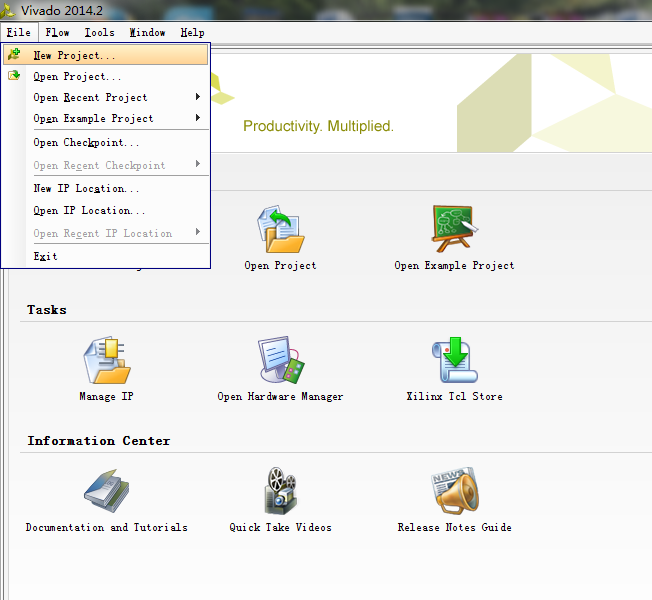
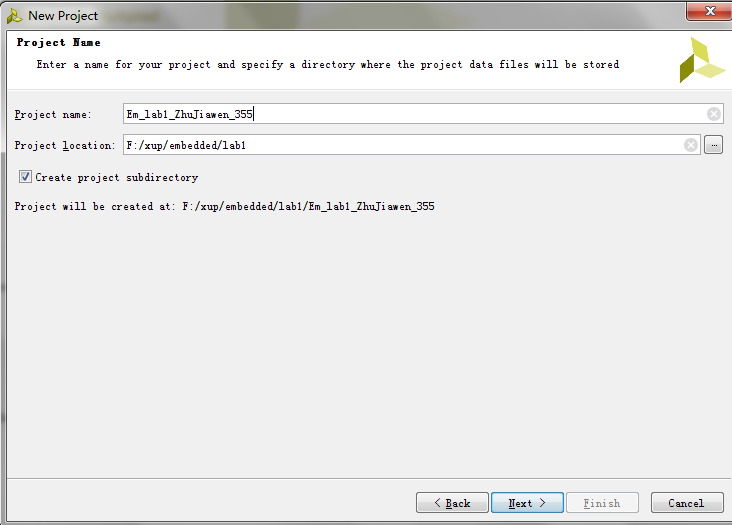
**Lab1 实验手册**

