

实验四 32 位 RISC 处理器的设计与实现 – 实验步骤

(Loongarch32 指令集版本)

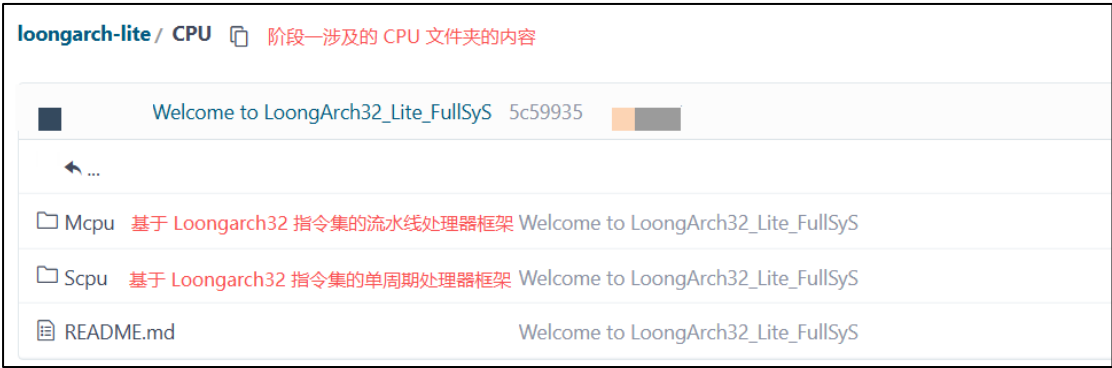
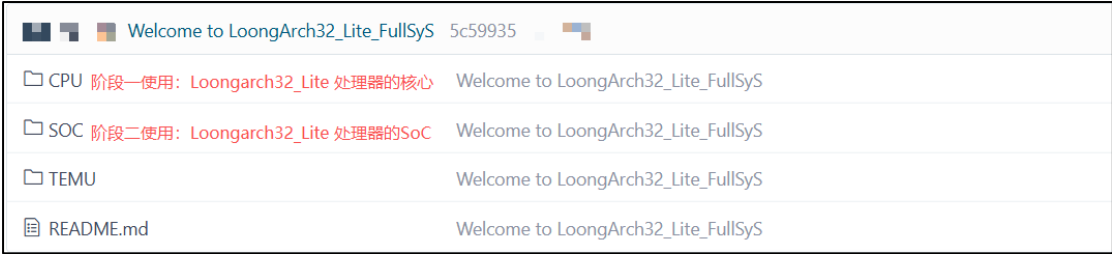
本实验将分为**两个阶段**完成，具体说明如下：

- 第一阶段：基于 Loongarch32 指令集，设计单周期/流水线 CPU 核。
- 第二阶段：将第一阶段完成的 CPU 核，集成到 SOC 系统中，并与若干外设连接以实现对应功能，最后通过测试程序的验证。

将本实验分为两个阶段的初衷是便于同学们循序渐进理解处理器与 SOC 系统的设计流程，并降低本实验的理解与操作难度，希望同学们可以认真完成，在实验中理解与体会课堂所学内容。

阶段一：CPU 核的设计与实现

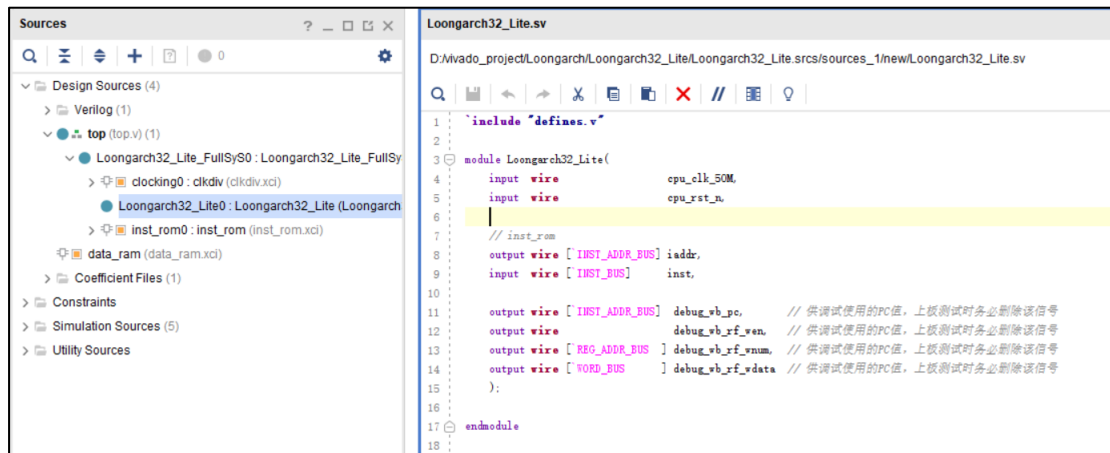
1. 本实验将继续使用**实验三中教师提供的 Gitee 仓库**中的文件，与实验四相关的文件解释如下：



