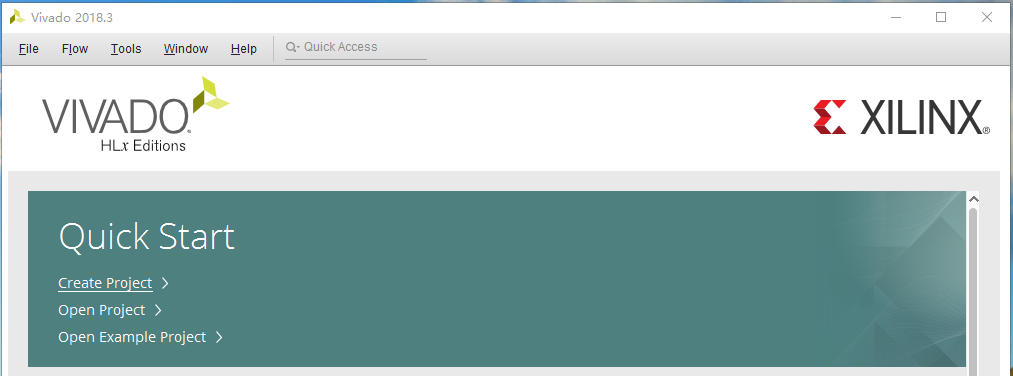
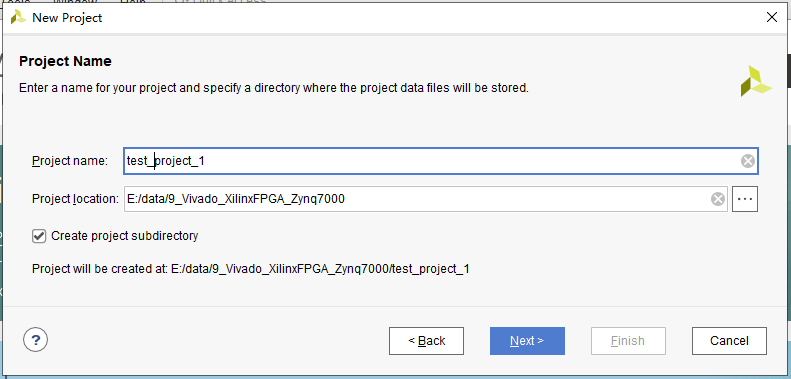
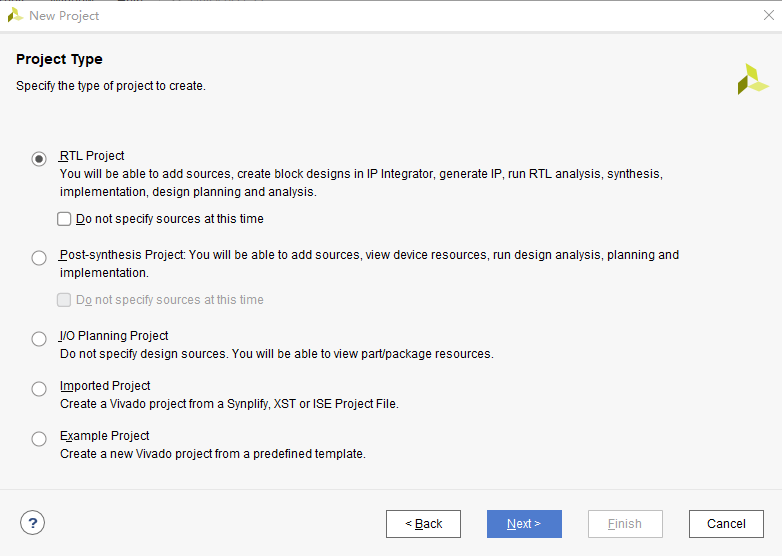
# 基本设计流程

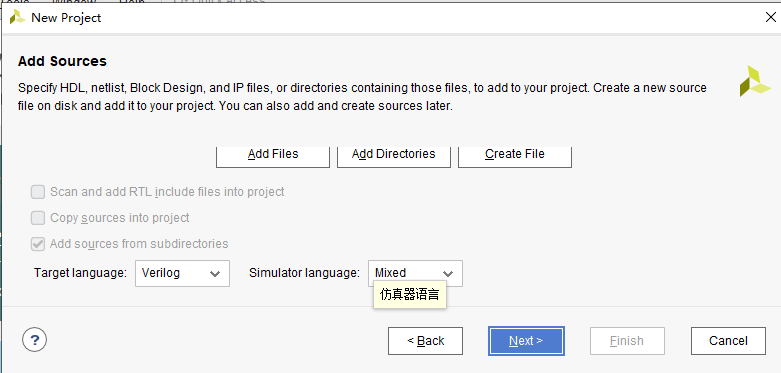
将板级支持包复制到\xilinx\vivado\2018.3\data\boards\board\_files

