****

**实验3**

**MIPSfpag系统：自定制接口模块**



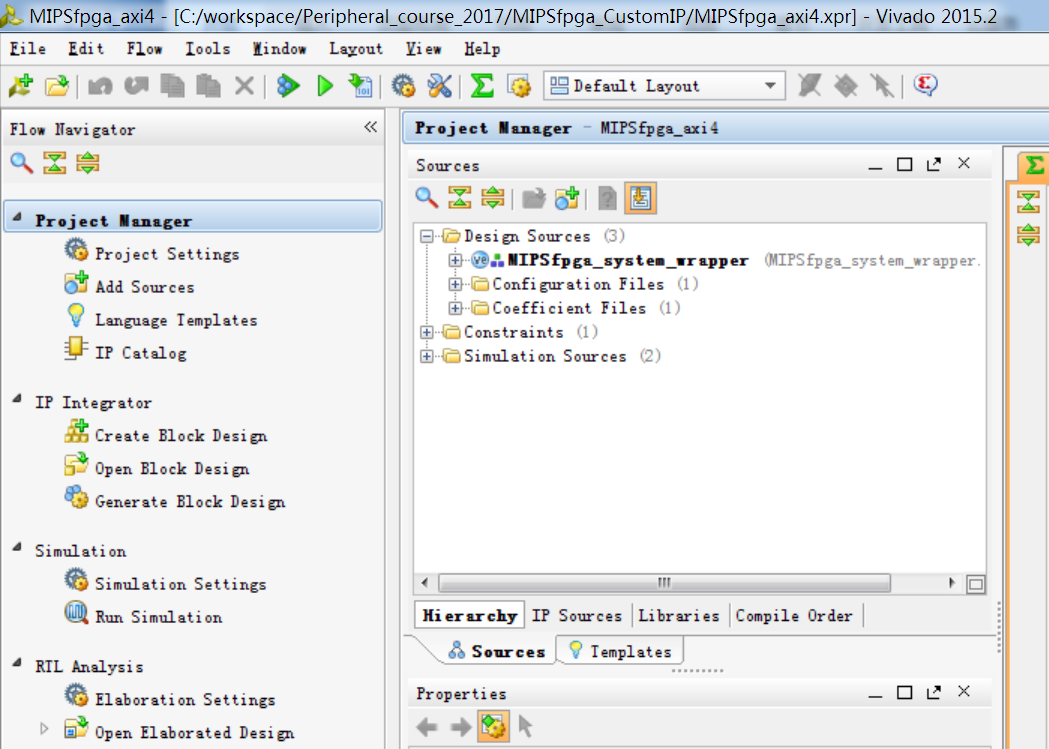
**实验3：MIPSfpga系统之自订定制接口模块**

# 一、概述

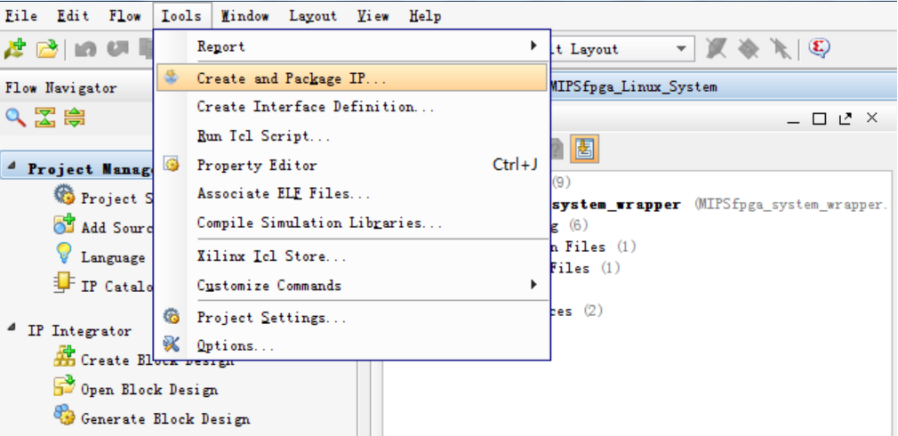
本实验将学习如何自己定制一个基于AXI4接口的外设模块。首先通过Vivado提供的自定制IP模板实现一个自定制的接口模块；然后，将这个自定制的接口模块添加到实验2完成的MIPSfpga处理器系统中；最后，编写MIPS C语言程序对这个新添加的自定制外设进行操作演示。

# 二、MIPSfpga处理器系统搭建

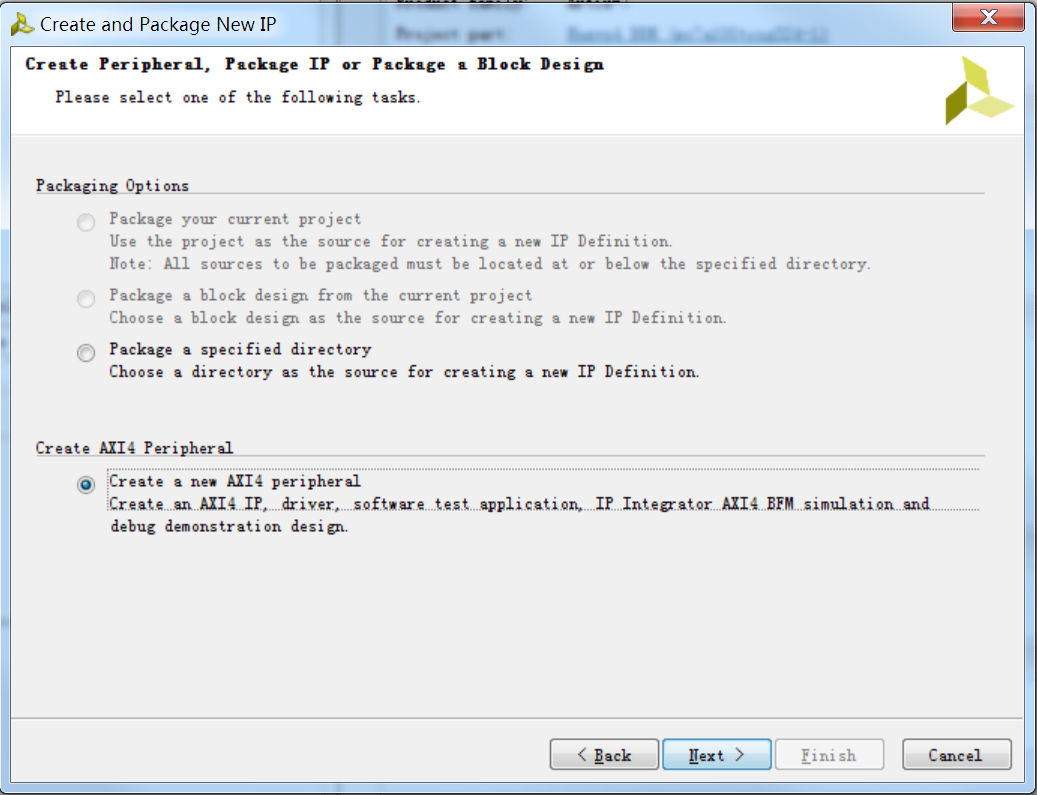
1. 复制自己完成的实验2的工程，并将其目录从MIPSfpga\_uart更名为MIPSfpga\_CustomIP，启动Vivado，然后打开该工程。（因为只修改了该工程的目录名称，因此该工程的名称仍然是MIPSfpga\_axi4）



