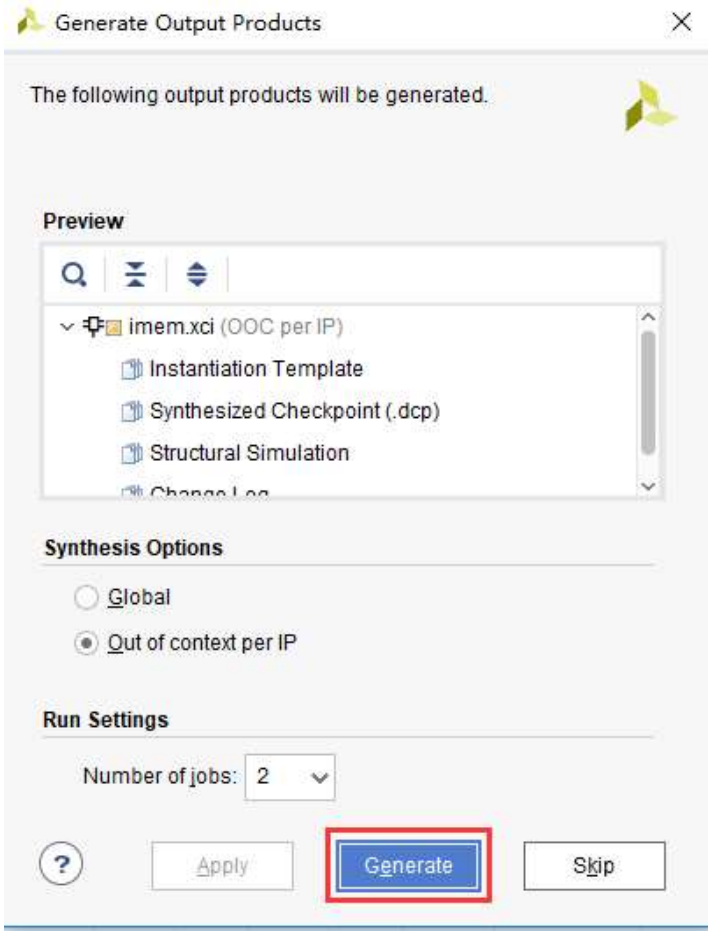
1. 写汇编代码
2. Venus生成机器代码
3. prep-machine-code.py去除开头的0x
4. prep-coe-code.py将之变为coe文件：
   - 在机器代码的开头加上两行

     memory_initialization_radix=16;

     memory_initialization_vector=
   - 其余各行末尾添加“,”
   - 最后一行末尾添加“;”