- 各小组需结合自身实际选择**单周期处理器**或**流水线处理器**进行设计，两种处理器设计的最后评分将有区别。

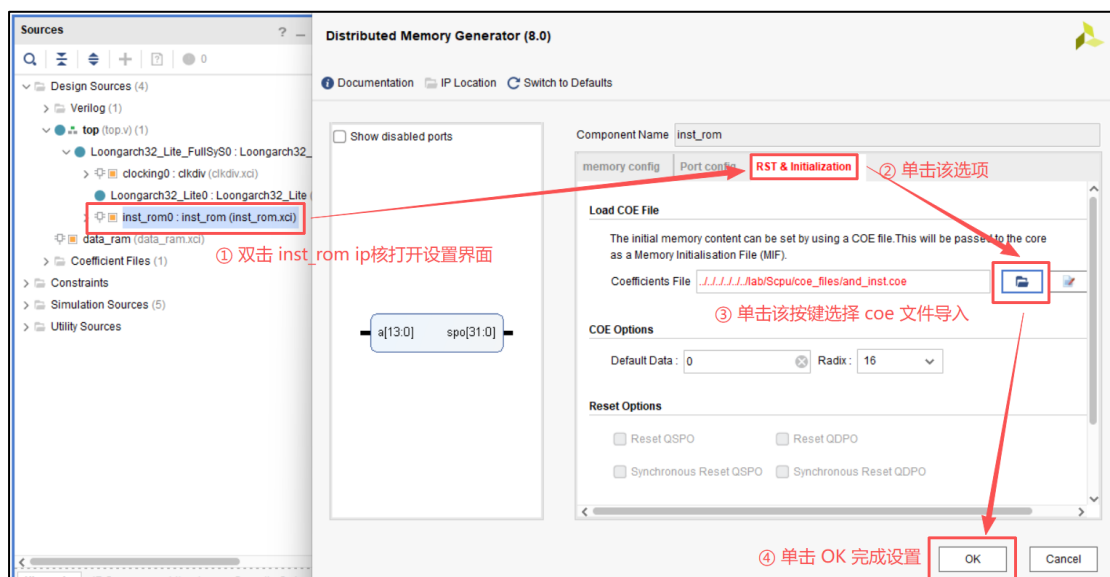
2. **单周期处理器**工程与设计说明：

① 实现单周期 CPU 核的文件 Loongarch32_Lite.sv 仅提供模块定义，同学们需要根据要求完成全部单周期处理器的相关设计：

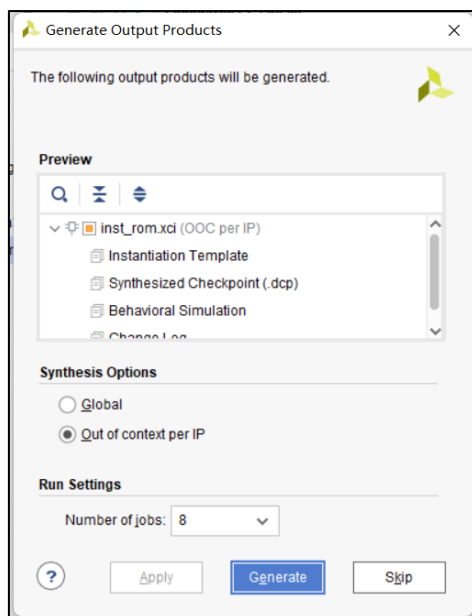


- top.v 为工程顶层文件，无需修改。
- Loongarch32_Lite_FullSys.sv 为 SOC 系统文件，本阶段仅需要在该文件实例化数据存储器 data_ram, 指令存储器 inst_rom 和 CPU 核已经为实例化完成。
- clkdiv.xci 文件为时钟分频 ip 核文件，无需修改。