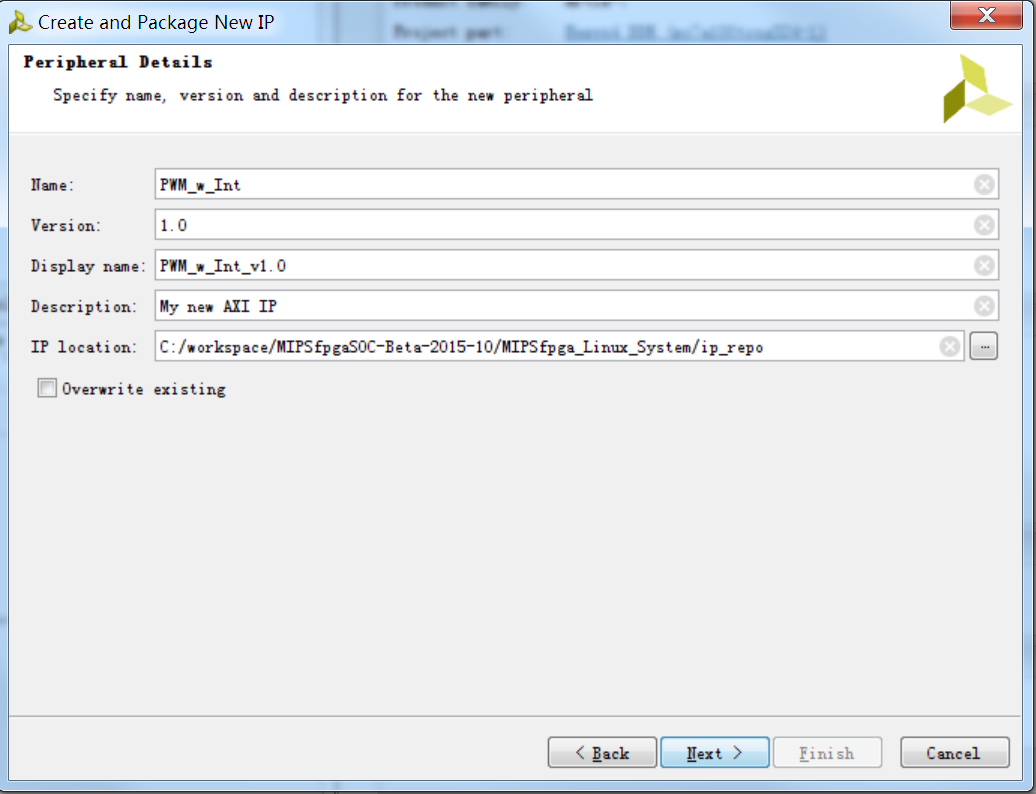
1. 选择Create and Package IP菜单



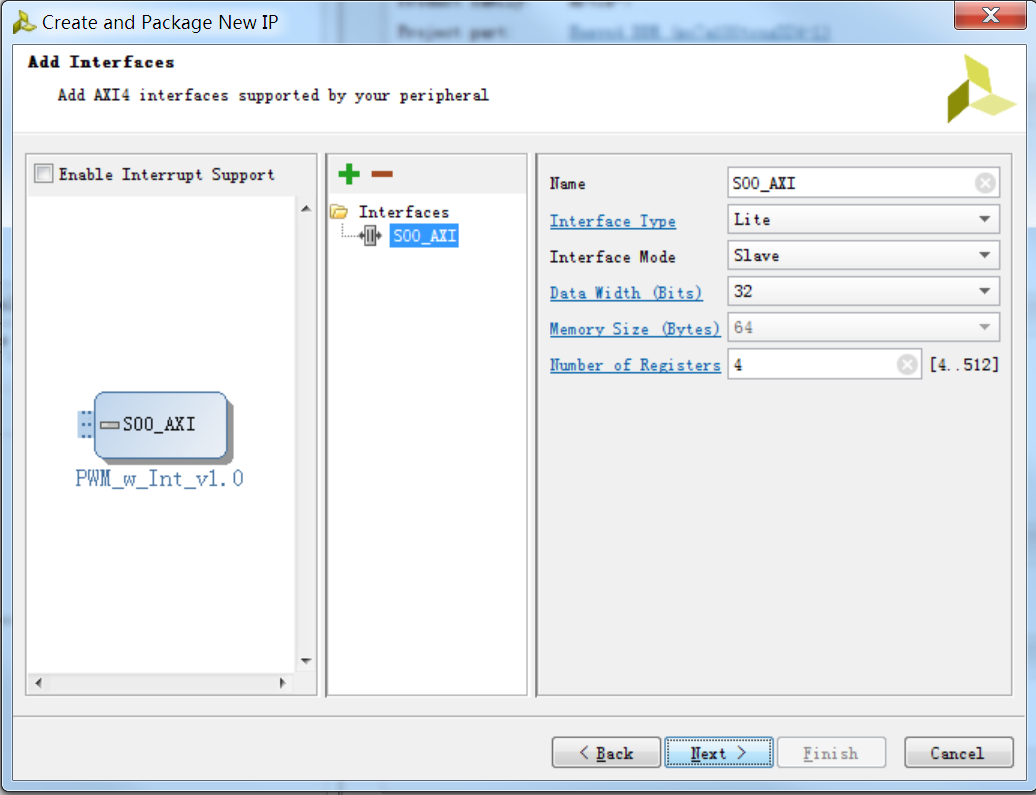
1. 点击Next后选择Create a new AXI4 peripheral选项