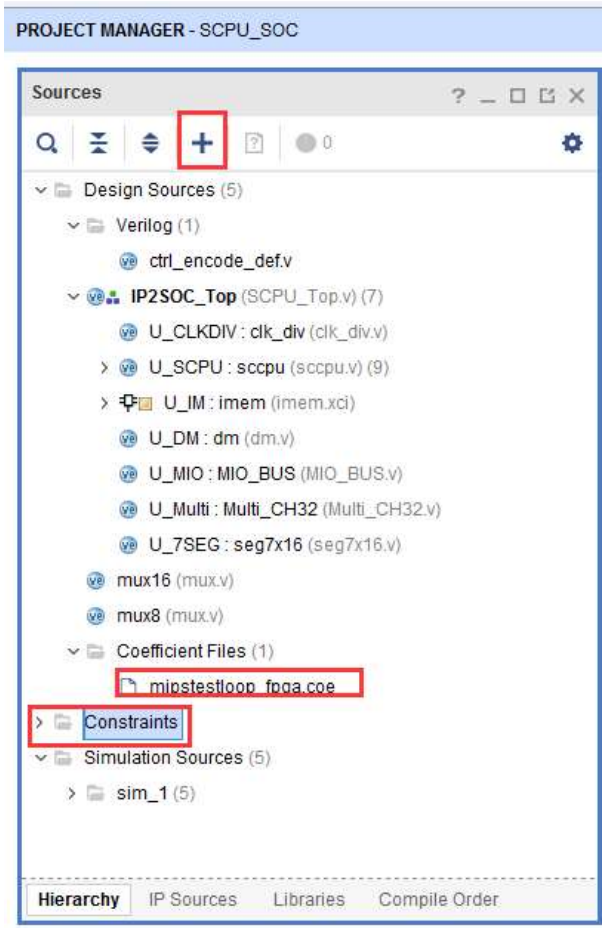


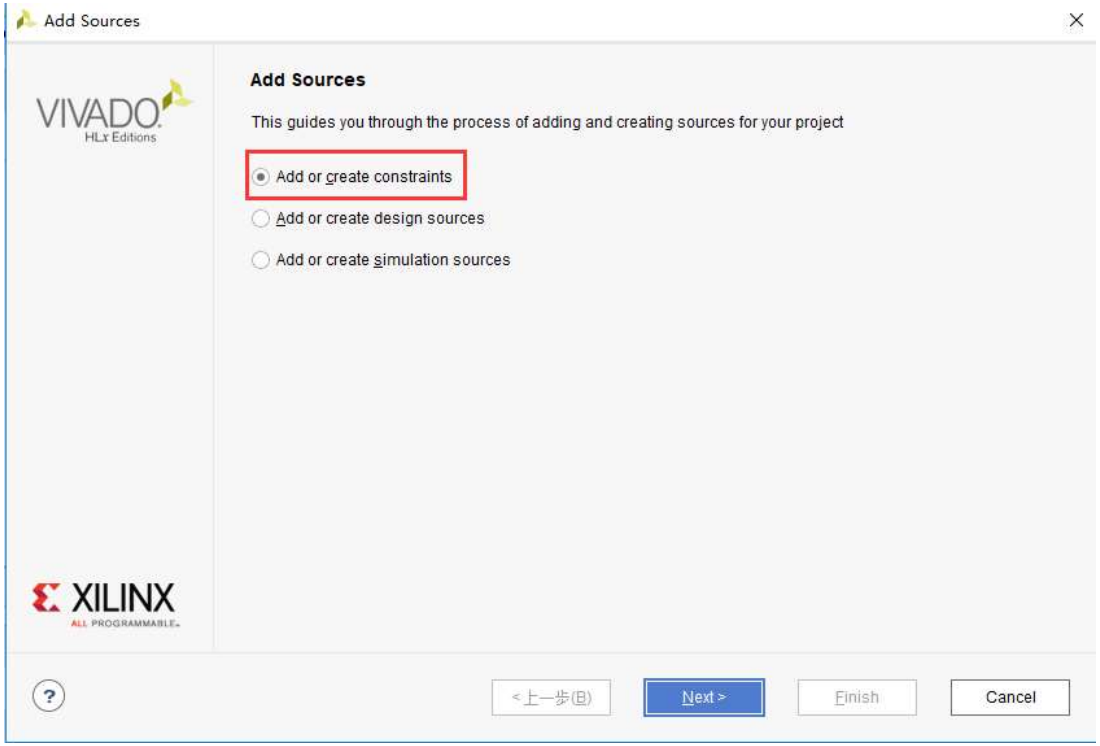
产生ROM IP核

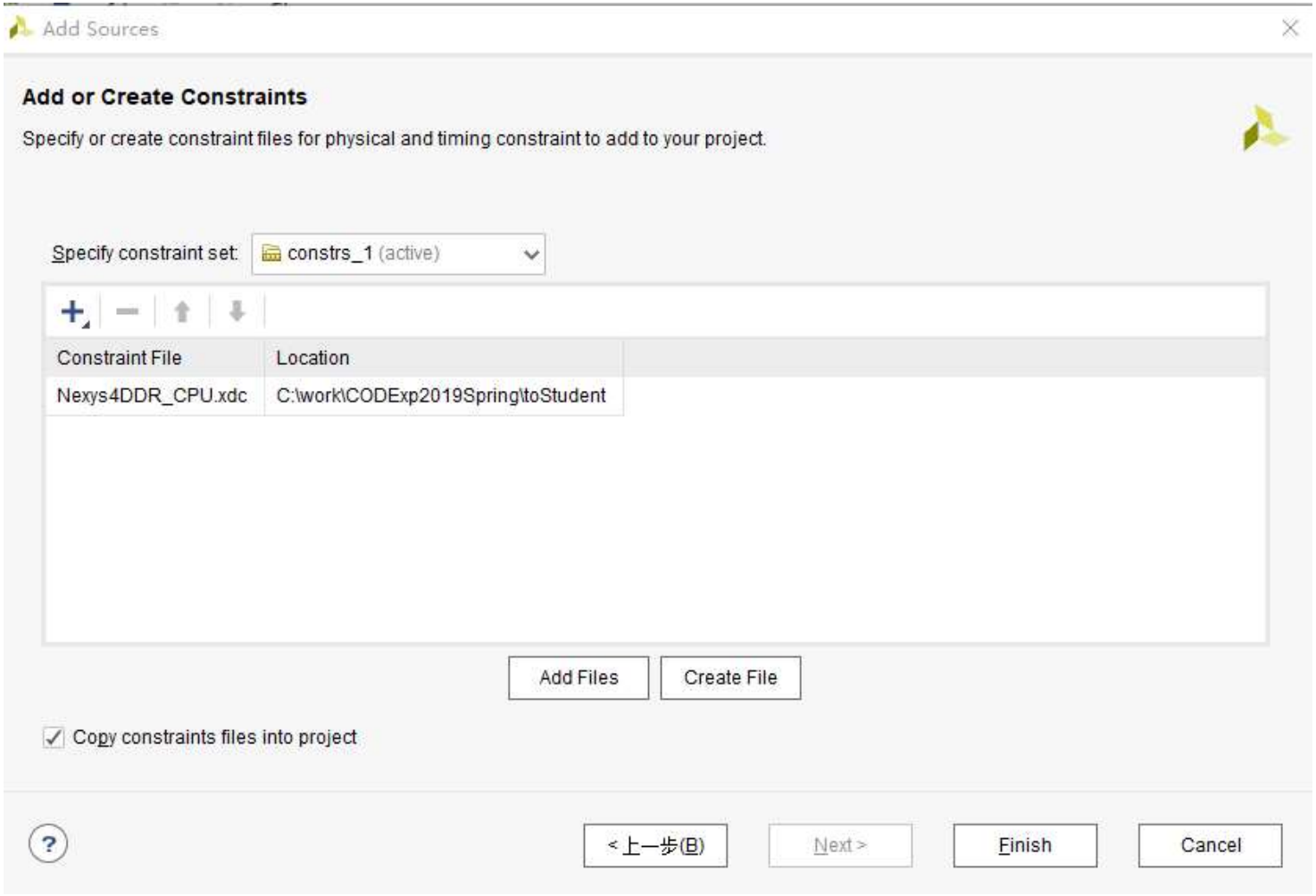

Generate Output Products
The following output products will be generated.
Preview
imem.xci (OOC per IP)
Instantiation Template
Synthesized Checkpoint (.dcp)
Structural Simulation
Synthesis Options
Global
Out of context per IP
Run Settings
Number of jobs: 2
Apply
Generate
Skip