启动软件vivado 2014.2

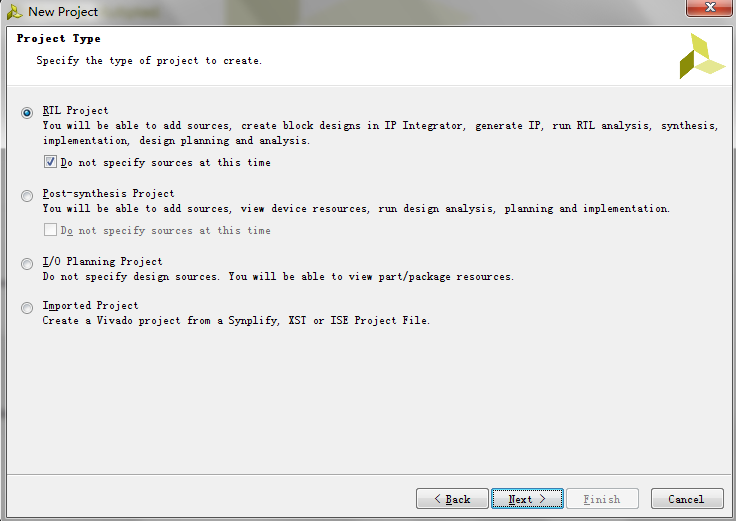
选择New Project 新建项目



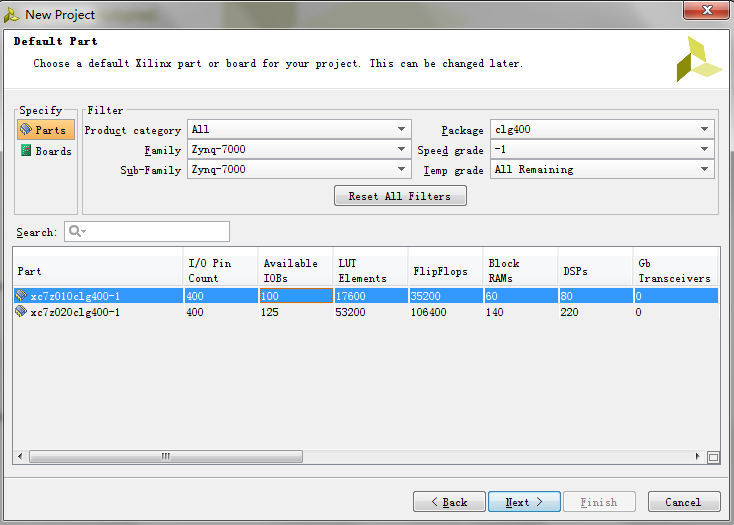
设置Project Name和Project location后，点击Next



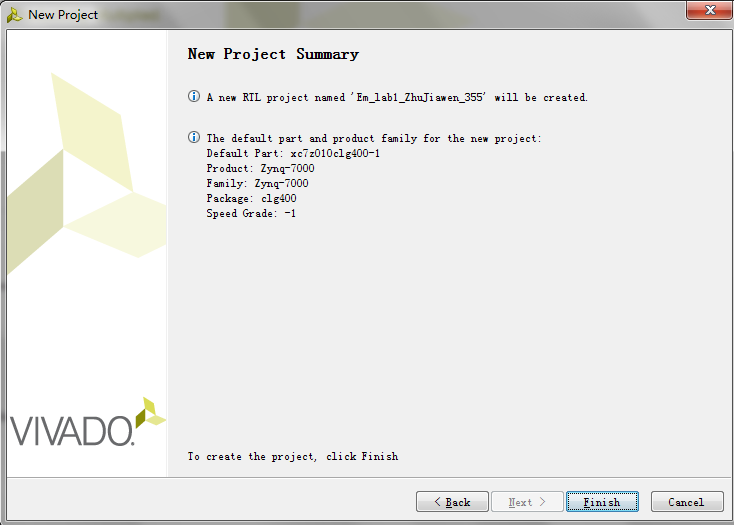
选择项目的类型，点击Next