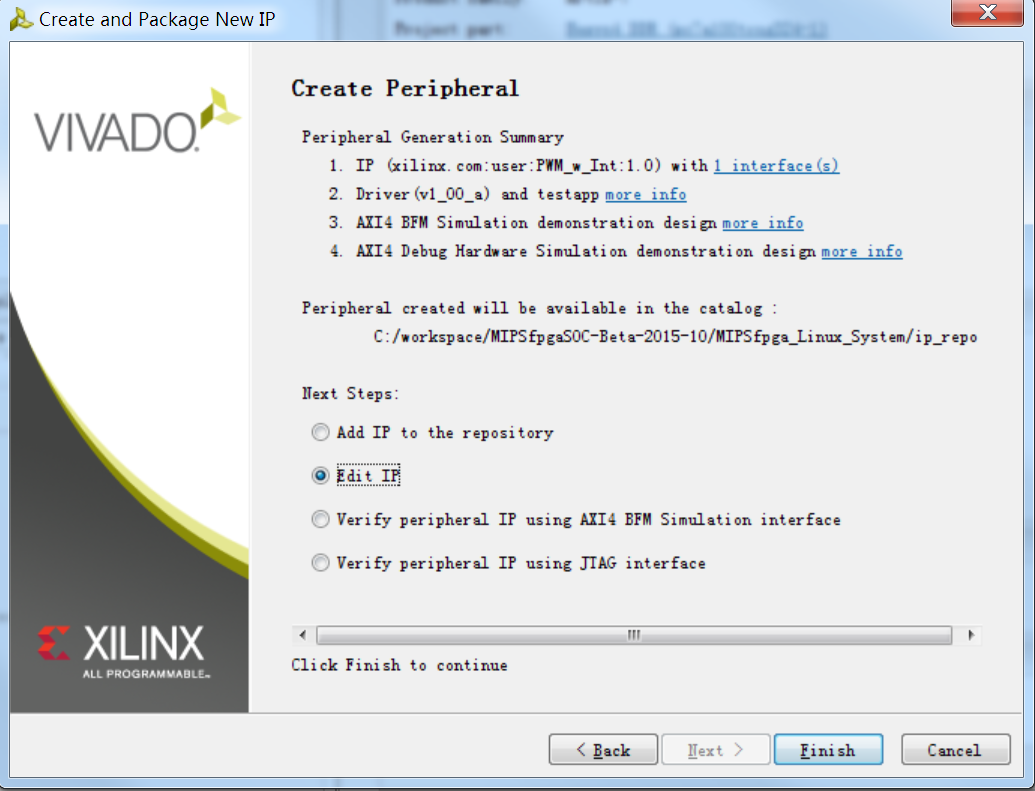
1. 输入IP模块的名称、版本号等信息。



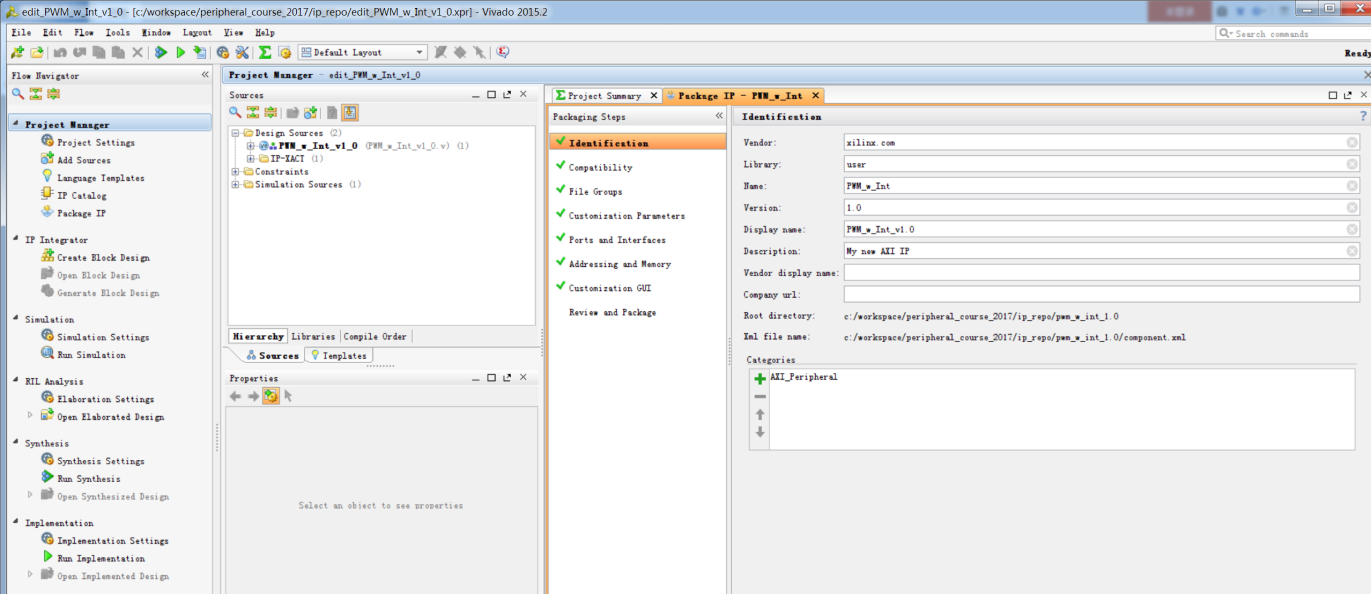
1. 设定该IP模块AXI4接口的类型和参数



1. 选择Edit IP选项，然后点击Finish。

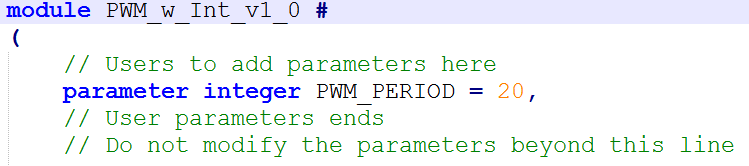


1. 此时会弹出另一个Vivado界面，其工程名称为edit\_PWM\_w\_Int\_v1\_0，位于/ip\_repo目录下。

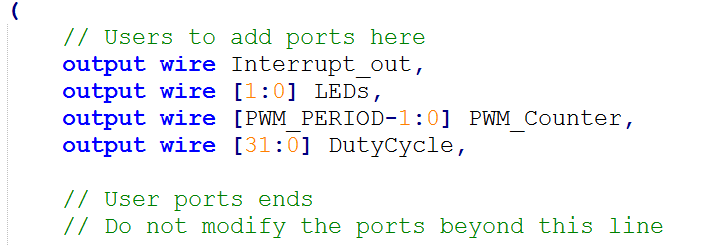


1. 在edit\_PWM\_w\_Int\_v1\_0中打开PWM\_w\_Int\_v1\_0.v文件，按照下图对该文件进行编辑。