添加约束文件

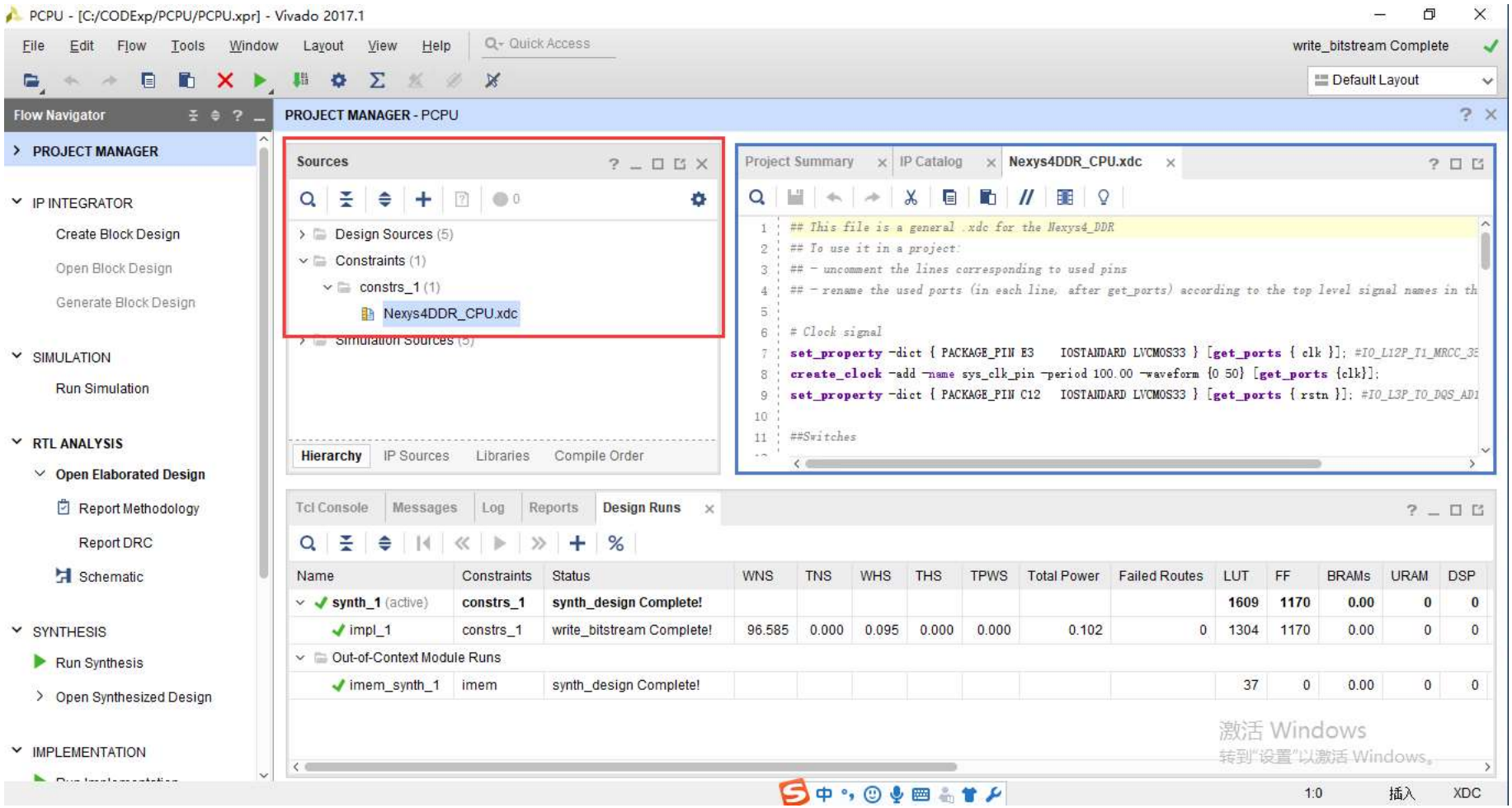
Flow Navigator->Project Manager->Add Sources







添加给你的代码目录下的Nexys4DDR_CPU.xdc



Project Manager->Sources窗口中
双击Constraints下constrs_1下Nexys4DDR_CPU.xdc
FPGA板上的各个部件特性可参考发给你们的Nexys4-DDR_rm.pdf