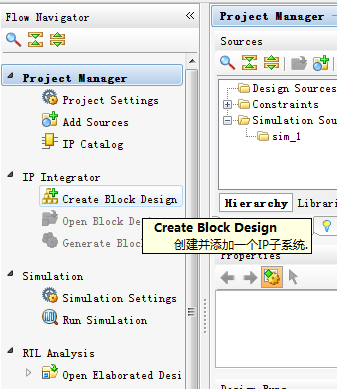
选择硬件，点击Next



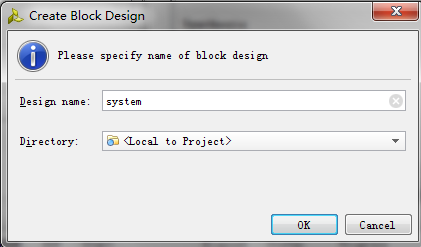
点击Finish完成项目的新建



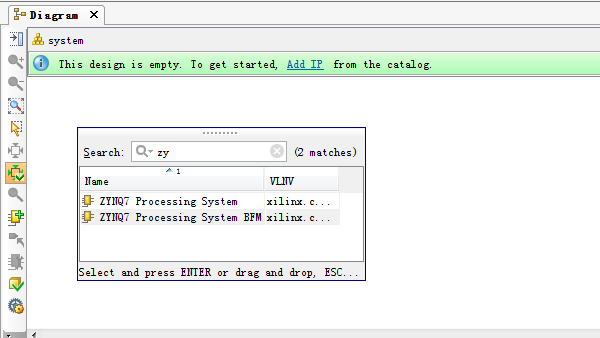
在Flow Navigator面板内选择Create Block Design 创建一个IP子系统