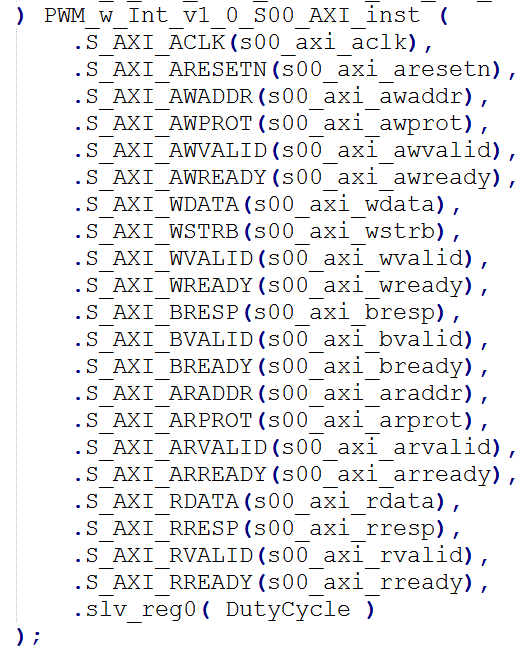
编辑位置1：



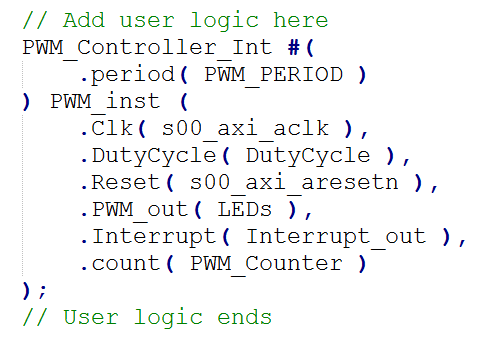
编辑位置2：



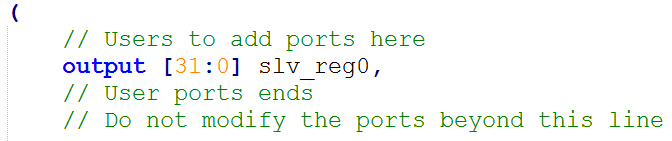
编辑位置3：



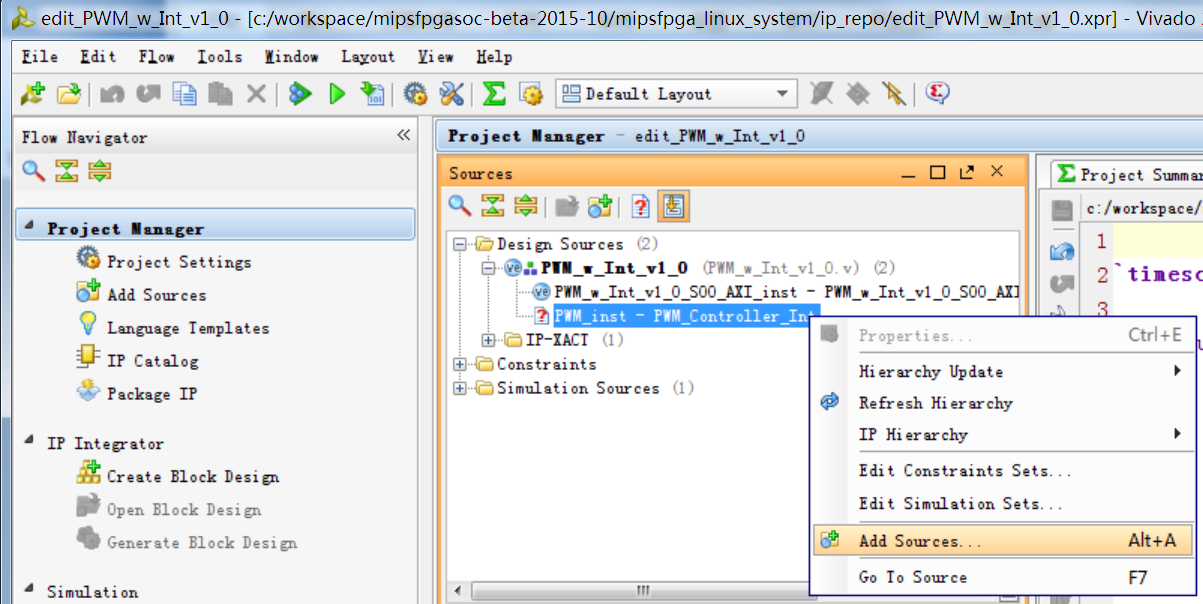
编辑位置4：



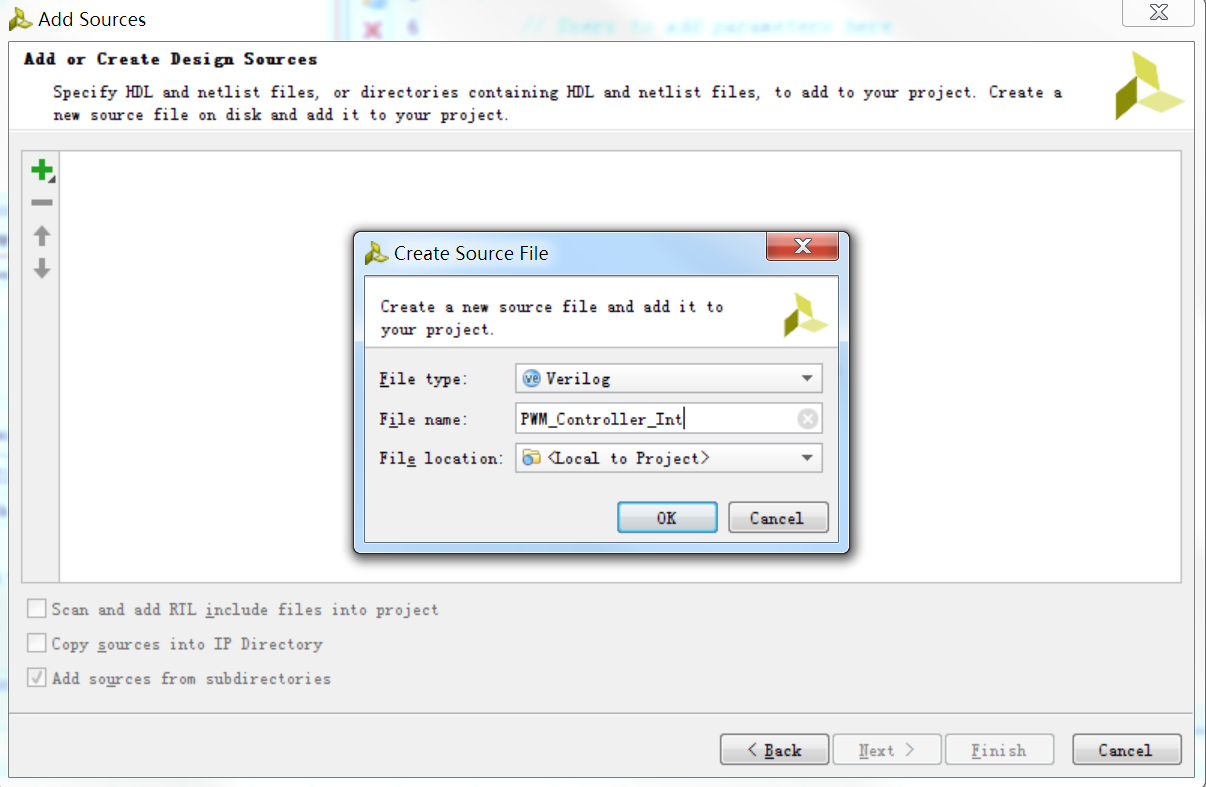
1. PWM\_w\_Int\_v1\_0.v文件编辑完成后，再打开下层的PWM\_w\_Int\_v1\_0\_S00\_AXI.v文件，按照下图对该文件进行编辑。