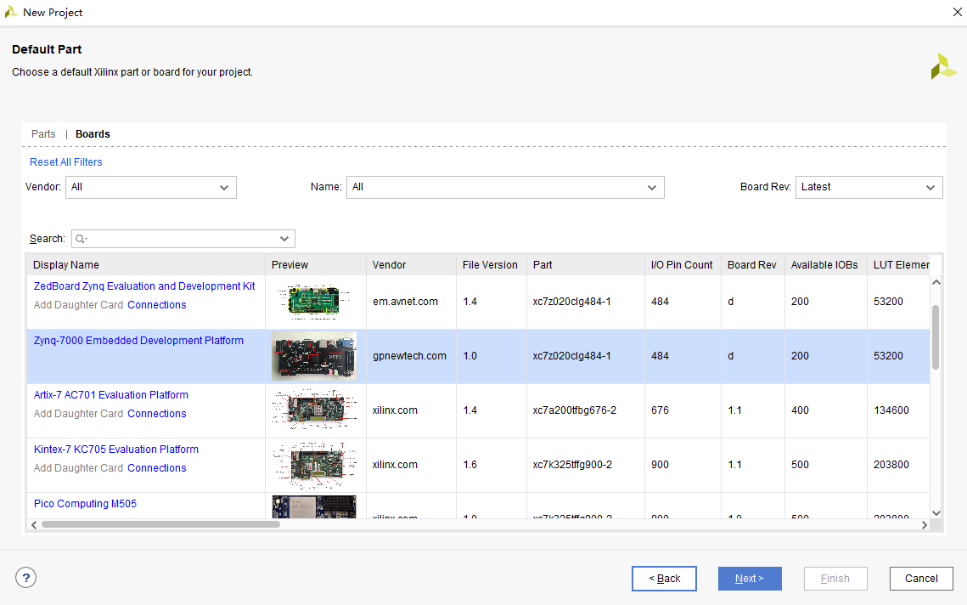
## 一、创建新工程





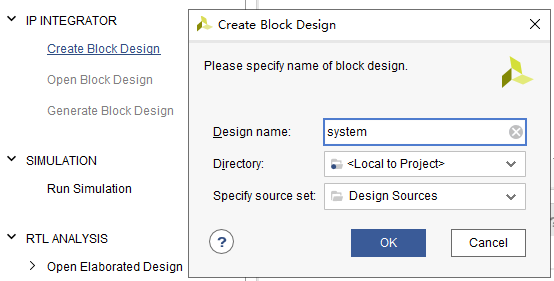




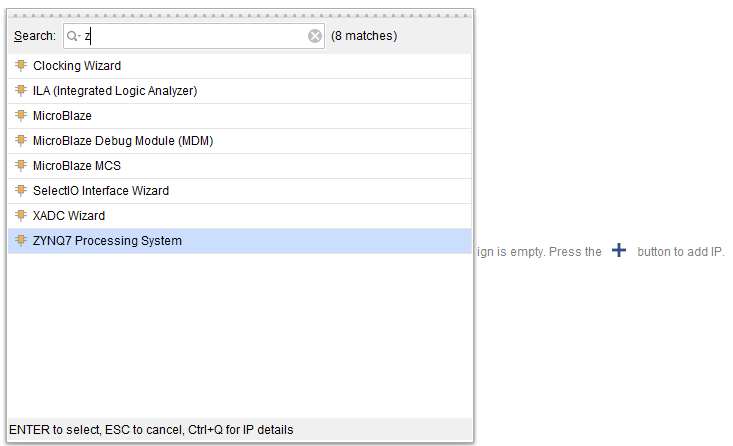


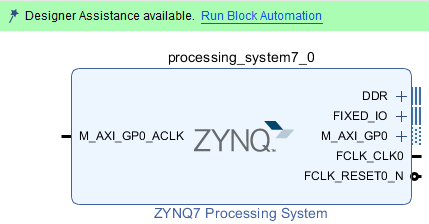
## 二、使用IP集成器创建处理器系统

使用Vivado中的IP集成器创建一个新的设计块，生成基于ARM Cortex-A9处理器的嵌入式硬件系统。

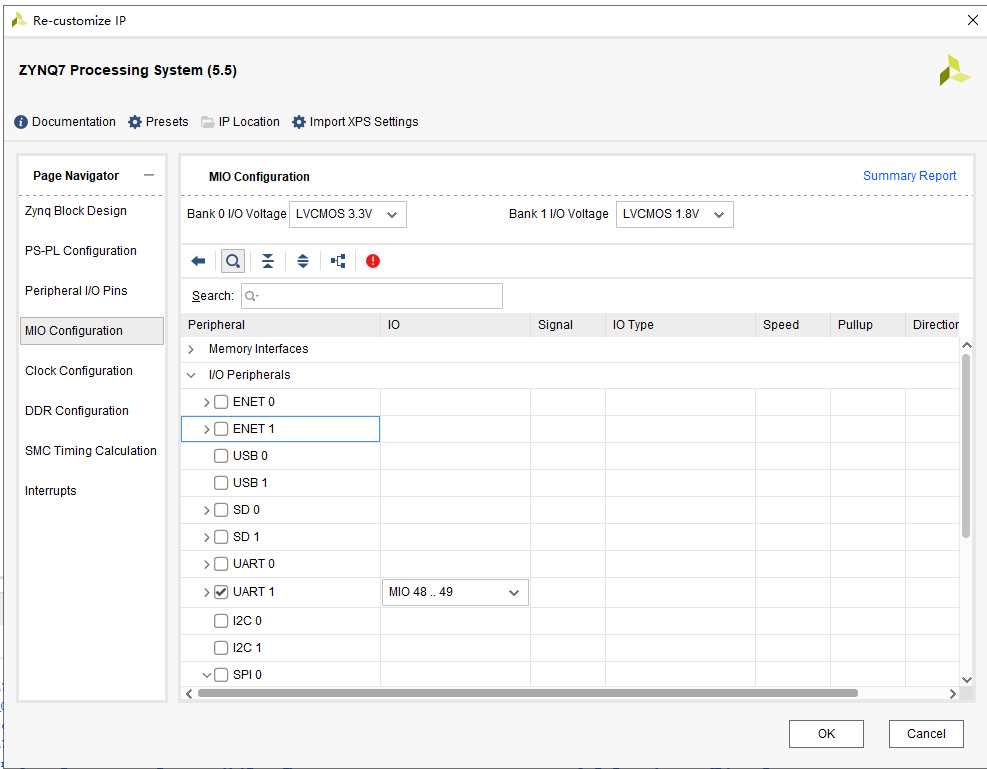


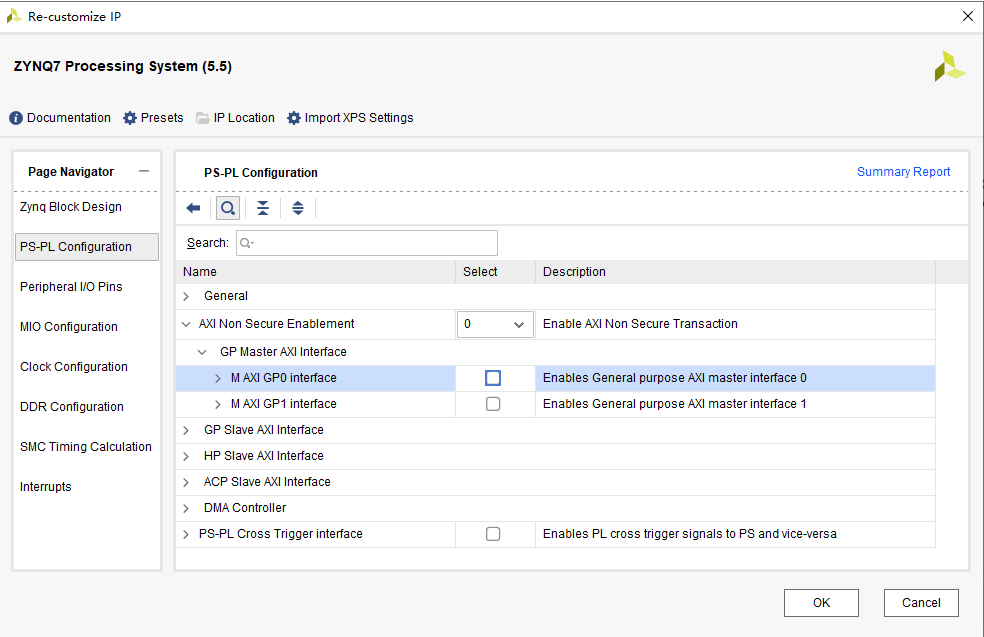
添加IP核：

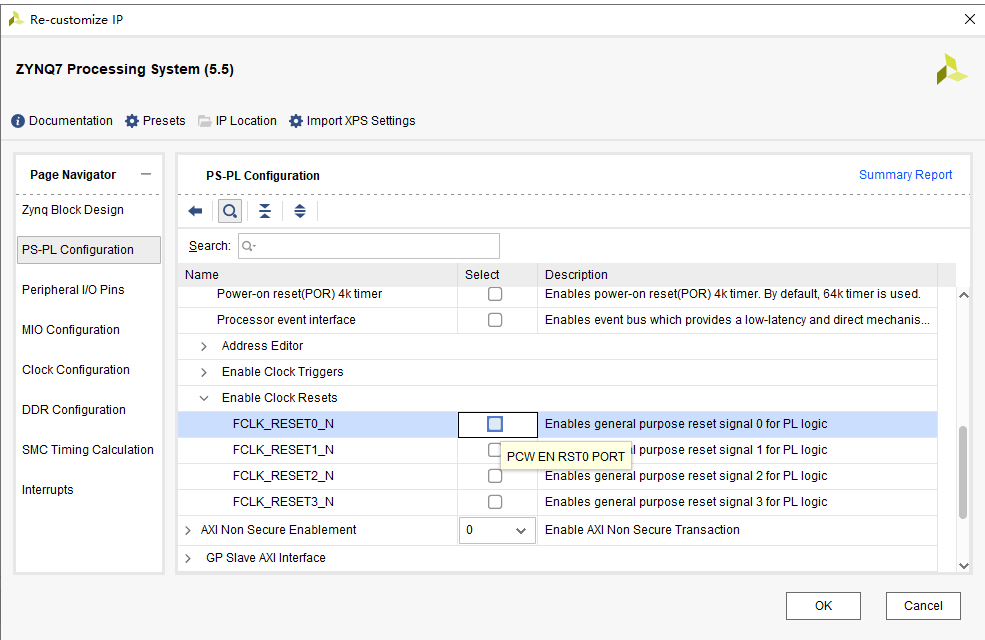


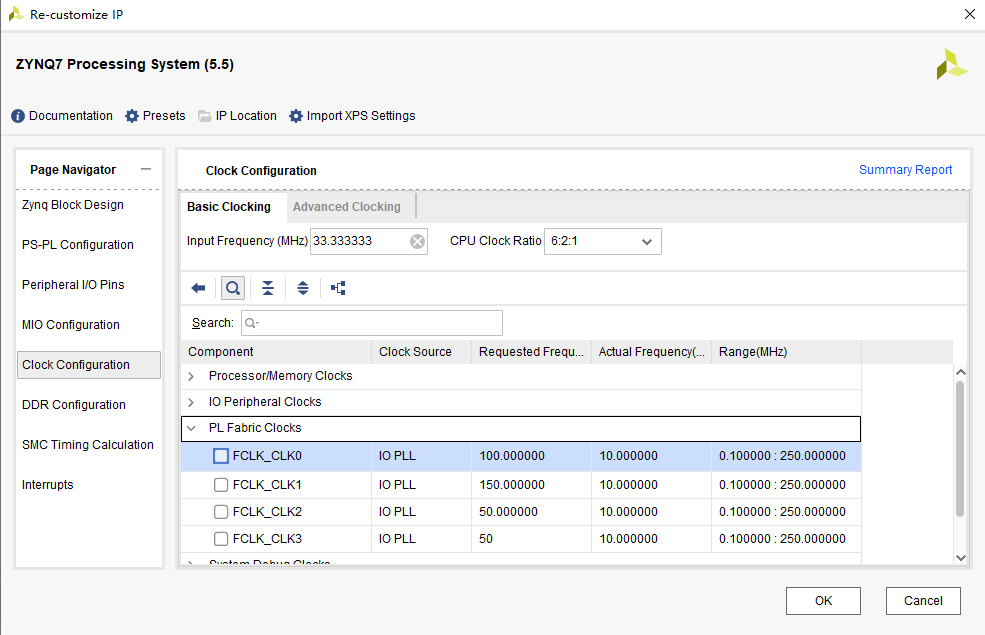


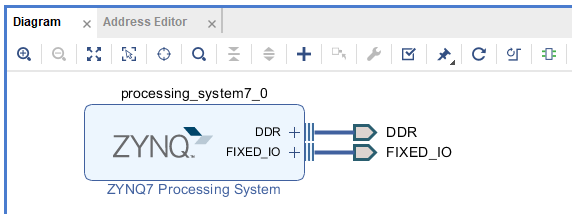
删减不必要的配件和端口：







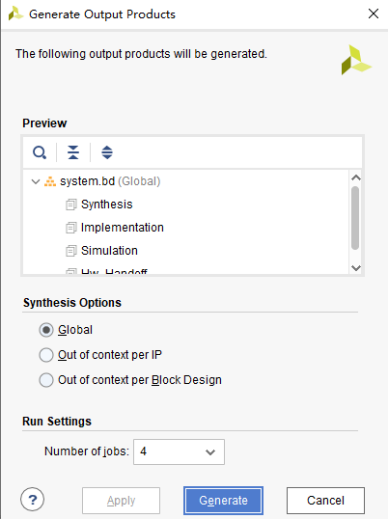
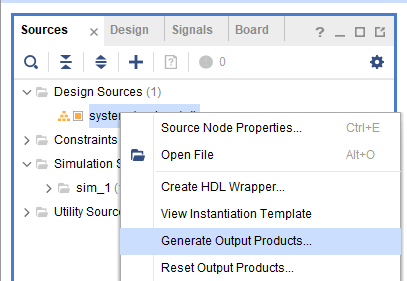


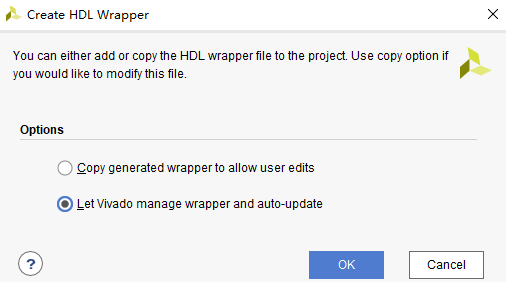
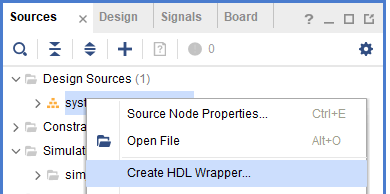


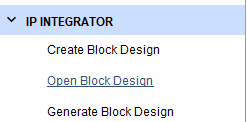