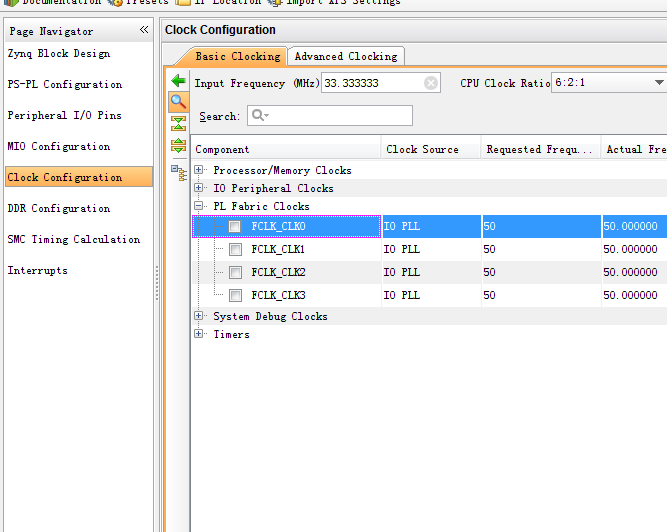
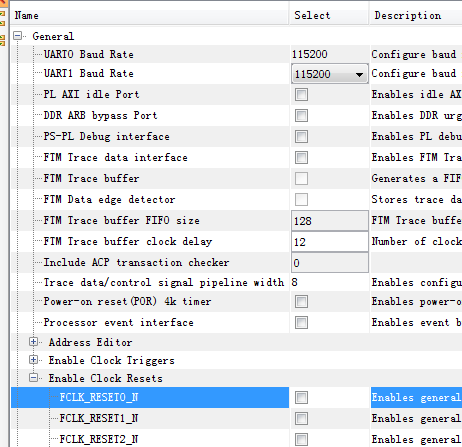
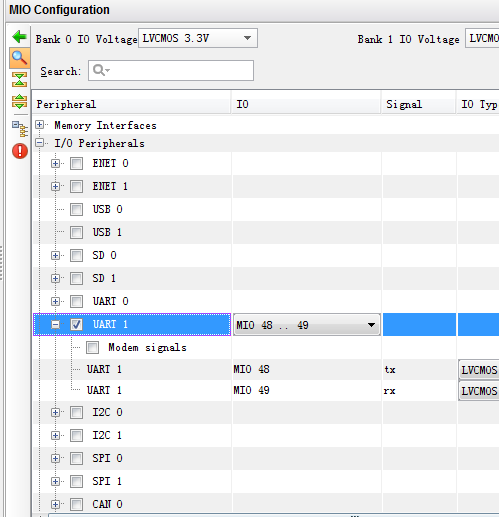
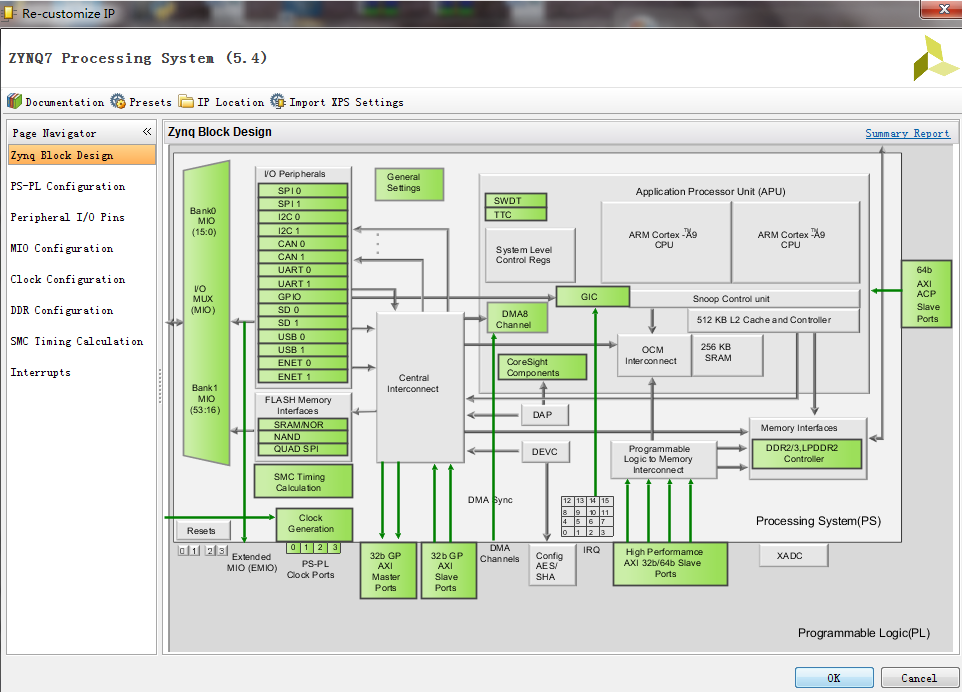
设置系统的名称为system