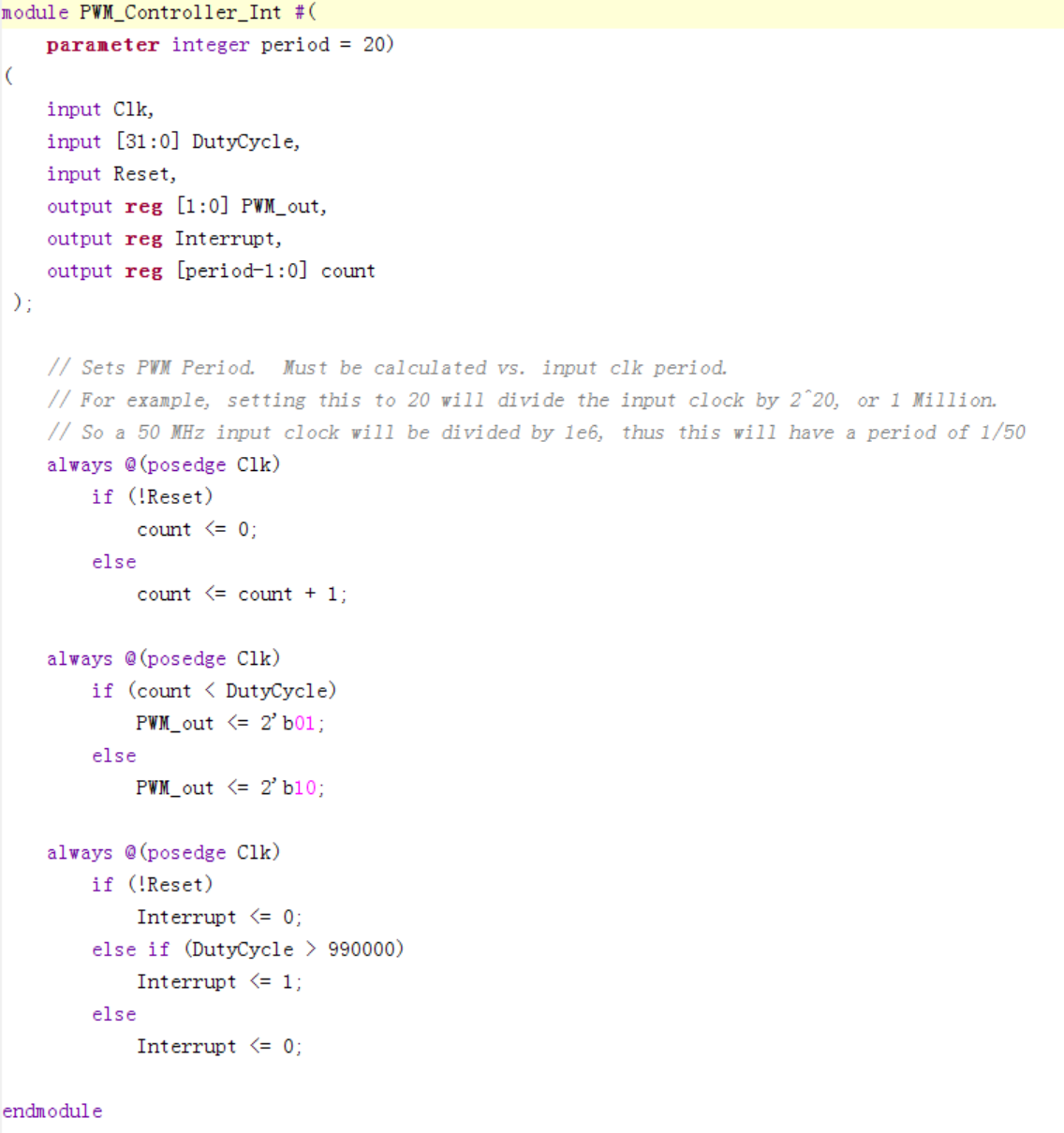
1. 选择Add Source



1. 新建一个叫做PWM\_Controller\_Int的设计文件

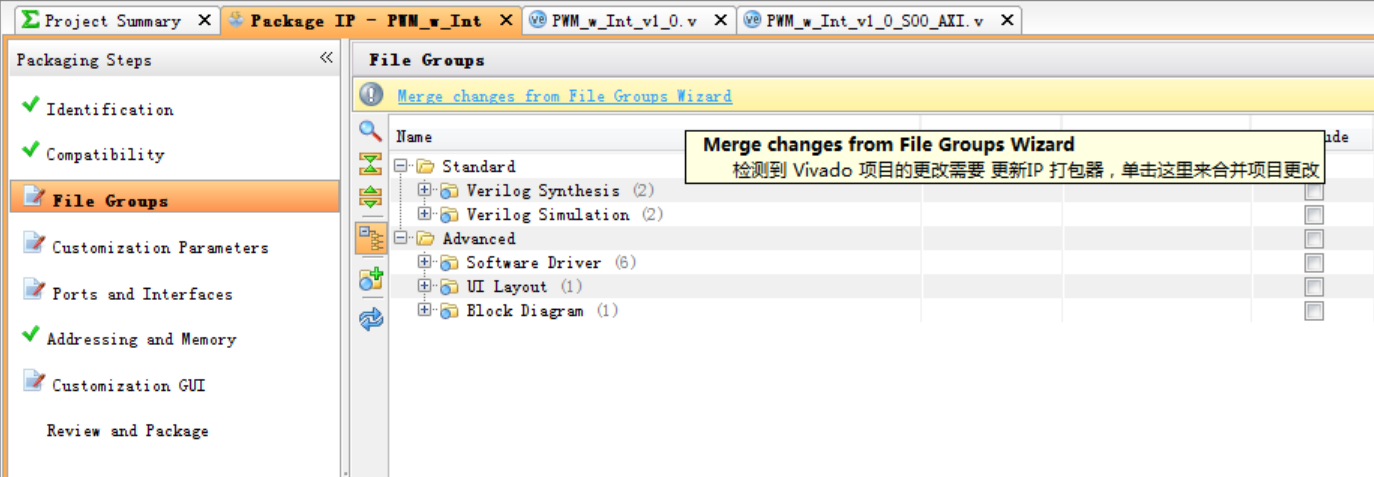


1. 按照下图编辑PWM\_Controller\_Int.v文件

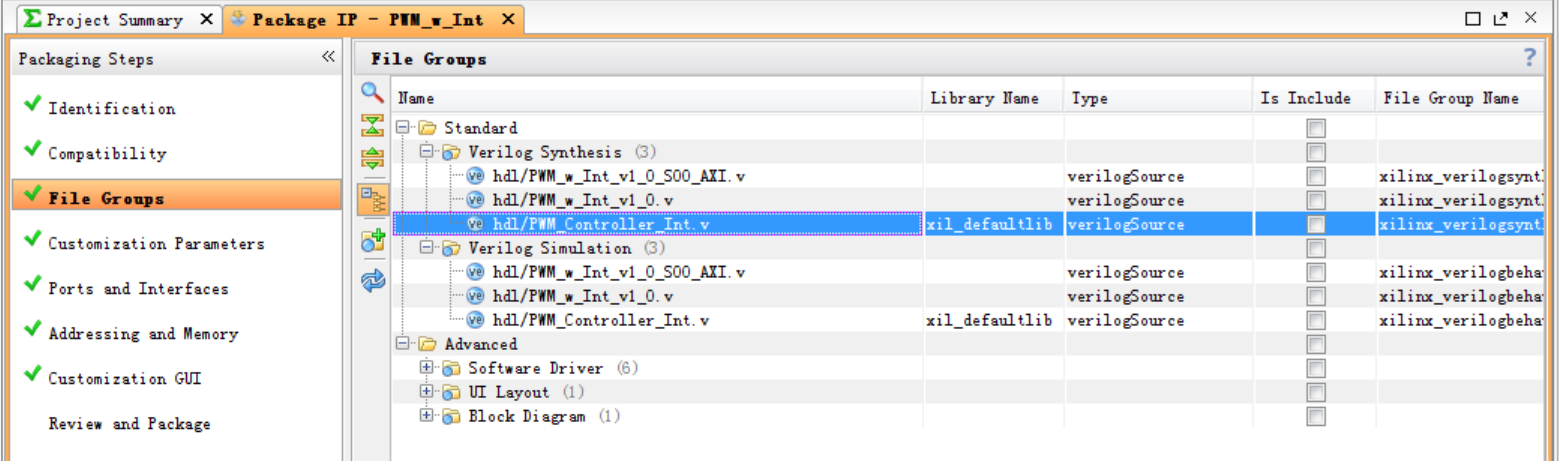


1. 可以运行综合（Synthesis）和仿真（Simulation）对该模块设计的正确性进行验证。
2. 找到Package IP的File Groups选项，将PWM\_Controller\_Int.v文件加入。