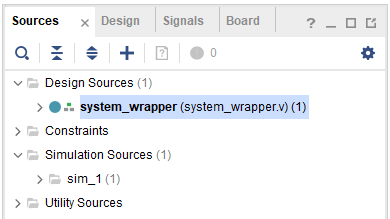
按“F6”检查设计是否有误。

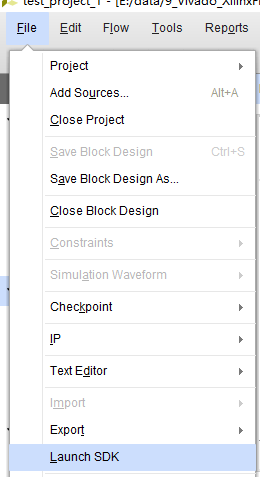
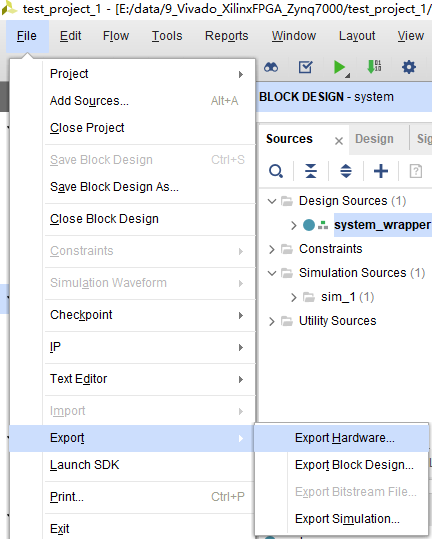
## 三、生成顶层HDL并导出设计到SDK

生成IP集成器输出，顶层HDL，启动SDK。





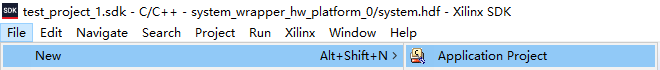


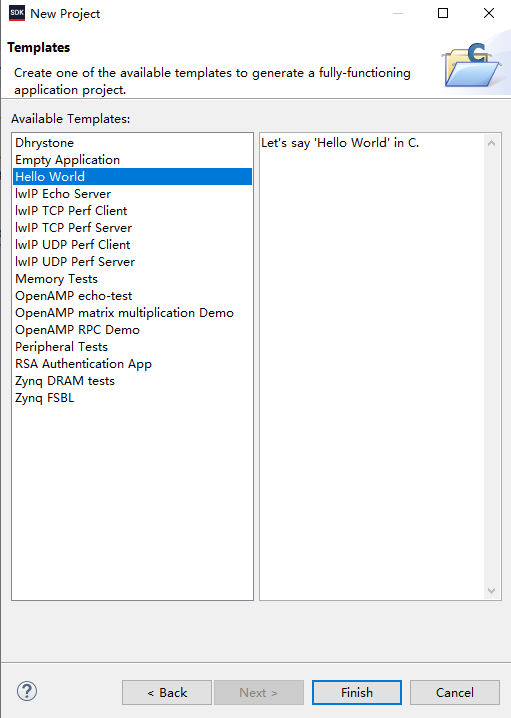
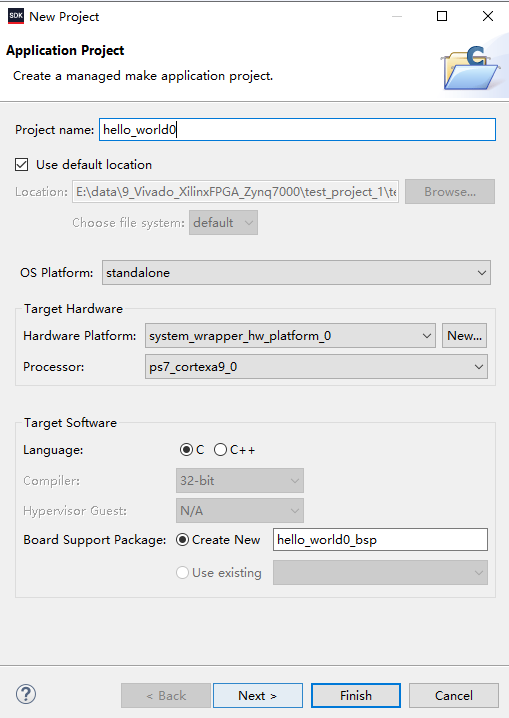


自动打开了SDK工具，左侧自动创建了system\_wrapper\_hw\_plantform\_0硬件目录。右侧自动打开system.hdf文件，硬件描述文件保存着工程硬件配置的基本信息。同时也给出了PS系统的地址映射和驱动信息。在后续软件设计中，SDK工具将使用hdf提供的硬件信息。

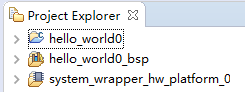
## 四、创建应用测试程序

使用标准工程模板，生成简单的应用程序工程。

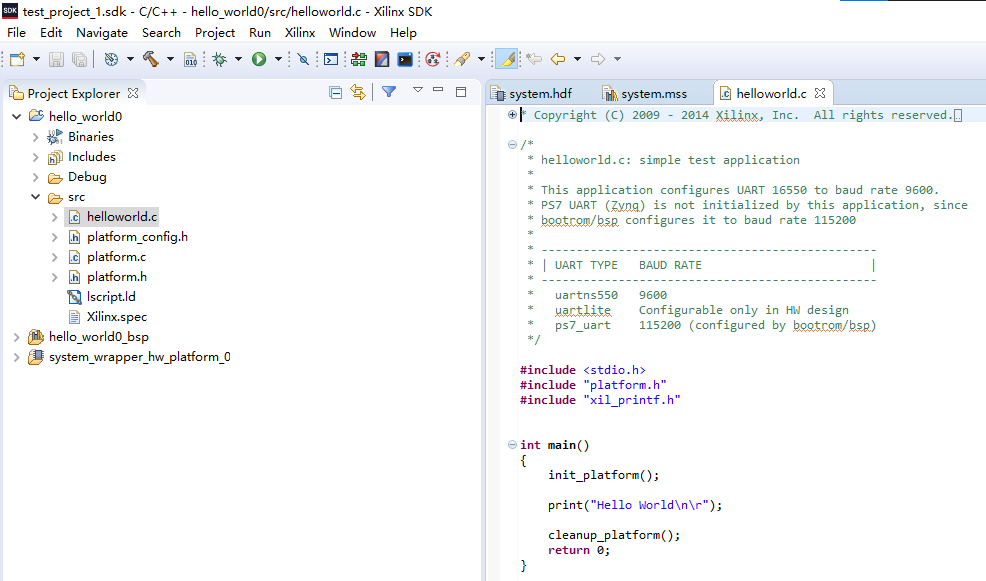




SDK工具将自动创建应用工程hello\_world0和板支持包工程hello\_world0\_bsp。下图三项依次是：用于验证设计的软件程序功能的应用工程，板级支持包的一部分，第一级启动引导的一部分用于初始化PS。