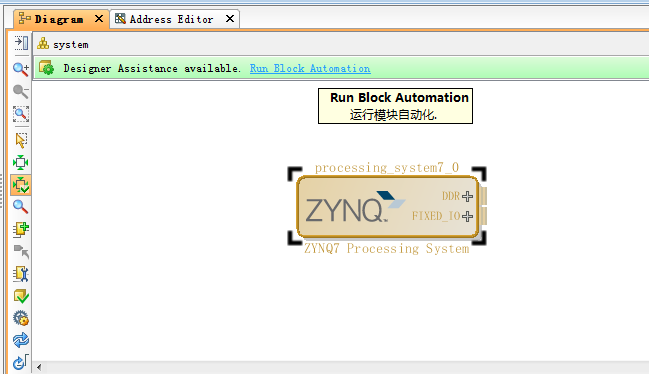
在Diagram面板内点击Add IP,并选择开发板型号



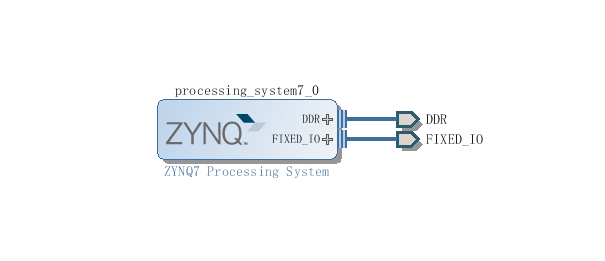
双击生成的IP进行配置，勾选UART1并去除其他所有的勾选



设置完点击Run Block Automation运行模块自动化



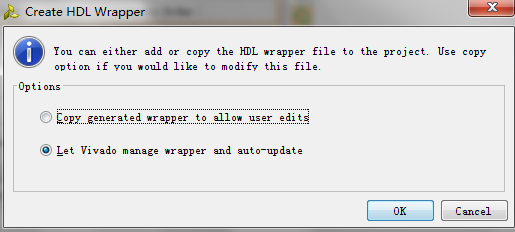
得到如下结果



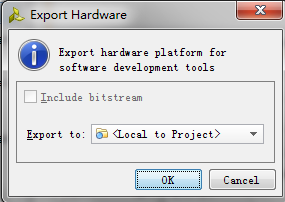
点击Sources选项，找到system

右键选择generate output products

再次右键选择create HDL Wrapper

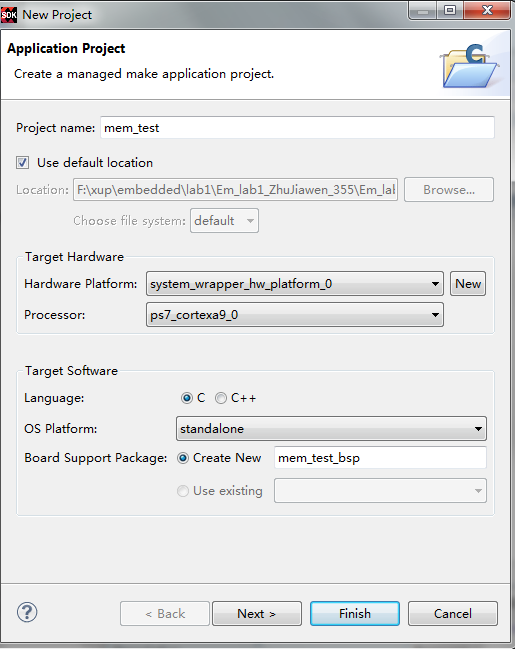


在File->export中选择export hardware

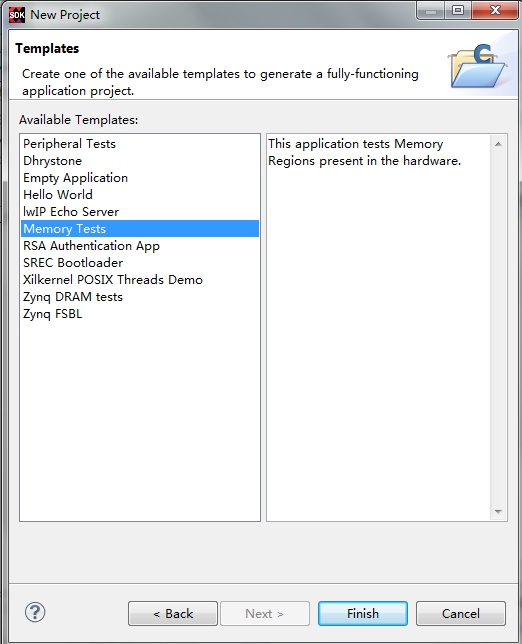


选择File->Lauch SDK

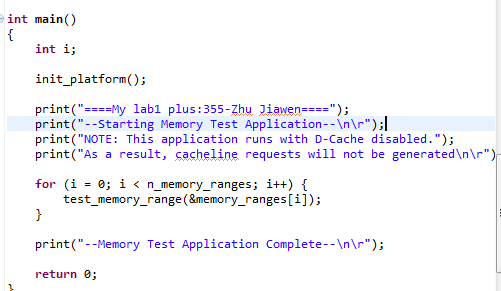
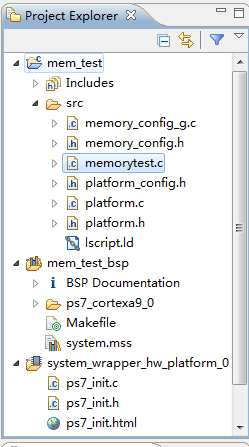
打开SDK后，New Project