如果File Groups选项上不是绿色的钩，选择Merge changes from File Groups Wizard自动加入该文件：

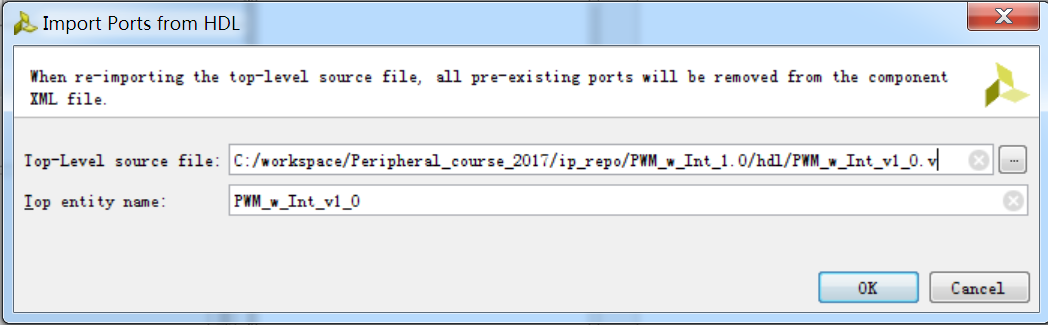


如果File Groups选项上已经是绿色的钩，则手动加入该文件：

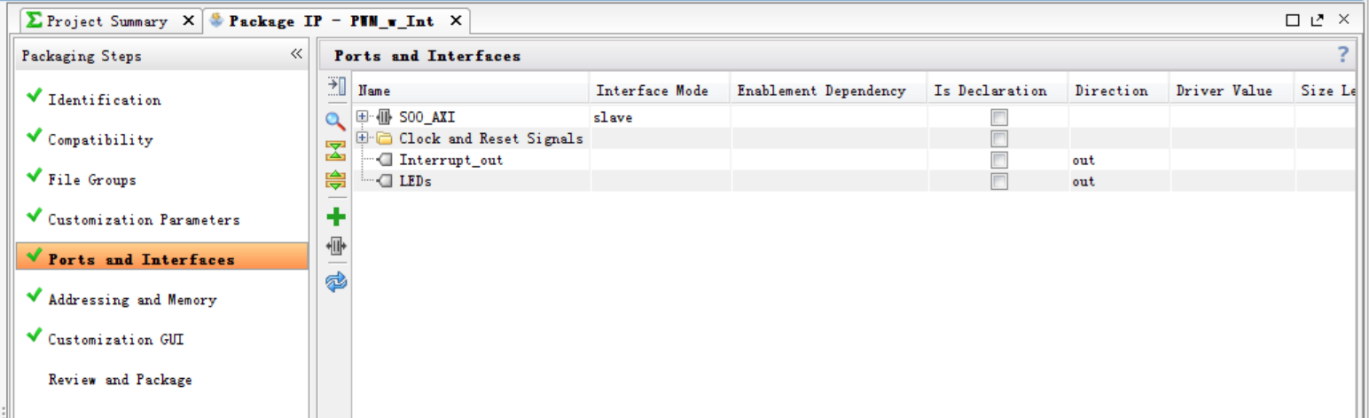


1. 将模块的Interrupt\_out和LEDs信号设置为输出引脚

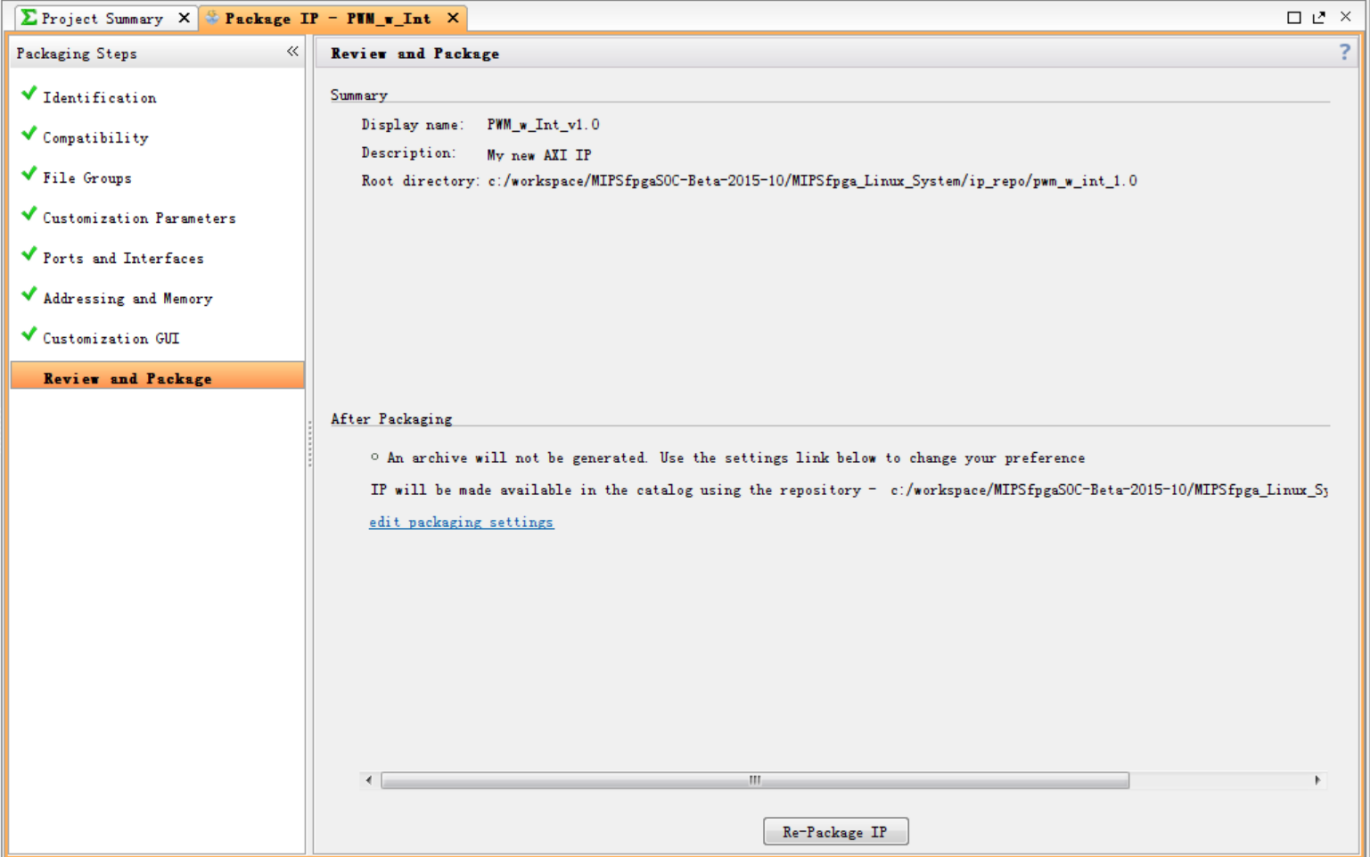
如果Ports and Interfaces选项上不是绿色的钩，点击自动加入；否则点击鼠标右键，选择Import IP Ports，找到模块的顶层文件：



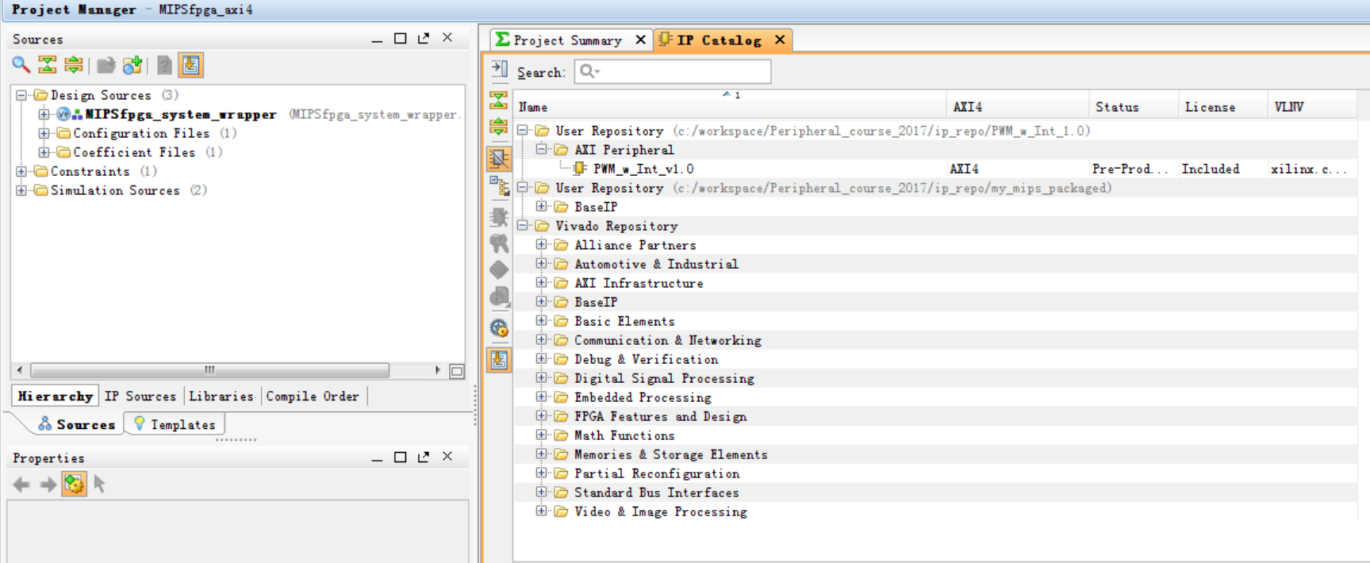
只留下Interrupt\_out和LEDs，去除其它引脚：



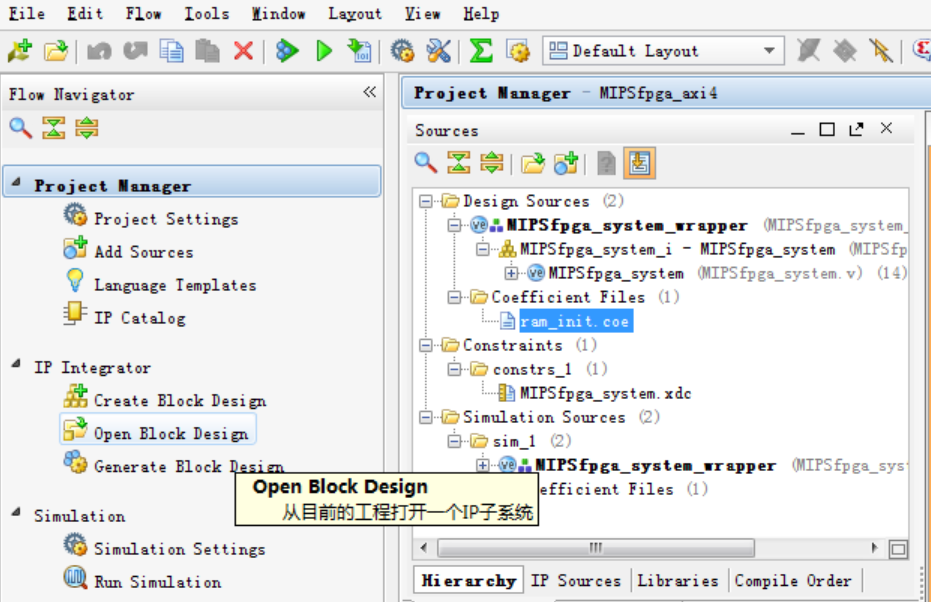
1. 如果还有其它不是绿色钩的选项，点击相应的操作将其去除。最后选择Click Review and Package选项，点击Re-Package IP按钮，完成IP的封装。