测试代码如下：



## 五、设计验证

**5.1 验证前的硬件准备**

正确设置启动模式：M0到M4跳线组用于设置Zynq-7000SoC启动时的模式（连接上面GND为0、连接下面VCC为1）。

**00000：JTAG模式 00010：四-SPI模式 00110：SD卡模式**

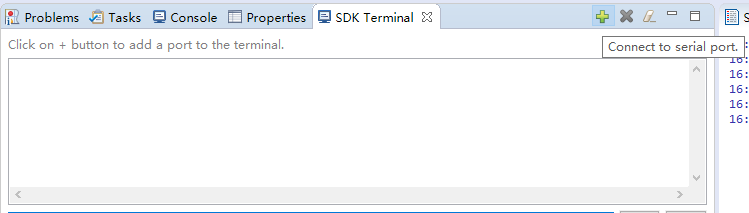
安装USB-UART驱动程序：芯片PL2303SA，驱动位置zynq\_example\PL2303\_Prolific\_ DriverInstaller\_v1.12.0.exe

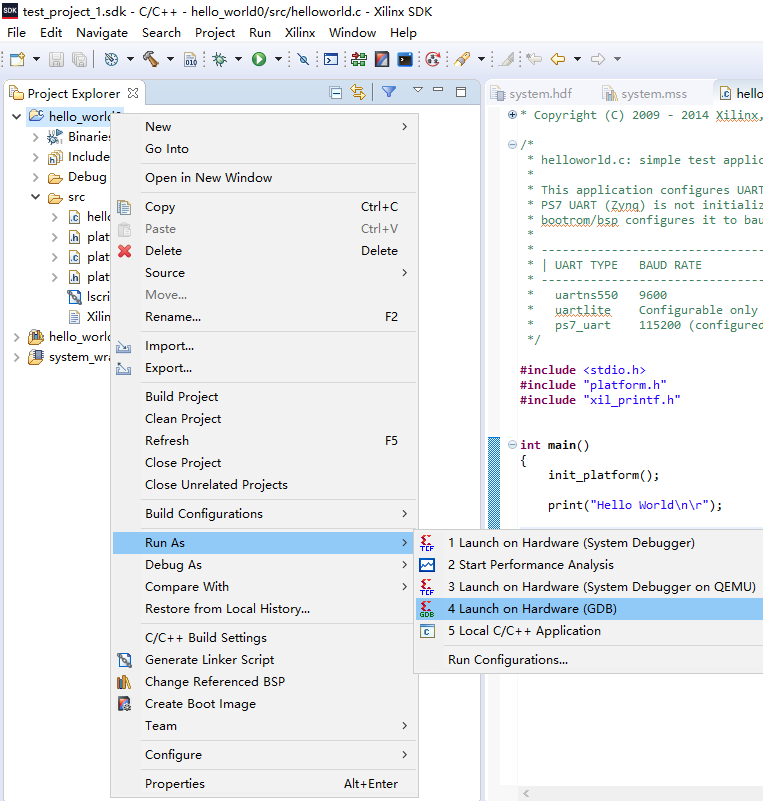
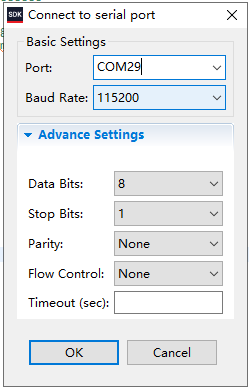
Z7-EDP-1开发平台供电方式：J6插座外接5V\*3A直流供电方式，电源开关SW10。

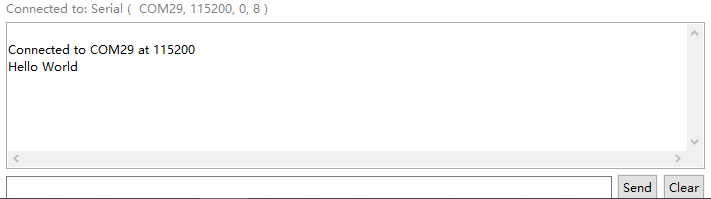
JTAG电缆连接方式：具备板载仿真器，USB接J12下载程序。

**5.2 设计验证的具体实现**

开发板设置为JTAG下载模式，连接串口线、JTAG下载线以及电线，打开电源开关。







File-->Exit 关闭SDK。 File-->ClosePoject关闭工程文件。

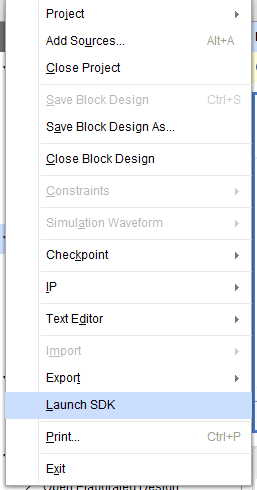
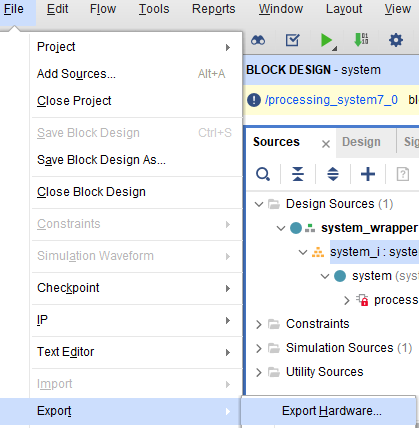
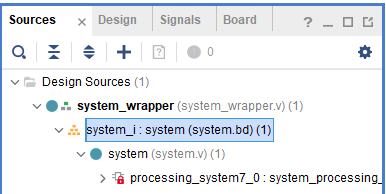
## 六、SDK调试工具的使用

通过在SDK工具中新建一个存储器测试工程，掌握SDK调试工具的特性和功能。

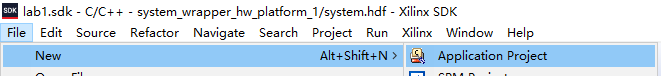
**6.1 打开前面设计的工程**

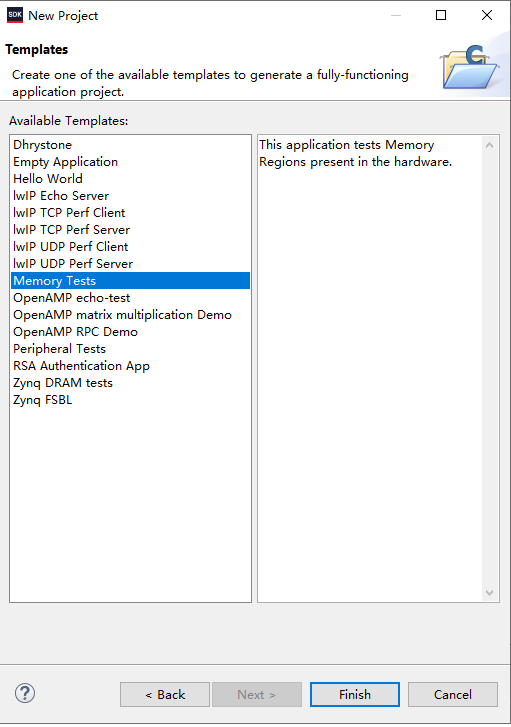
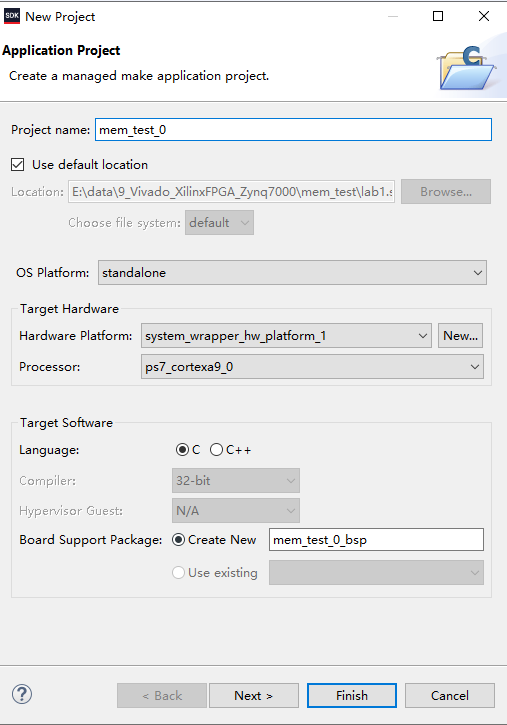
将样例工程zynq\_example\lab1复制到新建文件夹mem\_test。并打开该工程。