点击Next,选择Memory Tests

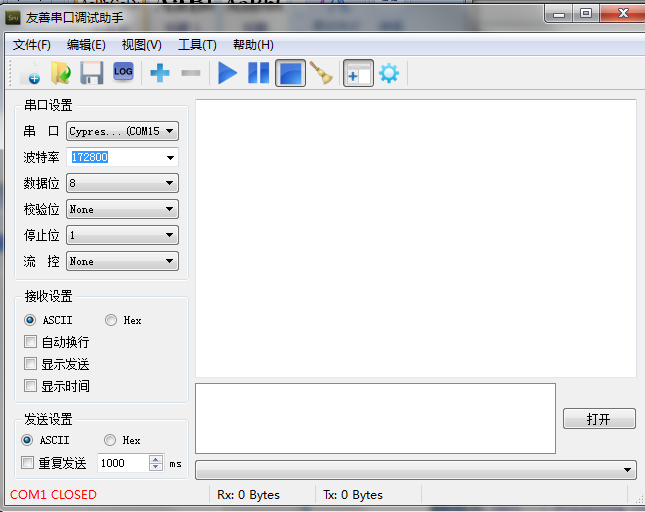


找到memorytest.c文件修改代码



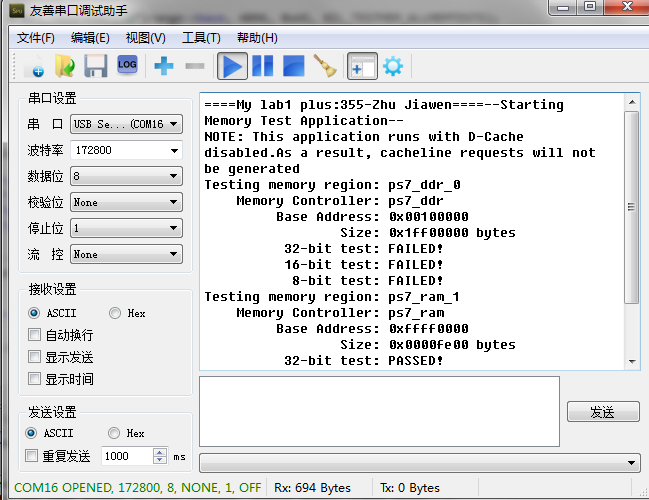
增加一行代码后，crtl+s保存，系统自动编译

打开串口调试软件,设置波特率为172800



Run工程

观察串口软件的输出



**Lab2 PL中添加Ip cores**

1.介绍：

扩展processing system:增加两个GPIO IPS

2.目标：

(1)配置PS的GP Master port，用来连接IP到PL中

(2)hardware design中添加IP

(3)建立一些compiler settings

3.过程

6个主要步骤：

(1)vivado中打开项目

(2)使用IP Integrator add且配置GPIO外设

(3)连接外部端口

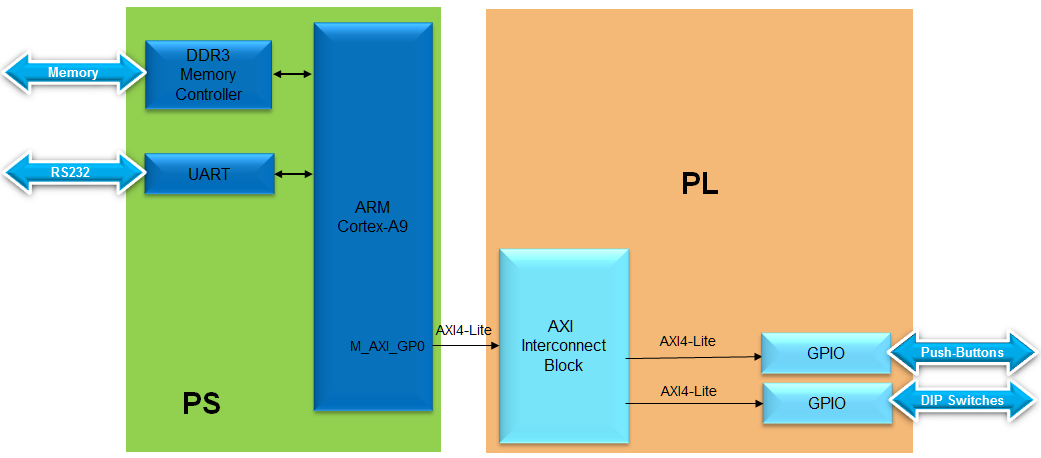
(4)生成bitstream

(5)export到SDK

(6)SDK中创建TestApp应用程序

(7)使用硬件验证设计

4.设计描述：



实验步骤：

1.打开Lab1,另存为Lab2

2.添加两个GPIO实例

2-1

Enable AXI\_M\_GP0接口,FCLK\_RESET0\_N,FCLK\_CLK0端口

从IP catalog增加两个GPIO外设到processor system中

2-1-1

Sources panel中，展开system\_wrapper，双击system.bd打开IP Integrator

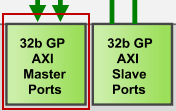
2-1-2.