约束文件

- 约束文件将verilog中定义的端口号与FPGA板子上的IO口建立起联系

```
# Clock signal
set_property -dict { PACKAGE_PIN E3    IOSTANDARD LVCMOS33 } [get_ports { clk }]; #IO_L12P_T1_
create_clock -add -name sys_clk_pin -period 100.00 -waveform {0 50} [get_ports {clk}]
set_property -dict { PACKAGE_PIN C12   IOSTANDARD LVCMOS33 } [get_ports { rstn }]; #IO_L3P_T0_

##Switches

set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMOS33 } [get_ports { sw_i[0] }]; #IO_L24N
set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { sw_i[1] }]; #IO_L3N_
set_property -dict { PACKAGE_PIN M13   IOSTANDARD LVCMOS33 } [get_ports { sw_i[2] }]; #IO_L6N_
set_property -dict { PACKAGE_PIN R15   IOSTANDARD LVCMOS33 } [get_ports { sw_i[3] }]; #IO_L13N
set_property -dict { PACKAGE_PIN R17   IOSTANDARD LVCMOS33 } [get_ports { sw_i[4] }]; #IO_L12N
set_property -dict { PACKAGE_PIN T18   IOSTANDARD LVCMOS33 } [get_ports { sw_i[5] }]; #IO_L7N_
set_property -dict { PACKAGE_PIN U18   IOSTANDARD LVCMOS33 } [get_ports { sw_i[6] }]; #IO_L17N
set_property -dict { PACKAGE_PIN R13   IOSTANDARD LVCMOS33 } [get_ports { sw_i[7] }]; #IO_L5N_
set_property -dict { PACKAGE_PIN T8    IOSTANDARD LVCMOS18 } [get_ports { sw_i[8] }]; #IO_L24N
set_property -dict { PACKAGE_PIN U8    IOSTANDARD LVCMOS18 } [get_ports { sw_i[9] }]; #IO_25_3
set_property -dict { PACKAGE_PIN R16   IOSTANDARD LVCMOS33 } [get_ports { sw_i[10] }]; #IO_L15
set_property -dict { PACKAGE_PIN T13   IOSTANDARD LVCMOS33 } [get_ports { sw_i[11] }]; #IO_L23
set_property -dict { PACKAGE_PIN H6    IOSTANDARD LVCMOS33 } [get_ports { sw_i[12] }]; #IO_L24
set_property -dict { PACKAGE_PIN U12   IOSTANDARD LVCMOS33 } [get_ports { sw_i[13] }]; #IO_L20
set_property -dict { PACKAGE_PIN U11   IOSTANDARD LVCMOS33 } [get_ports { sw_i[14] }]; #IO_L19
set_property -dict { PACKAGE_PIN V10   IOSTANDARD LVCMOS33 } [get_ports { sw_i[15] }]; #IO_L21
```

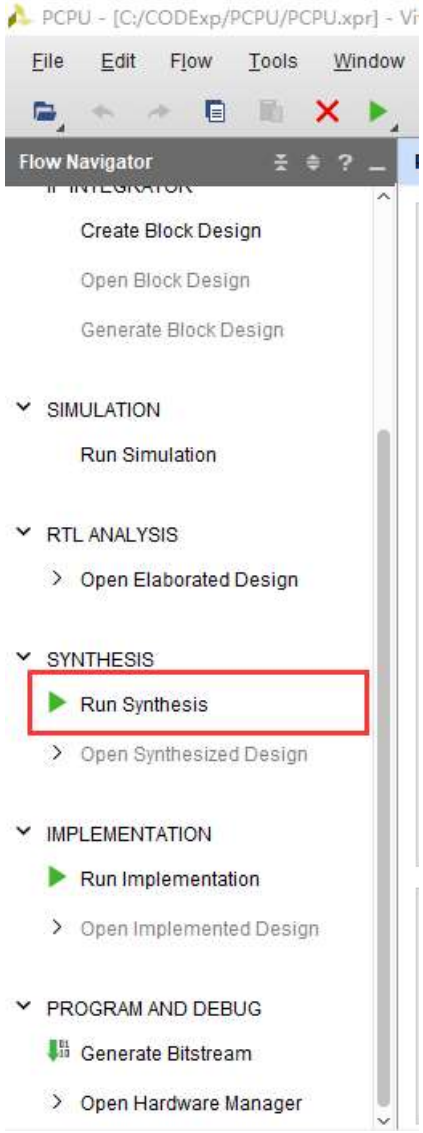
```
##7 segment display
set_property -dict { PACKAGE_PIN T10   IOSTANDARD LVCMOS33 } [get_ports { disp_seg_o[0] }]; #IO_L24N_T3_A00_D16_14 Sch
set_property -dict { PACKAGE_PIN R10   IOSTANDARD LVCMOS33 } [get_ports { disp_seg_o[1] }]; #IO_25_14 Sch=cb
set_property -dict { PACKAGE_PIN K16   IOSTANDARD LVCMOS33 } [get_ports { disp_seg_o[2] }]; #IO_25_15 Sch=cc
set_property -dict { PACKAGE_PIN K13   IOSTANDARD LVCMOS33 } [get_ports { disp_seg_o[3] }]; #IO_L7P_T2_A26_15 Sch=cd
set_property -dict { PACKAGE_PIN P15   IOSTANDARD LVCMOS33 } [get_ports { disp_seg_o[4] }]; #IO_L13P_T2_MRCC_14 Sch=ce
set_property -dict { PACKAGE_PIN T11   IOSTANDARD LVCMOS33 } [get_ports { disp_seg_o[5] }]; #IO_L19P_T3_A10_D26_14 Sch
set_property -dict { PACKAGE_PIN L18   IOSTANDARD LVCMOS33 } [get_ports { disp_seg_o[6] }]; #IO_L4P_T0_D04_14 Sch=cg
set_property -dict { PACKAGE_PIN H15   IOSTANDARD LVCMOS33 } [get_ports { disp_seg_o[7] }]; #IO_L19N_T3_A21_VREF_15 Sch

set_property -dict { PACKAGE_PIN J17   IOSTANDARD LVCMOS33 } [get_ports { disp_an_o[0] }]; #IO_L23P_T3_F0E_B_15 Sch=an[0]
set_property -dict { PACKAGE_PIN J18   IOSTANDARD LVCMOS33 } [get_ports { disp_an_o[1] }]; #IO_L23N_T3_FWE_B_15 Sch=an[1]
set_property -dict { PACKAGE_PIN T9    IOSTANDARD LVCMOS33 } [get_ports { disp_an_o[2] }]; #IO_L24P_T3_A01_D17_14 Sch=an[2]
set_property -dict { PACKAGE_PIN J14   IOSTANDARD LVCMOS33 } [get_ports { disp_an_o[3] }]; #IO_L19P_T3_A22_15 Sch=an[3]
set_property -dict { PACKAGE_PIN P14   IOSTANDARD LVCMOS33 } [get_ports { disp_an_o[4] }]; #IO_L8N_T1_D12_14 Sch=an[4]
set_property -dict { PACKAGE_PIN T14   IOSTANDARD LVCMOS33 } [get_ports { disp_an_o[5] }]; #IO_L14P_T2_SRCC_14 Sch=an[5]
set_property -dict { PACKAGE_PIN K2    IOSTANDARD LVCMOS33 } [get_ports { disp_an_o[6] }]; #IO_L23P_T3_35 Sch=an[6]
set_property -dict { PACKAGE_PIN U13   IOSTANDARD LVCMOS33 } [get_ports { disp_an_o[7] }]; #IO_L23N_T3_A02_D18_14 Sch=an[7]
```