② 指令存储器 inst_rom 在工程中已经完成实例化，只需要导入 coe 文件进行初始化即可使用：

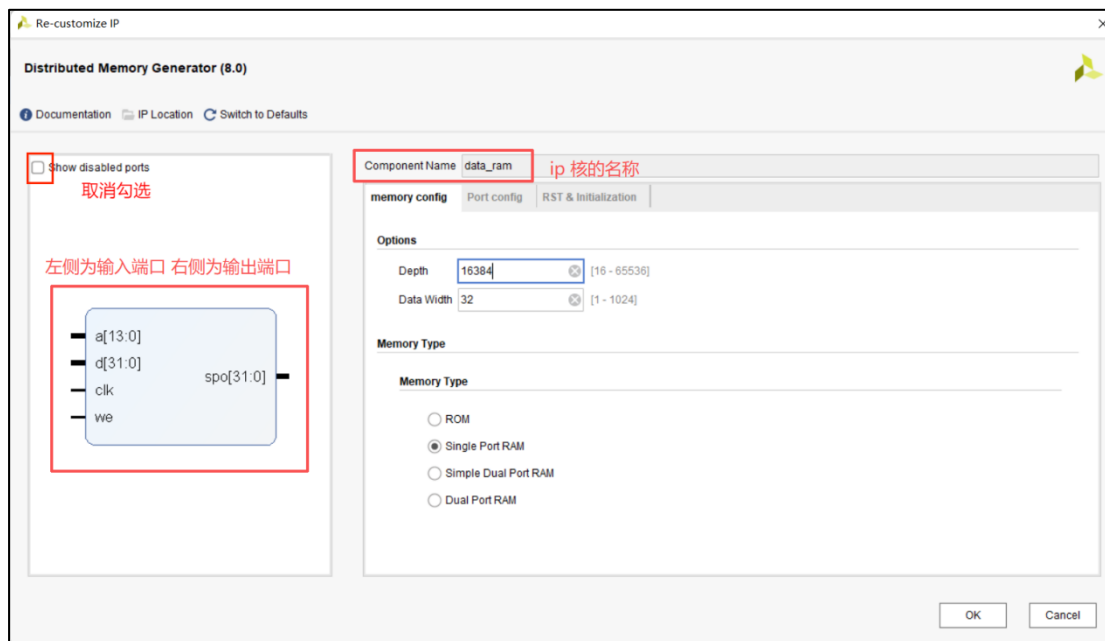


- 在弹出的界面点击“Generate”即可进行 ip 核初始化：



• 本阶段需要同学们在实验三完成的 temu 仿真器中自己编写测试程序，生成测试程序的 coe 文件进行导入，具体可参考实验三说明。

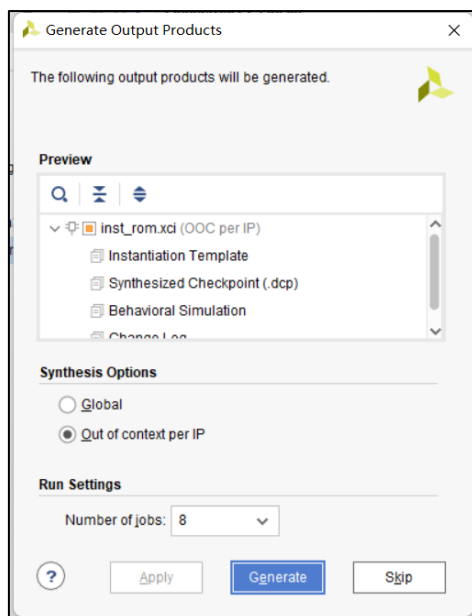
③ 数据存储器 data_ram 在工程中提供了未进行实例化的 ip 核，需要同学们在 Loongarch32_Lite_FullSyS.sv 文件进行实例化，data_ram 相关说明如下图：