双击Zynq block 打开Zynq配置窗口

2-1-3.

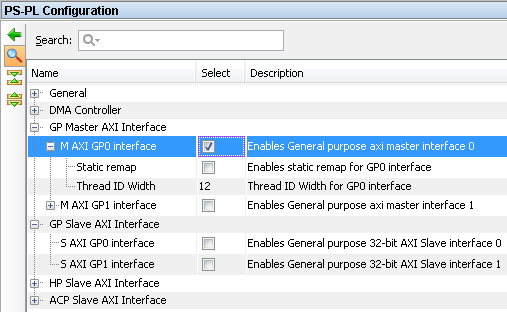
左侧Page Navigator中选择PS-PL Configuration

or Zynq Block Design视图中点击32b GP AXI Master Ports



2-1-4.

展开GP Master AXI Interface,勾选Enable M\_AXI\_GP0 interface来enable AXI GP0口



2-1-5.

展开General->Enable Clock Resets 勾选FCLK\_RESET0\_N选项（是系统时钟吗）

2-1-6.

选择Clock Configuration.展开PL Fabric Clocks,勾选FCLK\_CLK0（是PL时钟吗）

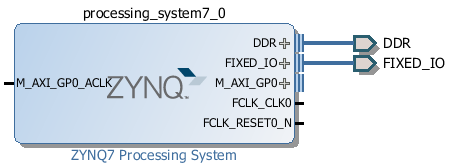
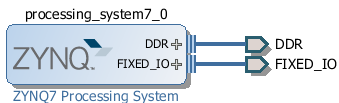
2-1-7.

注意现在processing system中的zynq模块有了M\_AXI\_GP0 interface,M\_AXI\_ACLK,FCLK\_CLK0及FCLK\_RESET0\_N port

点击重画diagram

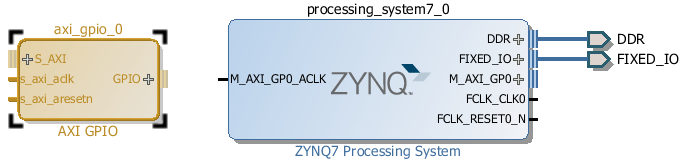


变为如下的图



2-1-8.

Add IP ,找到AXI GPIO后双击



2-1-10.

点击AXI GPIO 在Block properties面板中修改名字为sw\_4bit

2-1-11.

双击AXI GPIO，打开customization window 省略

2-1-12.

勾选All inputs（因为the switches are input only type devices）

GPIO Width设置为4（因为ZYBO板有4个switches）

注意：外设有两个channel可以配置，但是因为我们只希望用1个channel

2-1-14.

点击 Run Connection Automation 选择/sw\_4bit/S\_AXI

2-1-16.

注意到两个额外的模块:Processor System Reset及AXI Interconnect自动地添加到design中

2-1-17.

在添加一个GPIO外设得实例(Add IP),命名为btns\_4bit

2-1-18.

配置all inputs的width是4，这次手动配置

2-1-19.

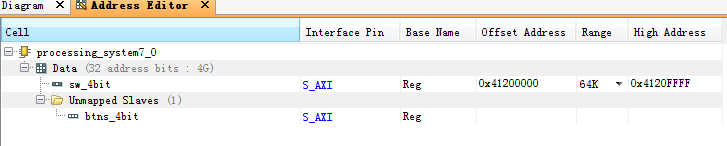
双击AXI Interconnect，number of Master Interfaces设置为2

2-1-20.

连接线

2-1-22.