综合（Synthesis）

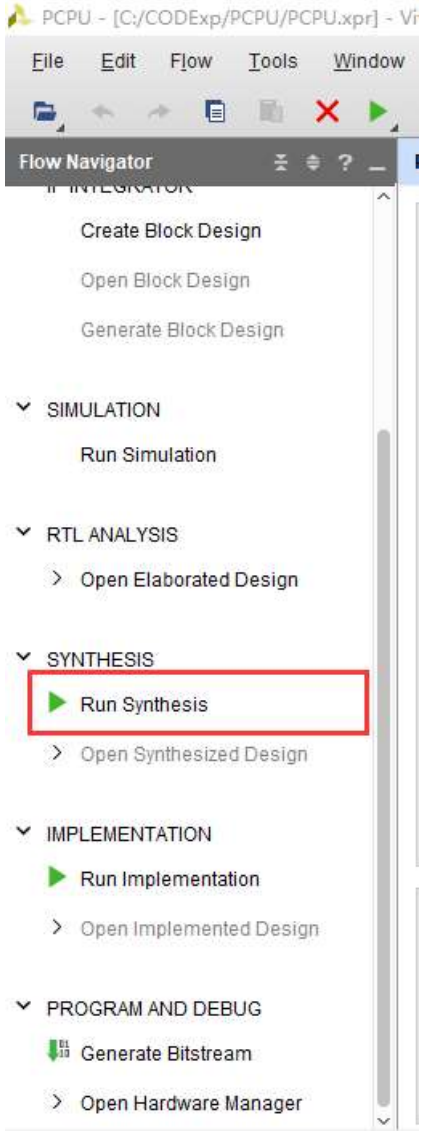
综合是将HDL语言等设计输入翻译成由与、或、非门和RAM、触发器等基本逻辑单元组成的逻辑连接（网表），并根据目标和要求（约束条件）优化生成的RTL层连接。

点击Run Synthesis

然后点OK



点击Run Synthesis
然后点OK



综合所需时间可能较长，注意观察窗口右上角的状态变化

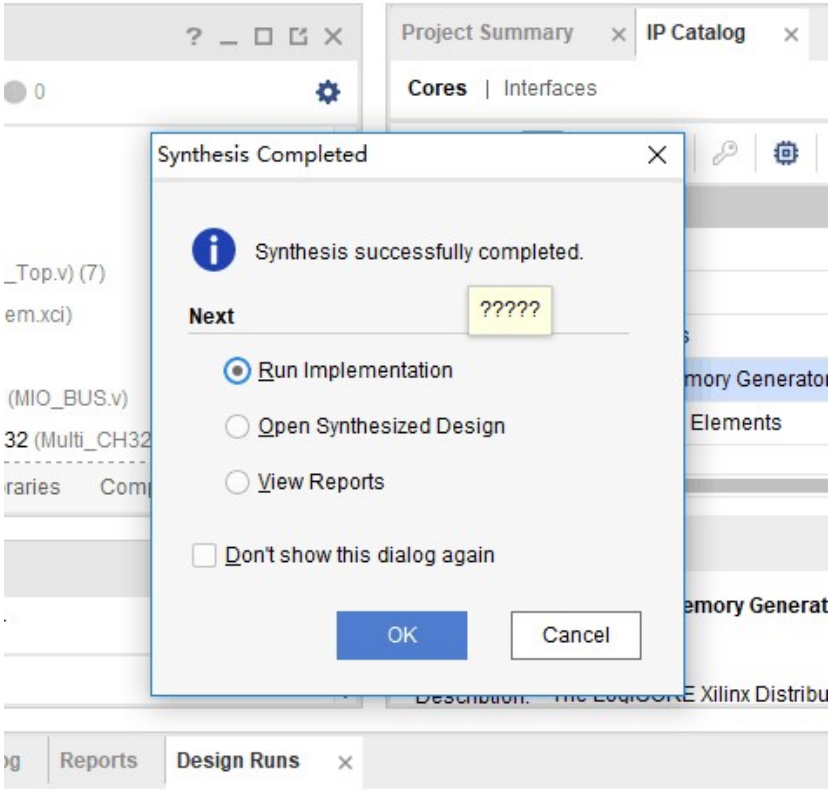
实现（Implementation）

综合完成后会出现如上窗口

实现是综合输出的逻辑网表翻译成所选器件的低层模块和硬件原语，将设计映射到器件结构上，进行布局布线，达到在所选器件上实现设计的目的。

点击Run Implementation，也可以选择Cancel后在左侧Flow Navigator中选择

同样的，这个过程也有可能比较慢，注意观察窗口右上角状态变化



生成bit文件(Generate Bitstream)