6.2 导入工程到SDK



**6.3 建立新的存储器测试工程**

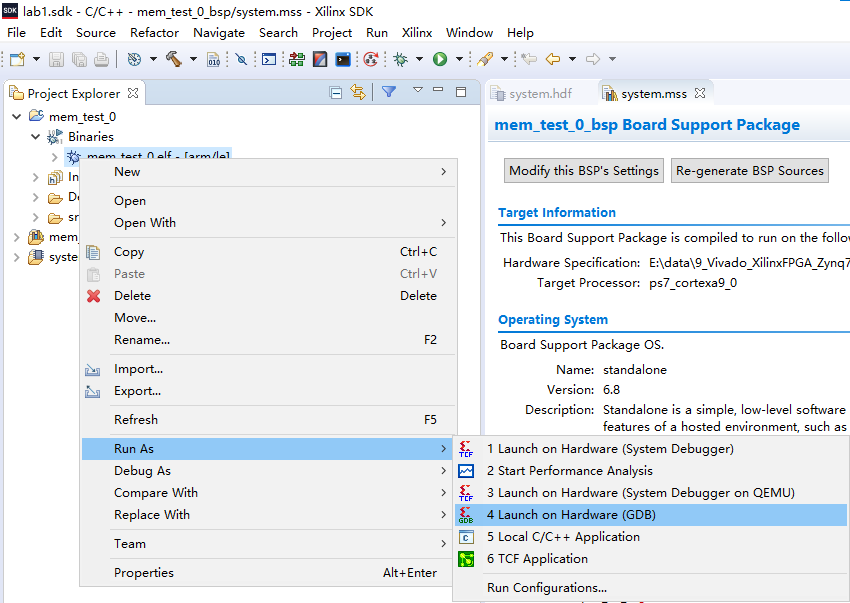




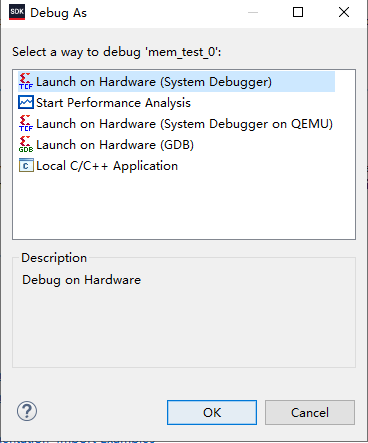
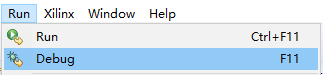
**6.4 运行存储器测试工程**

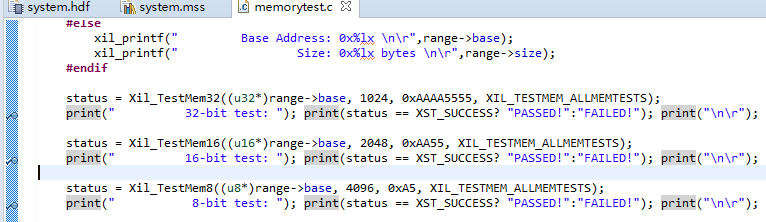
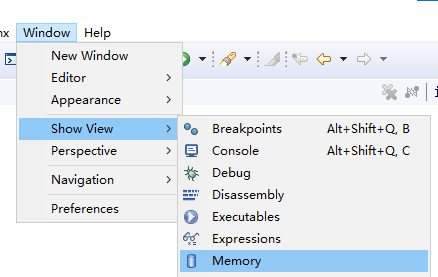
连接串口线与JETAG上电，打开串口。

如下图所示，运行程序。

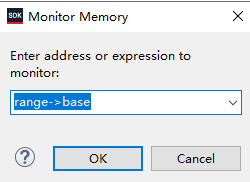
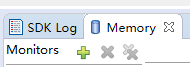


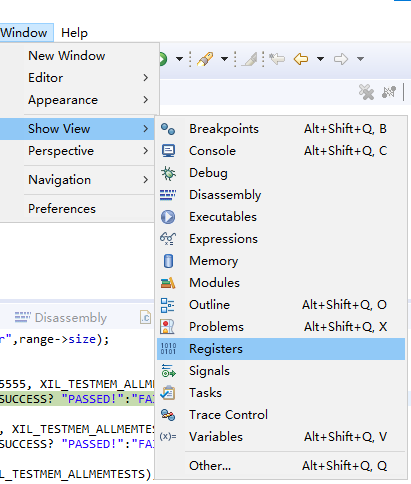
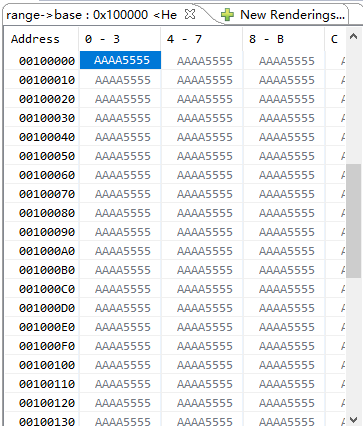
**6.5 调试存储器测试工程**

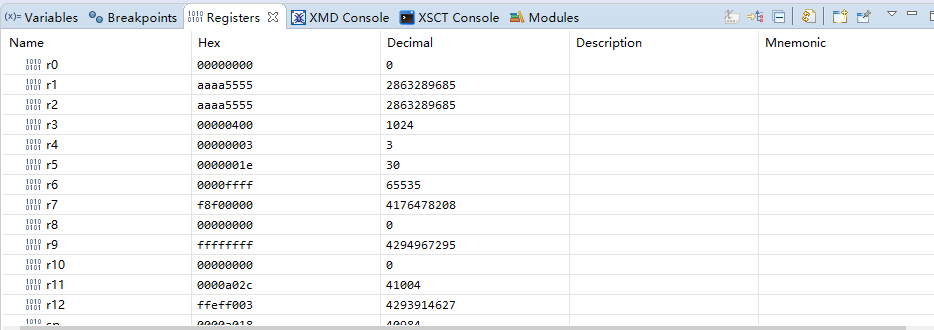




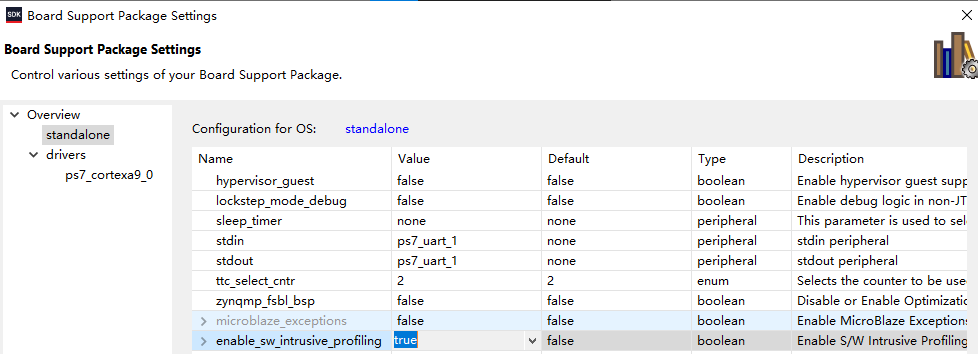
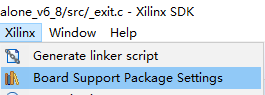
“F6”运行到第一个断点，添加要观测的存储器范围。