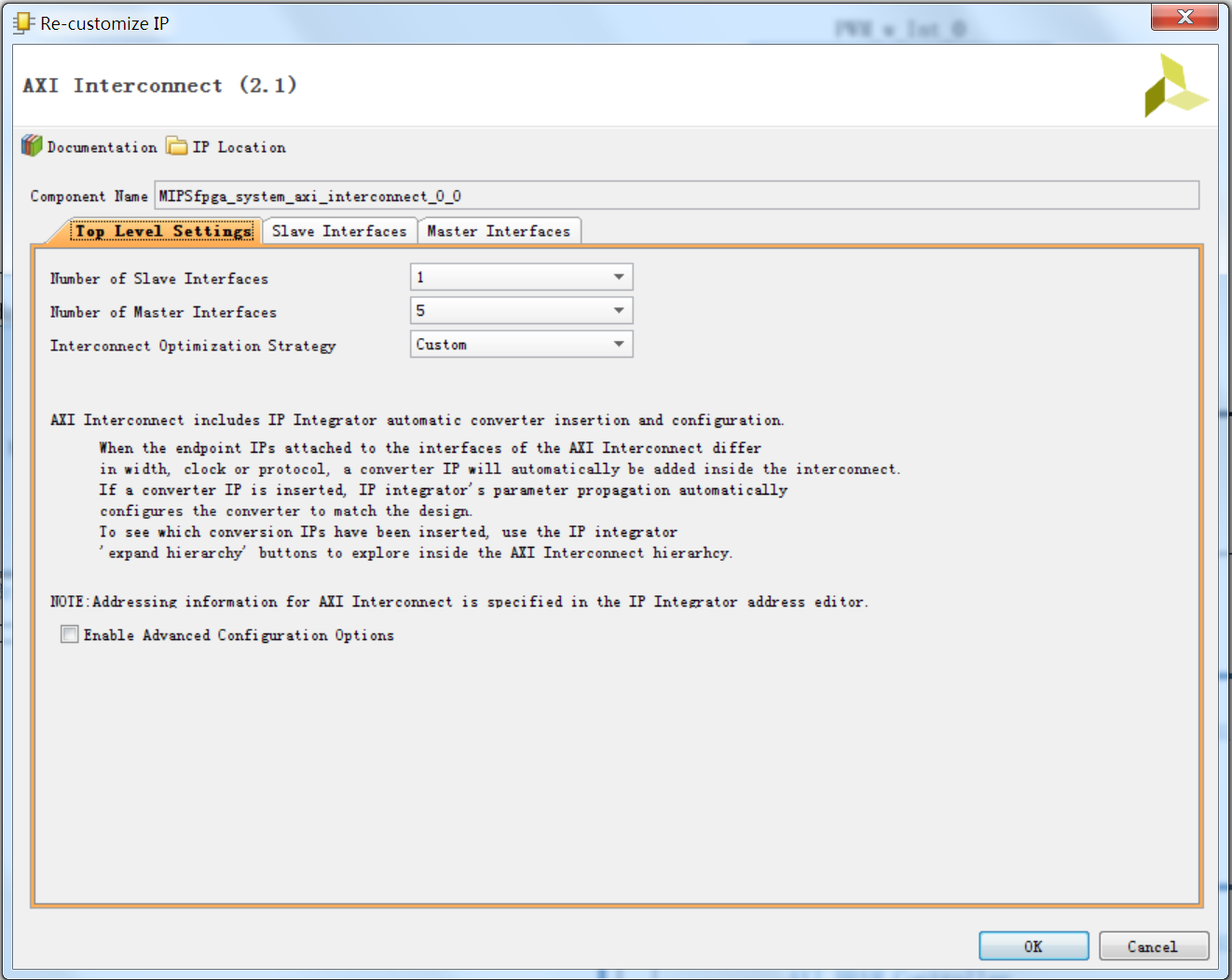
1. IP封装完成后，关闭该Vivado工程。此时将在主工程的 IP Catalog可以找到名为PWM\_w\_Int\_v1.0的用户自订制IP外设模块。

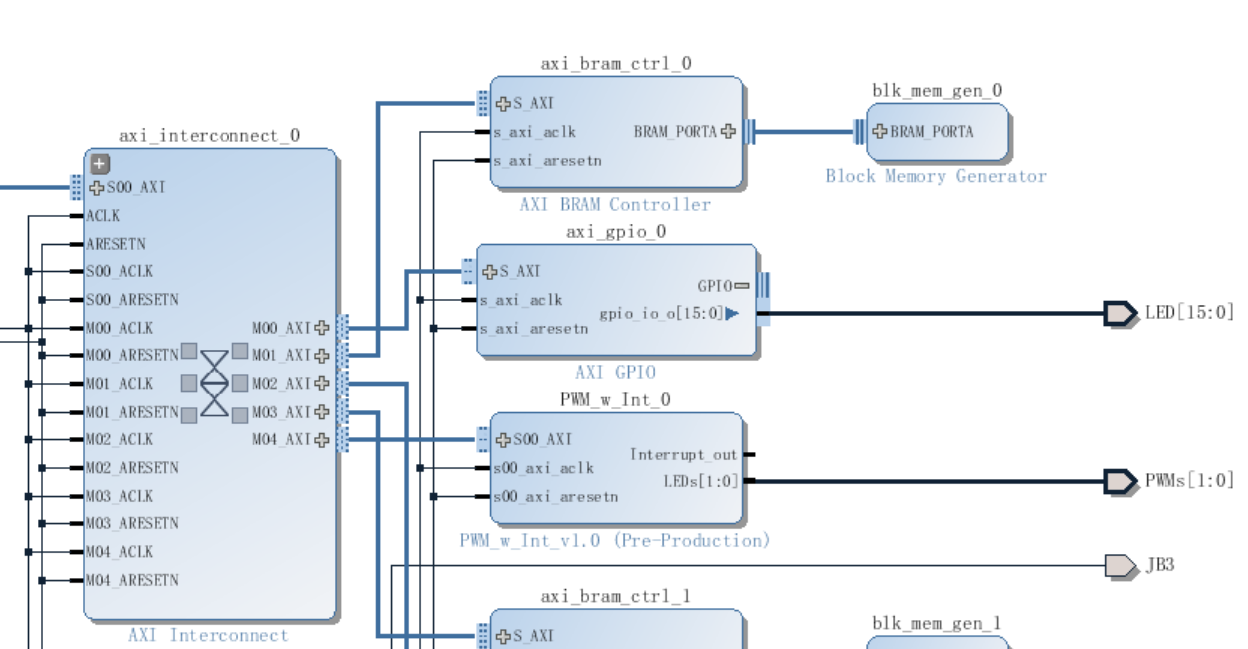


1. 点击Open Block Design打于先前的设计。

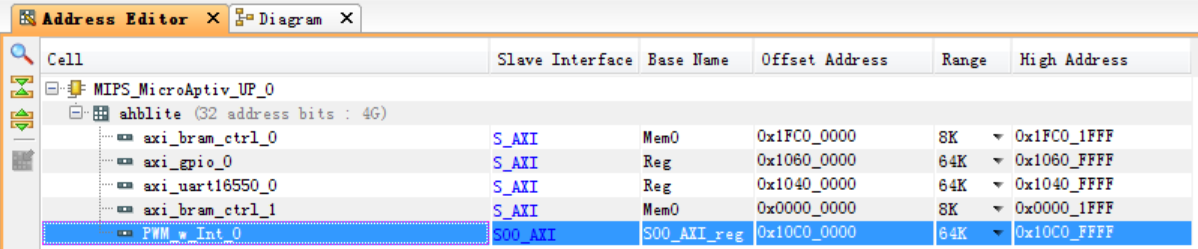


1. 添加PWM\_w\_Int\_v1.0用户自订制的IP模块，并增加一个AXI Interconnect模块的主端口，然后按照下图连接到MIPSfpga处理器系统。