• 同时数据存储器也需要导入 coe 文件进行初始化，方法与要求同 inst_rom。

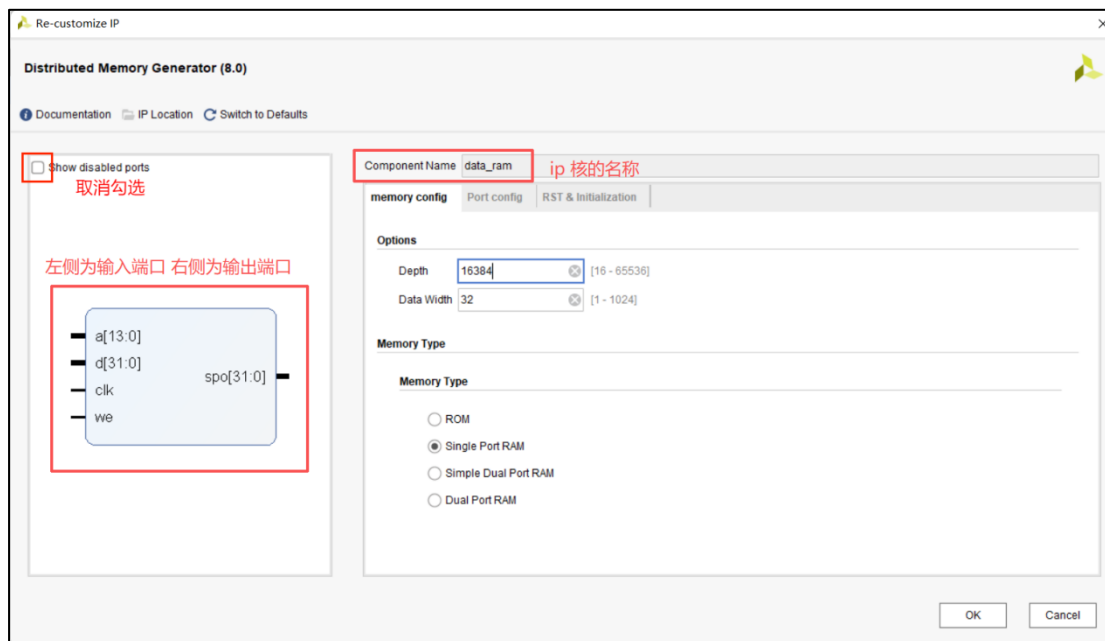
3. 流水线处理器工程与设计说明：

① 流水线 CPU 核（LoongArch32_Lite 模块）已经为各位提供基于五级流水线的一条指令的实现，同学们需要根据要求完成剩余流水线处理器的相关设计：



• 本阶段需要同学们在实验三完成的 temu 仿真器中自己编写测试程序，生成测试程序的 coe 文件进行导入，具体可参考实验三说明。

③ 数据存储器 data_ram 在工程中提供了未进行实例化的 ip 核，需要同学们在 Loongarch32_Lite_FullSyS.sv 文件进行实例化，data_ram 相关说明如下图：