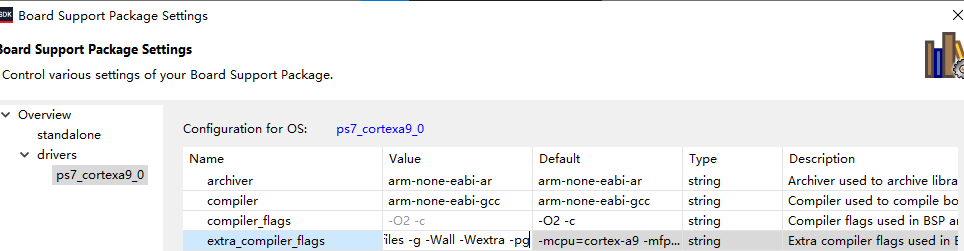


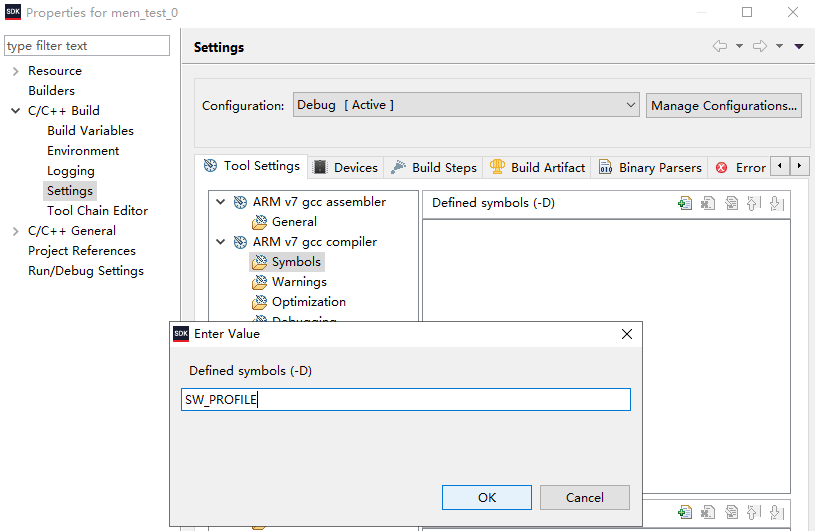
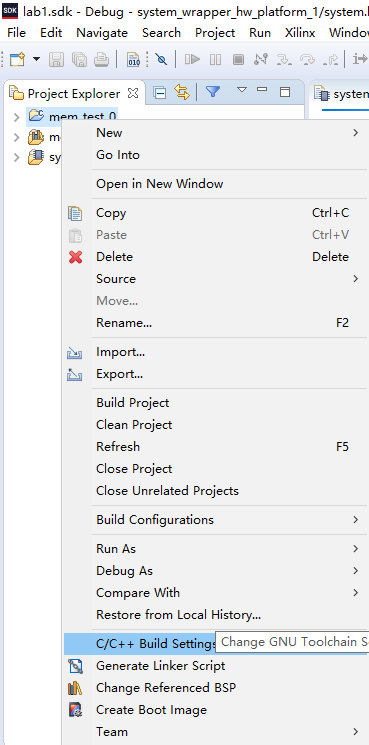