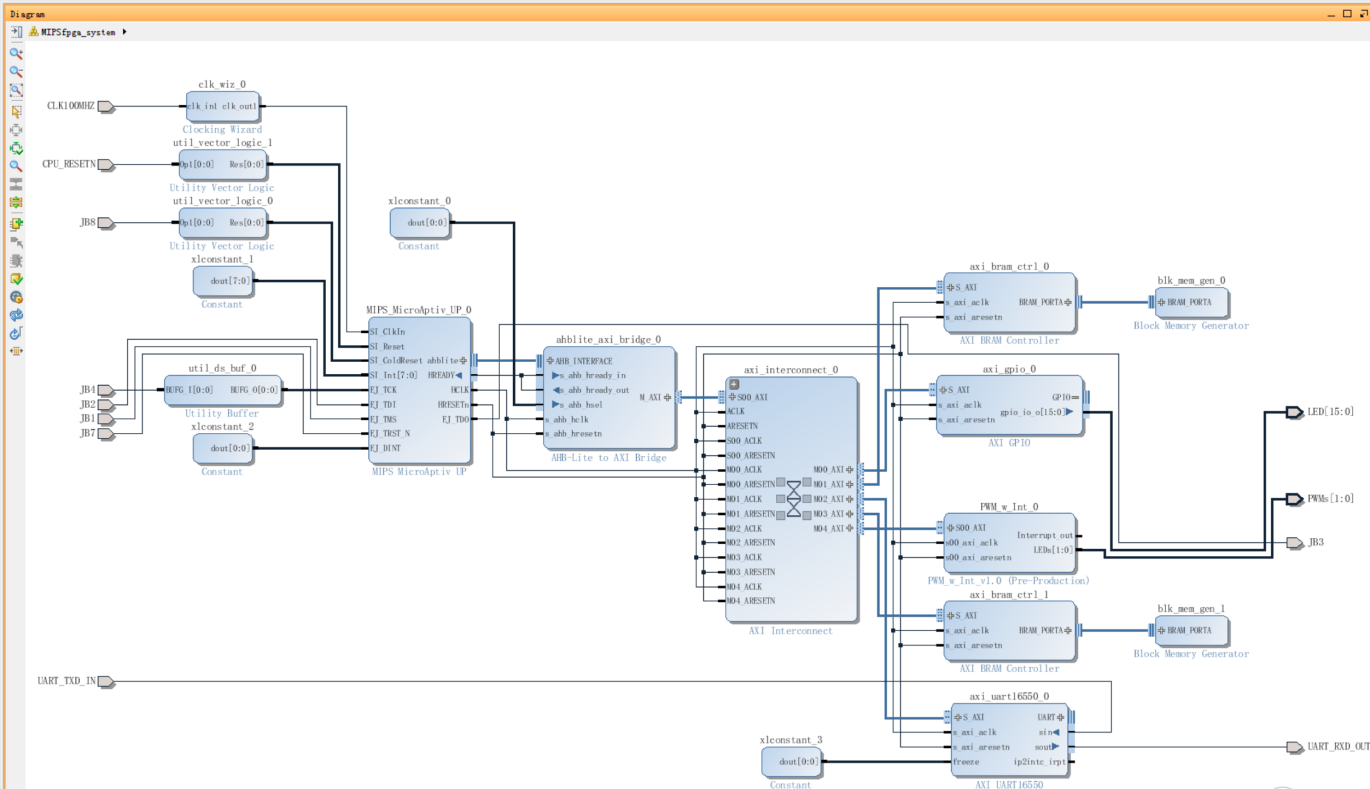




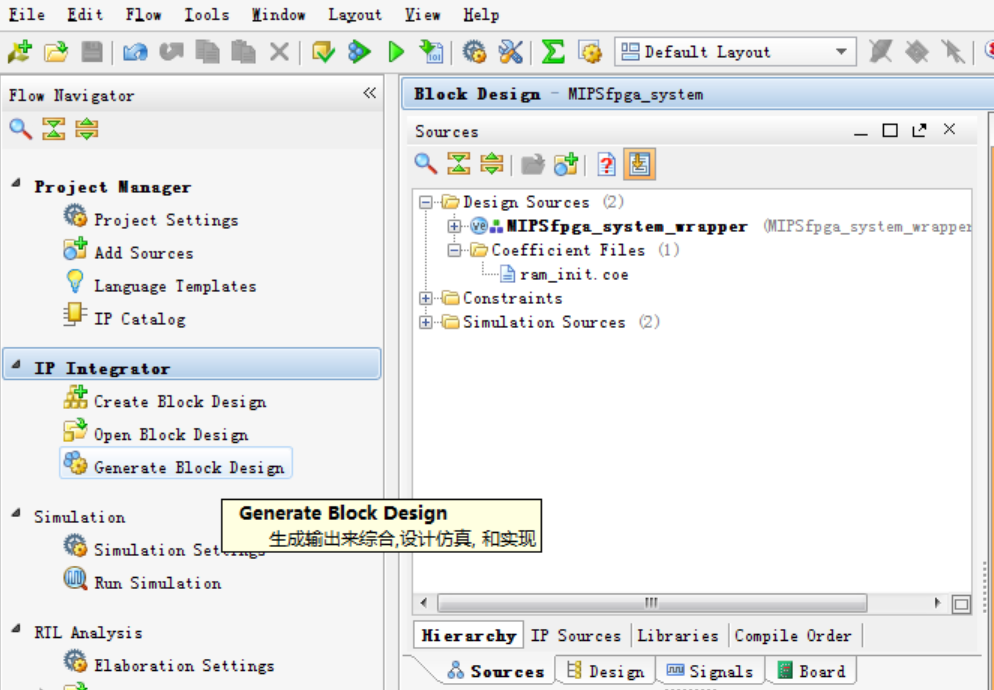
1. 打开Address Editor，将PWM\_w\_Int\_v1.0模块的地址设置为0x10C0\_0000。

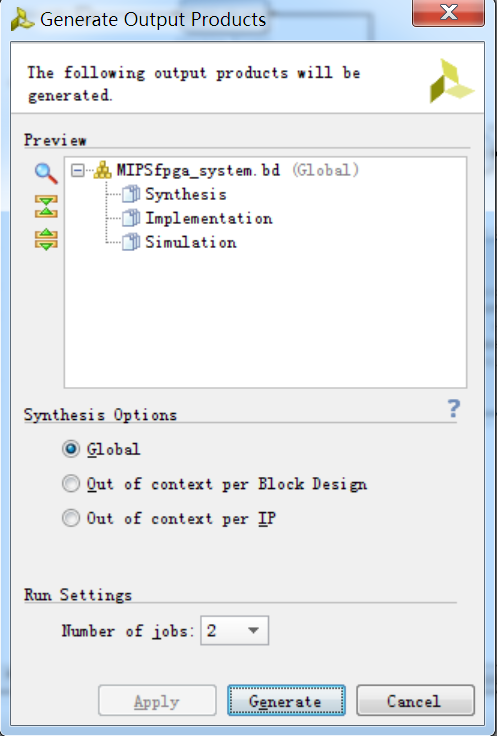


1. 此时整个MIPSfpga处理器系统如下图所示。



1. 点击Validate Design，对设计的正确性进行校验。校验过程中如果出现警告，点击OK忽略。
2. 点击Generate Block Design，弹出对话框后选择Generate更新MIPSfpga\_system\_wrapper文件。





1. 修改约束文件，绑定新添加的引脚PWMs。最后，点击Generate Bitstream按键，生成bitstream文件。

# 三、应用程序编译、调试和执行

1. 打到MIPSfpga\_CustomIP\_C目录，在该目录下用鼠标右键选择打开cmd命令窗口。在该命令窗口中输入make进行编译生成elf可执行文件。使用make clean命令可以将编译的结果清除。
2. 连接Nexys4 DDR开发板bit文件下载线缆和MIPSfpga调试器。
3. 打到Codescape\_Scripts目录，在该目录下用鼠标右键选择打开cmd命令窗口。在命令窗口中输入如下命令运行loadMIPSfpga.bat批处理文件：

loadMIPSfpga.bat C:\workspace\MIPSfpga\_Peripheral\_2017\MIPSfpga\_CustomIP\_C

1. 同时打开一个串口终端，将波特率设置为115200，操作串口同时观察PWM模块指示灯的输出。

# 四、动手实践

1. 自己封装一个用于数码管显示的IP模块，并将该模块添加到MIPSfpga\_CustomIP工程实现的MIPSfpga处理器系统中，生成相应的比特流文件。
2. 修改MIPSfpga\_CustomIP\_C目录中的相关文件，编写一个可以用于数码管显示接口模块控制的演示程序。