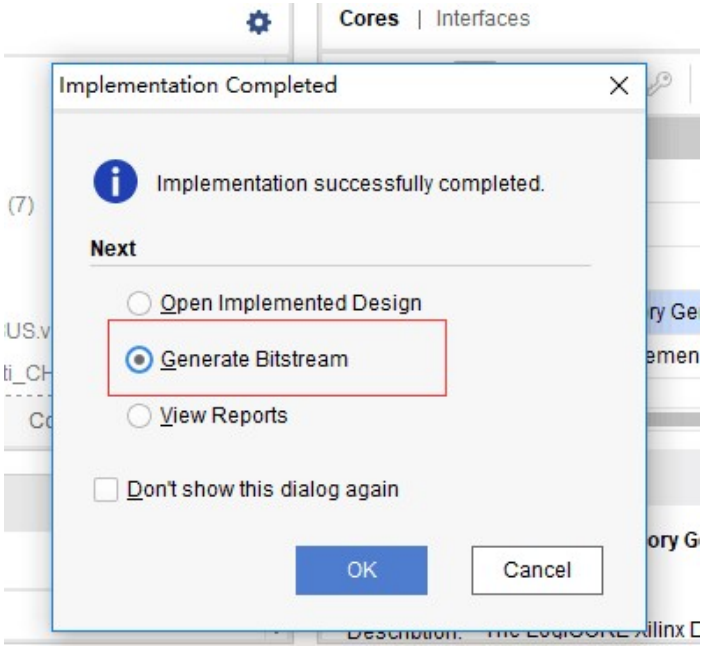
实现完成后会出现如上窗口

选择Generate Bitstream，点击Ok

也可以选择Cancel后在左侧Flow Navigator中选择

若在综合和实现之前点击Generate Bitstream会先执行综合、实现，再生成比特流文件。

同样的，这个过程也有可能比较慢，注意观察窗口右上角状态变化



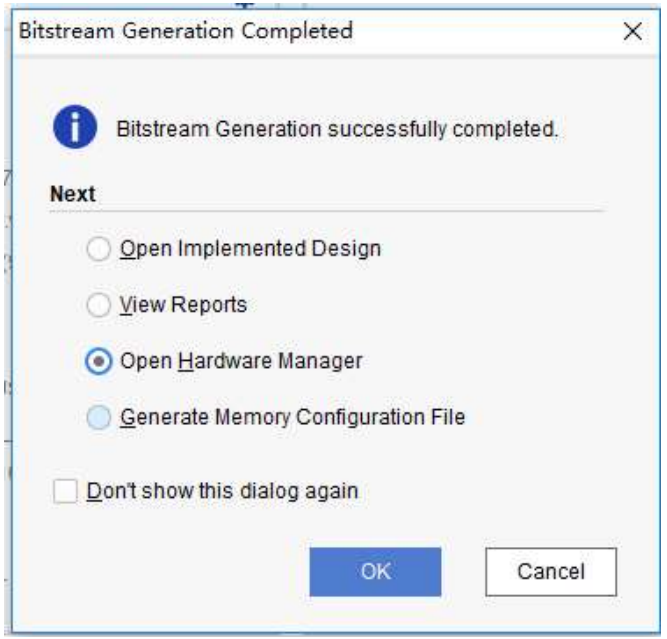
Open Target

位流生成后会出现如上窗口

选择Open Hardware Manager，点击Ok

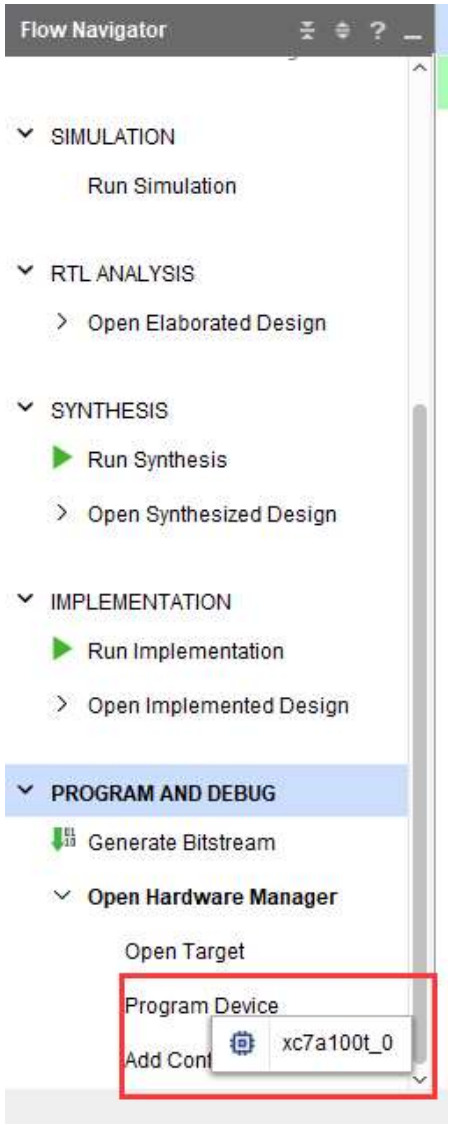
也可以选择Cancel后在左侧Flow Navigator中选择

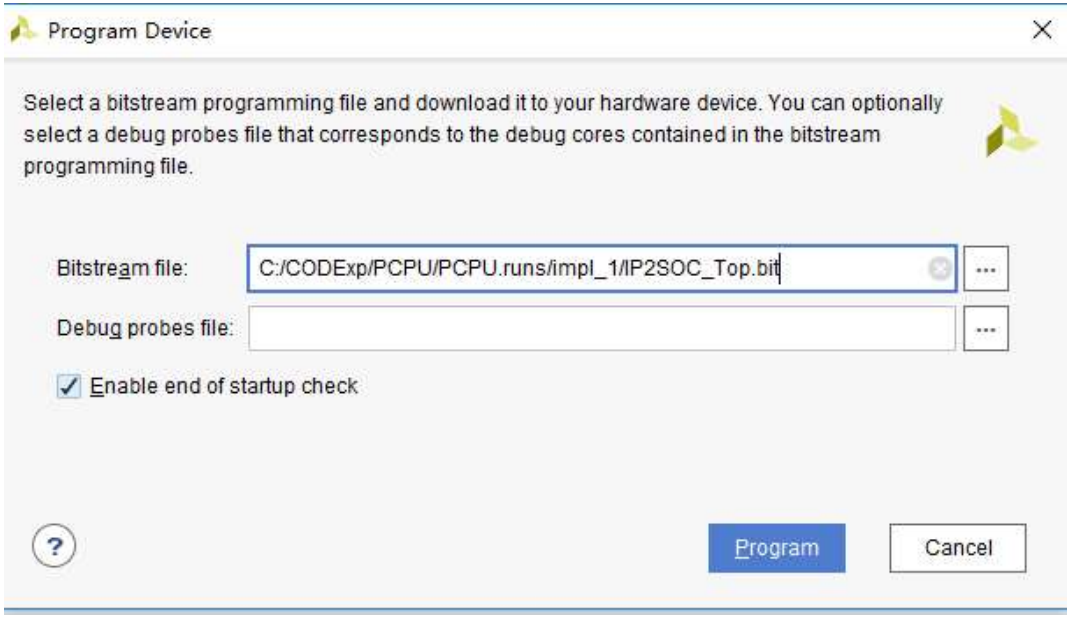
同样的，这个过程也有可能比较慢，注意观察窗口右上角状态变化



下载（Program）

- 如果此前没有连接硬件，则在Open Hardware Manager中选择Open Target
- 选择Auto Connect
- 注意：这时要物理上将设备连接到你的计算机，并打开开关
- 然后再点击Program Device->xc7a100t_0