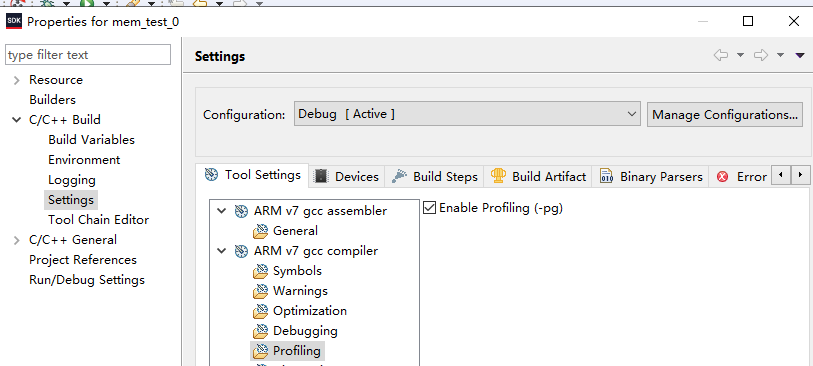


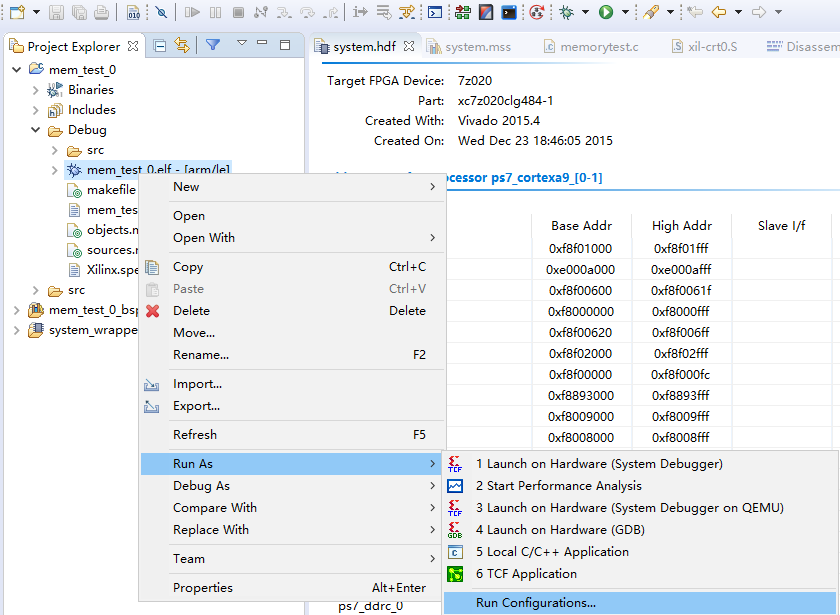
## 七、SDK性能分析工具



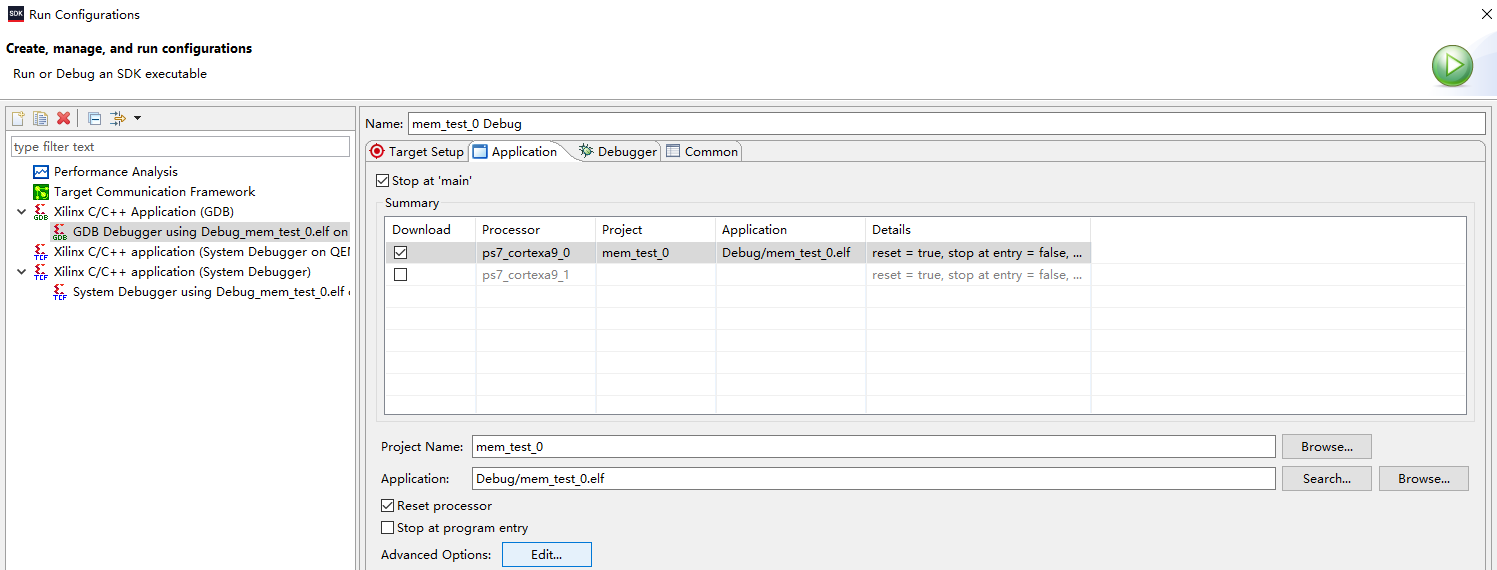


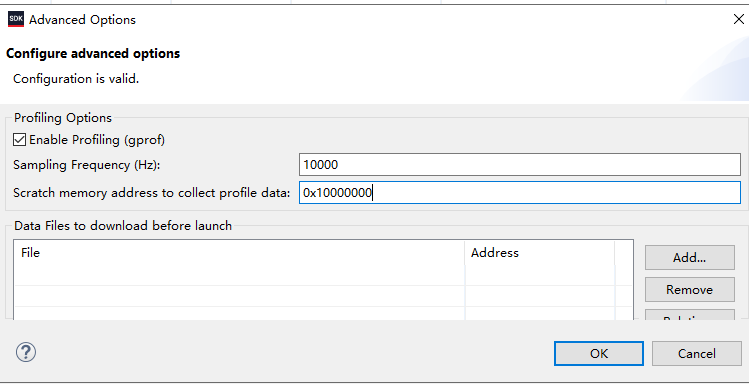






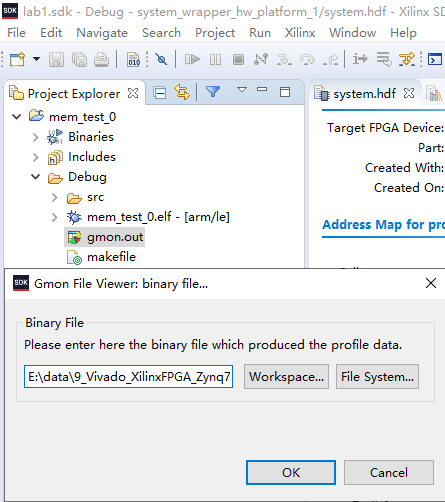
点击下图编辑：

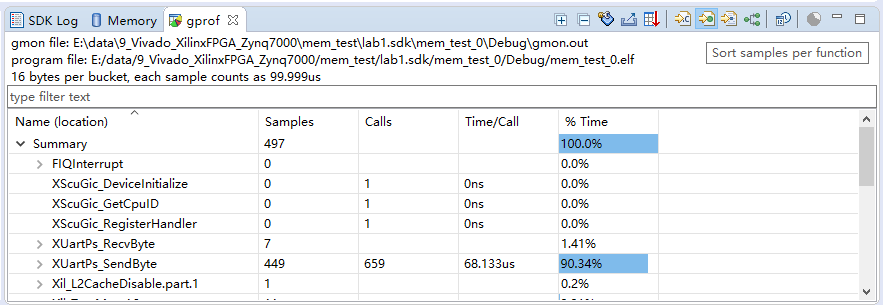




点击“RUN”运行程序。

双击gmon.out，点击OK。





# ARM GPIO的原理和控制实现

方法：①寄存器，②函数。

## 一、GPIO模块原理

MIO和EMIO，加载指令读一个GPIO组，保存指令写一个GPIO组。

Zynq7000SoC的GPIO控制与状态寄存器映射地址0xE000A000。多路复用MIO将BANK0和BANK1连接到芯片引脚。

0组MIO[31:0]，1组MIO[53:32](22个可用)，2组EMIO[31:0]，3组EMIO[63:32]。

控制器：DATA\_RO、DATA、MASK\_DATA\_LSW、MASK\_DATA\_MSW、DIRM、OEN。

EMIO信号：EMIO接口是PS和PL之间的简单连线，输入是来自PL的布线连接与OEN寄存器输出无关，输出线无三态使能所以OEN对输出无影响，输出使能这是简单地从PS输出受DIRM/OEN控制。

中断功能：……

GPIO编程流程：复位、时钟、GPIO引脚配置、将数据写到GPIO输出引脚、从GPIO输入引脚读取数据、将GPIO引脚设置为唤醒事件。

GPIO引脚配置：每个GPIO引脚都可以单独配置为输入或输出，但是BANK0的[8:7]必须配置为输出。例如将MIO的10号引脚作为输出，首先将方向设置为输出0x00000400写到gpio.DIRM\_0寄存器，然后输出使能将0x00000400写到gpio.OEN\_0寄存器。例如将MIO的10号引脚作为输入，将方向作为输入0x0写到gpio.DIRM\_0寄存器。

将数据写到GPIO输出引脚：①修改gpio.DATA\_0寄存器，②使用MASK\_DATA\_x\_ MSW/LSW寄存器。

读取GPIO输入引脚读数据：①读取gpio.DATA\_0寄存器，②设置中断。

GPIO作为唤醒事件：在CIC中使能GPIO中断，通过gpio.INT\_EN\_{0..3}寄存器使能对应中断。

I/O接口、寄存器说明略、其他API函数略。

底层读写函数：u32 Xil\_In32(INTPTRAddr)、void Xil\_Out32(INTPTRAddr,u32 Value)。

## 二、Vivado环境下MIO读/写控制的实现

两种控制方式：直接通过寄存器控制、调用SDK的API函数控制。

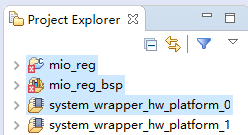
**2.1 调用底层读写函数编写GPIO应用程序**

“IP设计”-->”打开块设计”

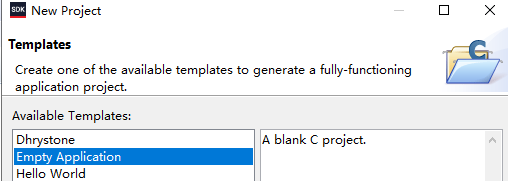
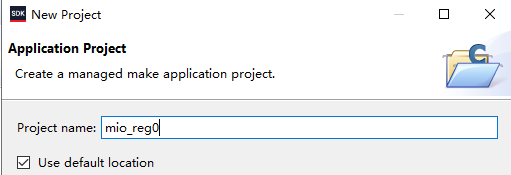
“文件“-->”导出“-->”导出硬件“

“文件“-->”打开SDK“ 自动打开SDK工具

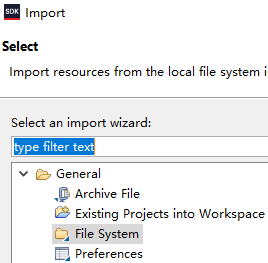
【附加】：清除下列三个已经存在的文件

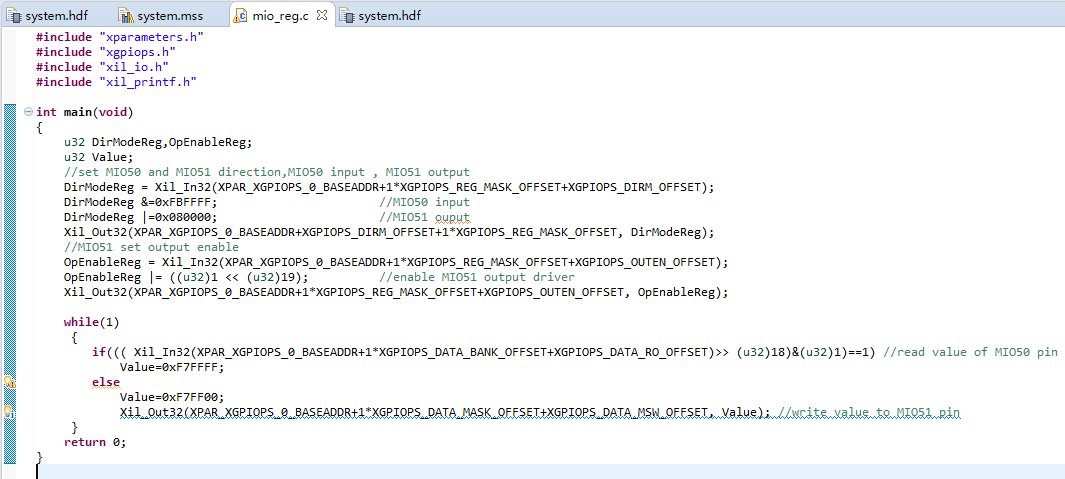
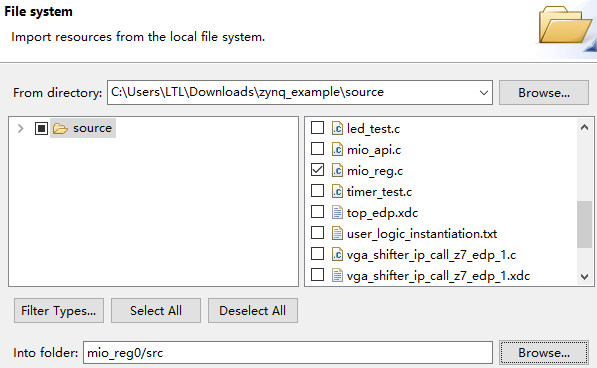