• 同时数据存储器也需要导入 coe 文件进行初始化，方法与要求同 inst_rom。

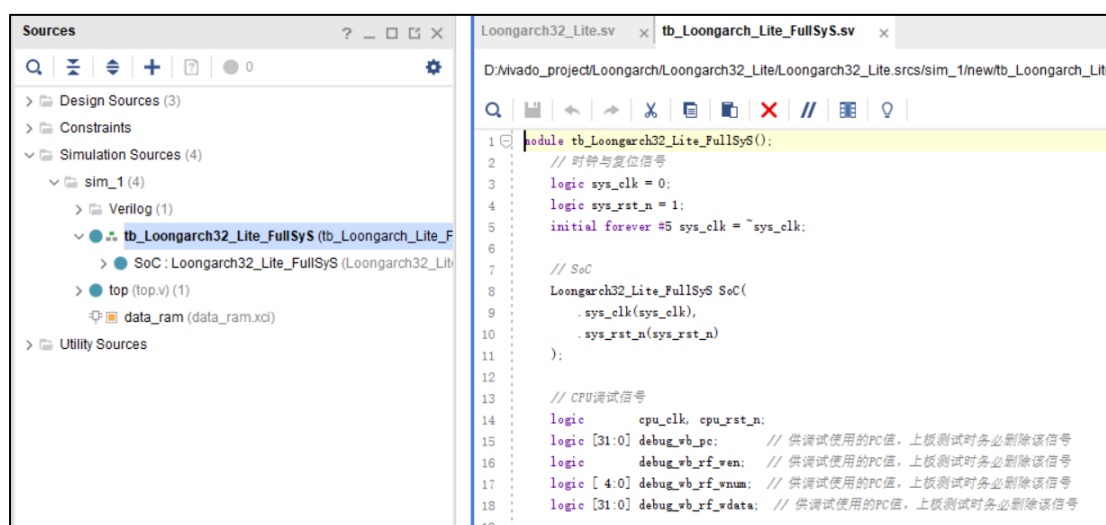
4. 本阶段验收说明：

① 本阶段无需在远程 FPGA 平台进行上板验证，只需要通过仿真验证处理器的功能是否完善即可。

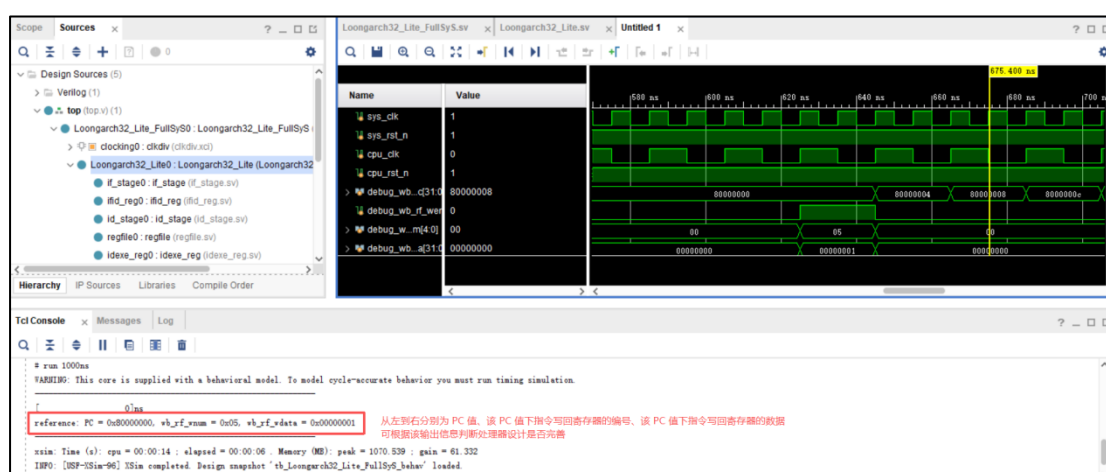
② 在仿真之前，**请确认完成了以下内容**：

- 单周期/流水线 CPU 的相关设计，满足 **14 条必做指令+3 条随机指令**。
- 实例化数据存储器 data_ram（仅在实现完访存指令时）。
- 在指令存储器 inst_rom 和数据存储器 data_ram 中导入了测试程序对应的 coe 文件。

③ 单周期/流水线工程已经提供了仿真文件 tb_Loongarch32_Lite_FullSyS.sv，该仿真文件可直接运行，随后可通过**仿真波形图**或**输出的 trace 信息**判断处理器的功能是否完善。