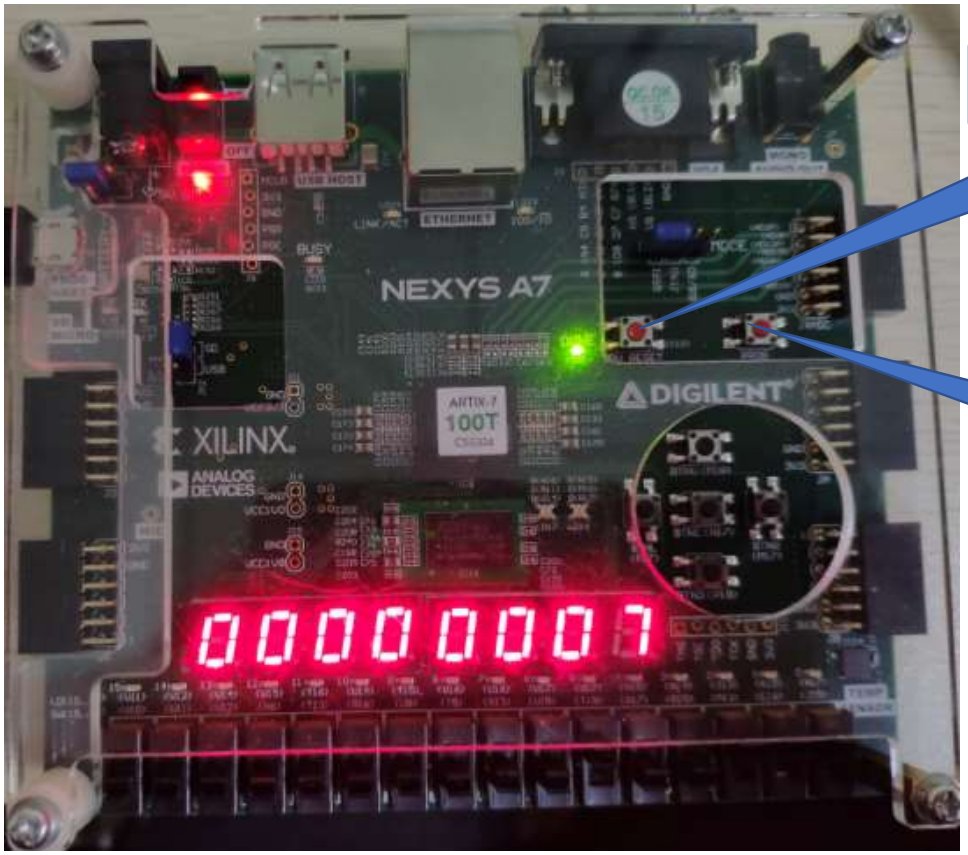



Program Device
Select a bitstream programming file and download it to your hardware device. You can optionally select a debug probes file that corresponds to the debug cores contained in the bitstream programming file.
Bitstream file:
C:/CODExp/PCPU/PCPU.runs/impl_1/IP2SOC_Top.bit
Debug probes file:
Enable end of startup check
Program
Cancel

如果到这里都没有错误，恭喜你，你的设计已经成功地下载到FPGA板上运行起来了！

接下来，了解一下如何查看结果吧！

运行结果检查



CPU复位按钮，按下后PC从0开始执行程序

- SW5-SW0为0时，七段数码管显示CPU数据；
- 拨动开关(SW15-SW6)，16个开关的状态被写入右边四个七段数码管（左边四个为0）
  - 例如当SW15，SW14，SW7为1，其他开关为0时，七段数码管上显示为0000C080