“文件“-->”新建“-->”应用工程“

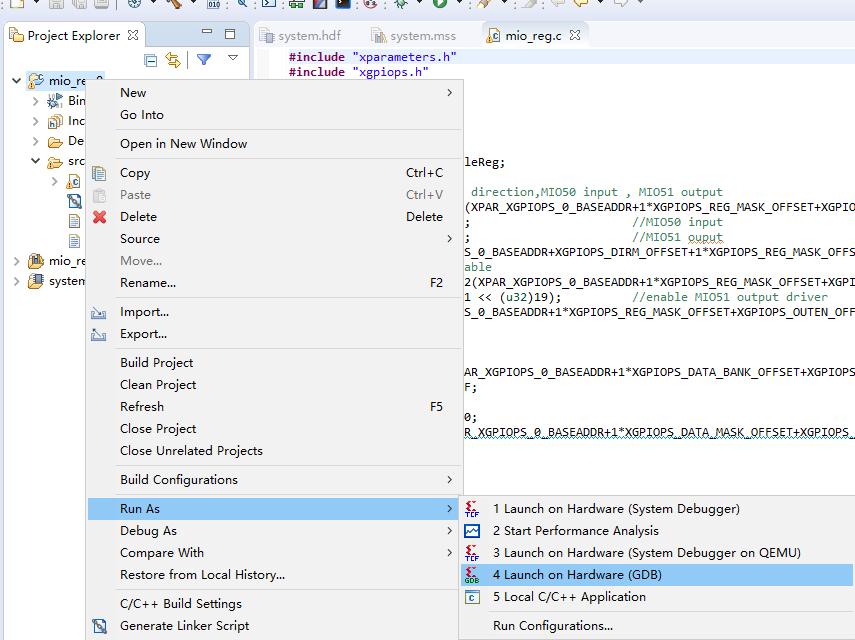


添加测试文件mio\_reg.c：





运行：按下图所示运行，按下U34按键，GPIO熄灭。



**2.2 调用API函数编写控制GPIO应用程序**

步骤差不多同上。

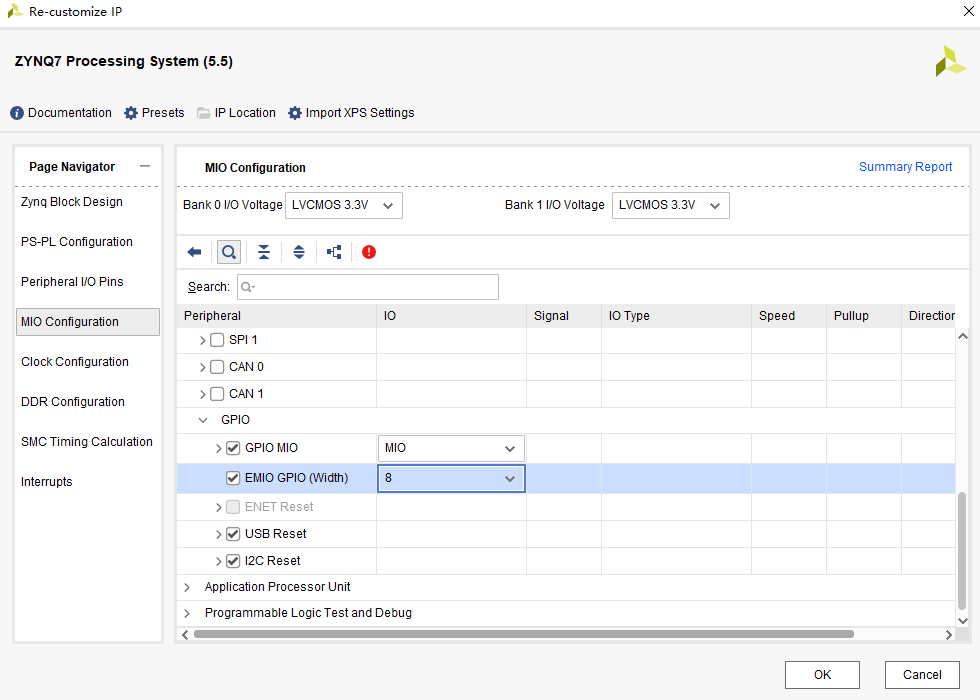


## 三、Vivado环境下EMIO读/写控制的实现

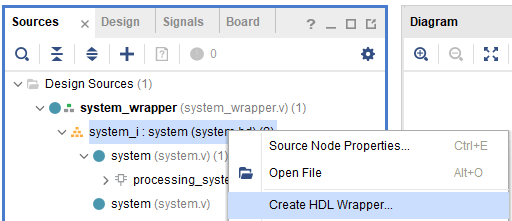
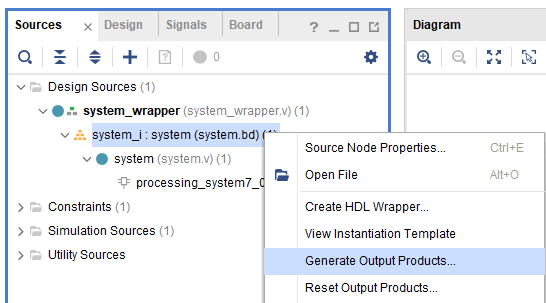
通过器件的PL部分，外部连接了8个LED灯，将第二组GPIO通过EMIO引入到PL连接到用于控制8个LED的8个引脚上。两种控制方式：寄存器、API。

**3.1 调用底层读/写函数编写GPIO应用程序**

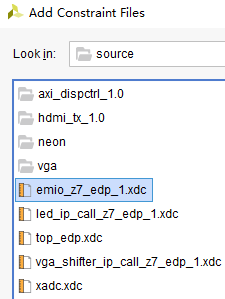
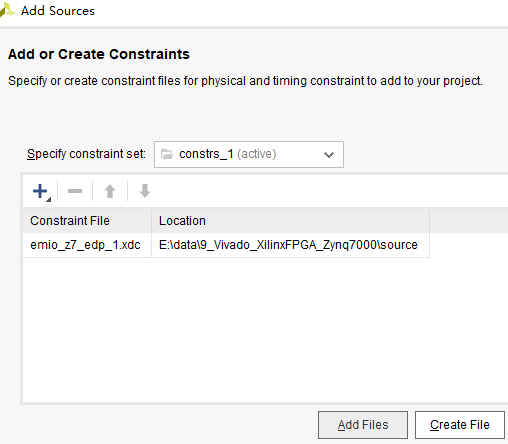
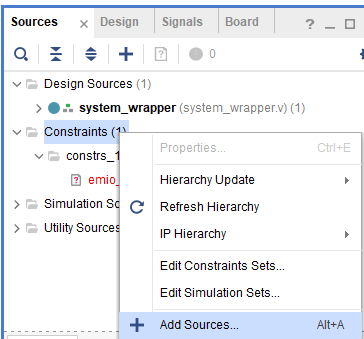
打开设计，双击ZYNQ模块，修改如下设置：

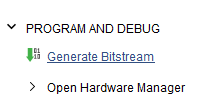


刷新，右键单击“GPIO\_0+“，选择”Make External“添加外部引脚。



添加约束文件：





**导出到SDK中：**

“文件“-->”导出“-->”导出硬件“ **要勾选上选项**

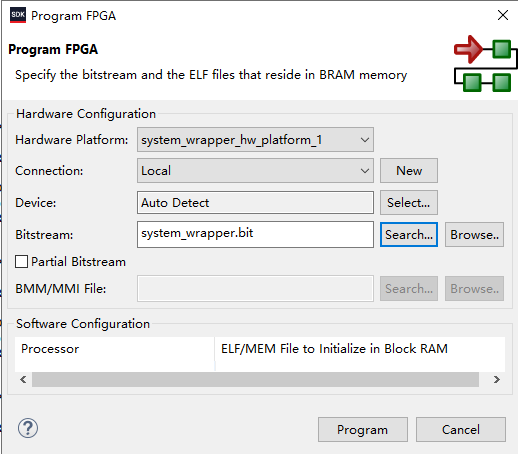
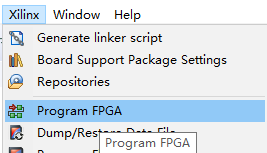
“文件“-->”打开SDK“ 自动打开SDK工具

【附加】：清除下列三个已经存在的文件

“文件“-->”新建“-->”应用工程“

……（同上的步骤添加测试文件）

**下载比特流文件到FPGA：**



**运行：GDB**

**3.2 调用API函数编写控制GPIO应用程序**

同上，打开设计、添加约束文件、导出硬件

# 附加：总体流程

1、设计IP核。

2、综合

3、打开综合后设计，配置引脚（添加约束文件）

4、生成比特流文件

5、导出硬件设计

6、打开SDK开发工具

7、创建测试工程

8、添加测试程序

9、下载比特流文件

10、运行测试程序