• 仿真输出的 trace 信息示例如下图：



• 确认处理器确实已经可以**完全正确运行 17 条要求的指令**时，本阶段即结束，可进行第二阶段的 SOC 集成。

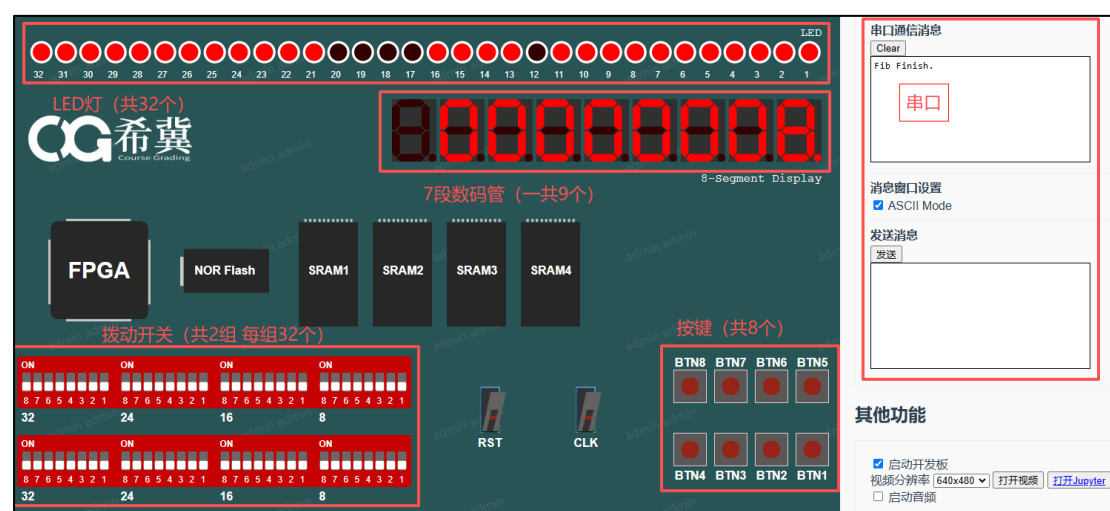
阶段二：SOC 的设计与实现

1. 在进行本阶段前，请确定完成以下内容：

- ① 已经实现了基于 Loongarch32 指令集的**单周期/流水线**处理器。
- ② 处理器已经可以满足 **14 条必做指令+3 条随机指令**的要求。

2. SOC 工程文件说明：

- ① SOC 系统文件为 Loongarch32_Lite_FullSyS. sv。
- ② 该文件内已经提供**若干外设的实现逻辑（仅供参考）**，本阶段所有外设均基于远程 FPGA 平台的部件设计，如下图所示：



• 以上部件在实验一与实验二均有涉及，如何使用可参考对应实验说明。

- ③ 本阶段需要在 Loongarch32_Lite_FullSyS. sv 文件内实例化 **CPU 核**（阶段一已完成）、**指令存储器**（已经提供 ip 核，实例化即可）、**数据存储器**（已经提供 ip 核，实例化即可），并实现与若干外设的连接。