开发板复位按钮，按下后加载Out of Box Demo

- SW5为1表示查看寄存器的值
- SW4-SW0为寄存器编号，当SW1为1时，其他为0时，表示R2
- 其他开关均为0
- R2 = 7
- 将SW15置1，CPU慢速运行；按下CPU复位按钮，程序从0地址开始执行，可以看到R2的值开始为5，后来变为7

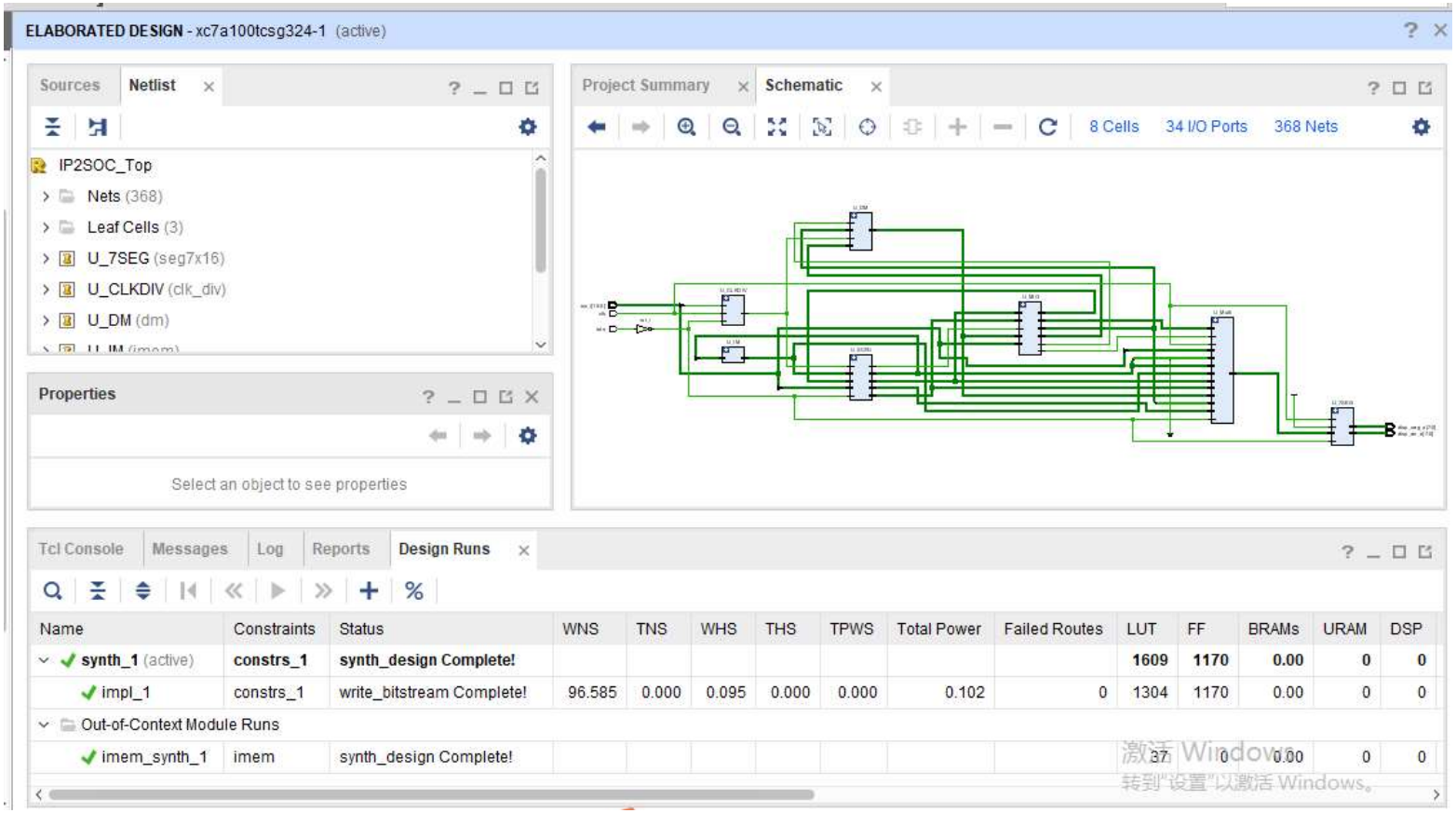
测试开关设置

- 开关作为输入(lw)，对应地址为 0xFFFF0004,
- 七段数码管作为输出（sw)，对应地址为 0xFFFF000C

开关设置						功能
SW15	0					CPU全速运行
	1					CPU慢速运行
SW5	SW4	SW3	SW2	SW1	SW0	七段数码管的内容
0	0	0	0	0	0	CPU数据（指令中写入七段数码管的数据，即sw指令写入地址0xFFFF000C的值）；如果代码中没有该指令，则显示默认值0xAA5555AA
0	0	0	0	0	1	指令编号（PC>>2）
0	0	0	0	1	0	指令地址（PC）
0	0	0	0	1	1	指令（IR）
0	0	0	1	0	0	CPU访问存储器或外设的地址
0	0	0	1	0	1	CPU写入存储器或外设的数据
0	0	0	1	1	0	数据存储器读出的数据
0	0	0	1	1	1	数据存储器的访问地址
0	x	1	x	x	x	0xFFFFFFFF
0	1	x	x	x	x	0xFFFFFFFF
1	寄存器的编号					对应寄存器的值

左侧Flow Navigator窗口中，RTL Analysis->Open Elaborated Design

可以看到项目的顶层设计，右击每个方块可以View Source，View Definition



注意事项

- 没有单步的功能
- 但是可以慢速运行
- 也可以运行完后再看寄存器的值
- 需记录下实验中遇到的问题是怎么解决的，并体现在实验报告中