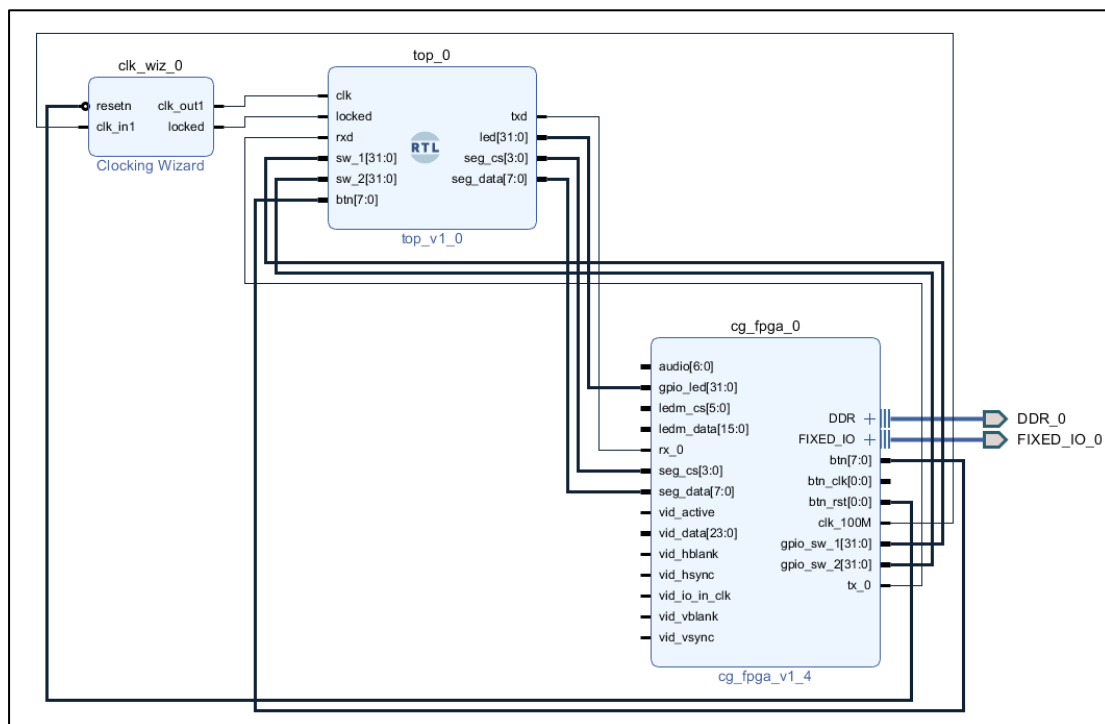
3. 本阶段验收第一部分：

- ① 本阶段首先要通过**提供的 benchtest 测试程序**验证：

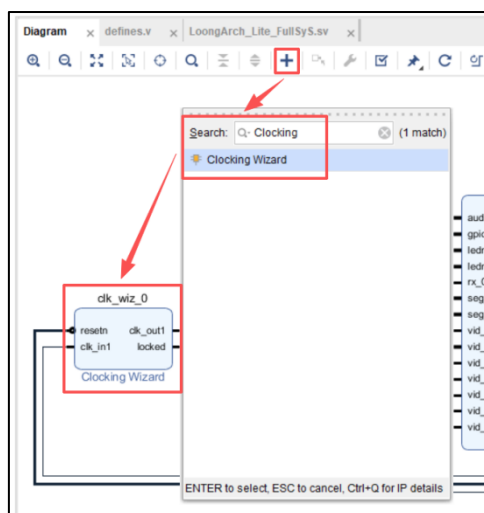
- 提供测试程序原文件 benchtest_code 压缩包，需要仔细阅读，理解测试程序的原理，否则连接外设将比较困难。
- 提供 benchtest.s 反汇编文件 benchtest.asm，供同学们理解测试程序。
- 本测试程序**必须要在 SOC 中实现串口外设**，但工程仅提供回环测试代码，需要同学们**自行修改该外设的实现逻辑**以达成和测试程序适配的目的。
- 提供**指令存储器 coe 文件**（inst_rom.coe）与**数据存储器 coe 文件**

(data_ram_X.coe X 为小组编号)，可直接导入对应存储器进行初始化。

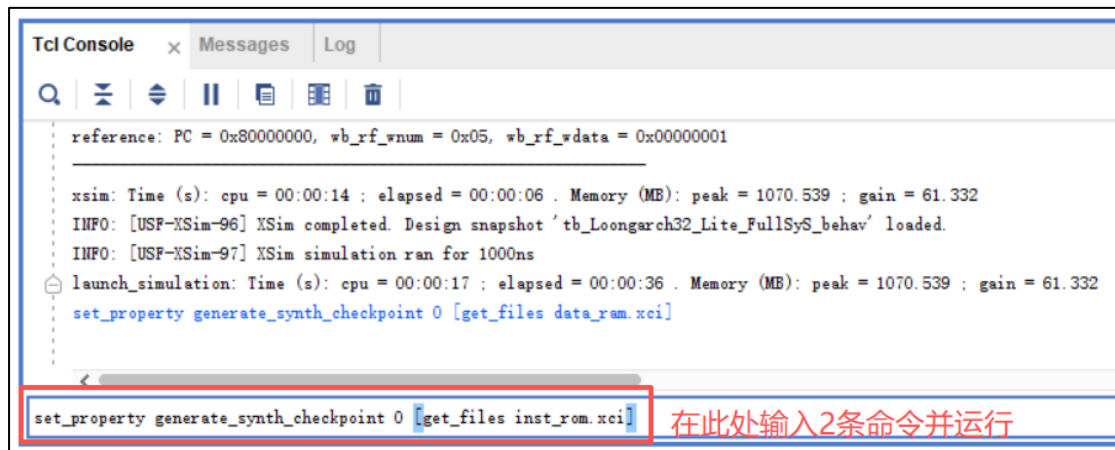
② 完成以上步骤后，参考实验一与实验二进行**项目综合操作**，bd 文件如下图：



• 与实验一和实验二不同，本次新增时钟分频 ip 核，可通过下图方法添加：



• 该方式导入的时钟分频 ip 核需要进行如下设置：



• 完成后 2 个存储器的图标会发生变化，**请确认完成**，否则项目综合会因为 top 模块内部包含 ip 核而出错：

```
> data_ram (data_ram.xci)
> inst_rom (inst_rom.xci)
```

④ 综合项目生成 bit 文件后烧写到远程 FPGA 平台，进行 **benchtest 测试程序验证**，测试步骤：

- 烧写完成后自动进行斐波那契数列计算。
- 串口输出“Fib Finish.”代表斐波那契数列计算正确。
- 通过串口输入“T”开始随机指令验证。
- 串口输出“All PASS!”代表随机指令测试通过。
- 以上步骤均达成即完成本测试程序测试！

4. 本阶段验收第二部分：

- 本阶段其次要通过**自己编写的测试程序**验证。
- 测试程序要求**至少使用一种非串口外设**，并修改对应外设的实现逻辑。
- 项目综合过程**同 benchtest 测试工程**，最后在远程 FPGA 平台验证效果符合测试程序预期即算完成！