点击Address Editor面板，展开processing\_system7\_0->Data->Unmapped Slaves

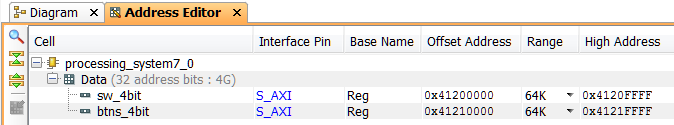


注意：sw\_4bit已经自动赋了地址，但是btns\_4bit没有（因为手动连接）

右键btns\_4bit，选择Assign Address或点击按钮。



两个外设的地址范围都是从0x40000000到0x7FFFFFFF(GP0的范围)



3.GPIO 外设连接外部端口

3-1.

push按钮和dip开关会被连接到板上相应的pins。增加location constraints.

3-1-2.

右键sw\_4bit实例的gpio端口，选择Make External来创建外部端口

3-1-3.

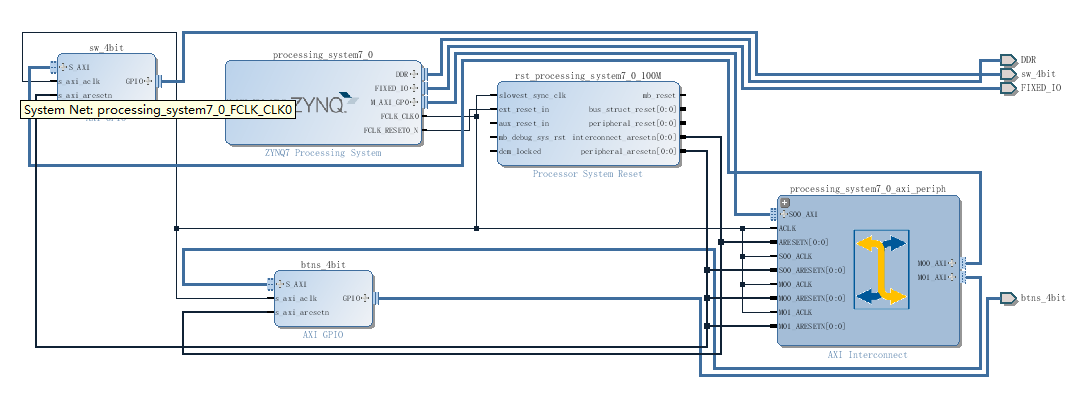
选择gpio端口，将名字改为sw\_4bit

3-1-4.

同样的，创建一个外部端口btns\_4bit,并命名为btns\_4bit

3-1-5.

运行Design Validation(Tools->Validate Design),验证没有错误



3-1-6.

sources视图中，右键system.bd，选择CreateHDL Wrapper来更新HDL Wrapper文件

3-2.

为sw\_4bit添加lab2.xdc文件，该文件是对sw\_4bit的限制

综合设计

打开I/O Planning布局

使用I/O Planning增加btns\_4bit限制

3-2-1.

Flow Navigator中点击Add Sources

3-2-2.

选择Add or Create Constraints选项，点击Next

3-2-3.

选择Add Files,路径切换到f:\xup\embedded\labs\lab2，选择lab2.xdc

3-2-4.

点击Finish

3-2-5.

在Sources视图中，注意到lab2.xdc在constraints的文件夹下了

3-2-6.

在Flow Navigator视图中，点击Run Synthesis。

综合完成后，选择Open Synthesized Design，点击Ok

3-2-7.

Layout dropdown menu中选择I/O Planning

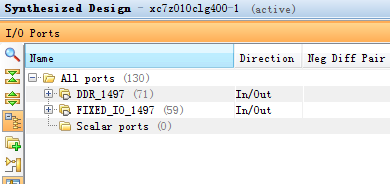
3-2-8.

I/O ports的tab中，展开btns\_4bit\_tri\_i,注意到pins还没有赋给这个外设。

sw\_4bit\_tri\_i已经赋了pin location.

注意到btns\_4bit\_tri\_i的I/O Std默认为LVCMOS18，而sw\_4bit\_tri\_i的值是LVCMOS33

3-2-9.



4.生成Bitstream,Export到SDK中

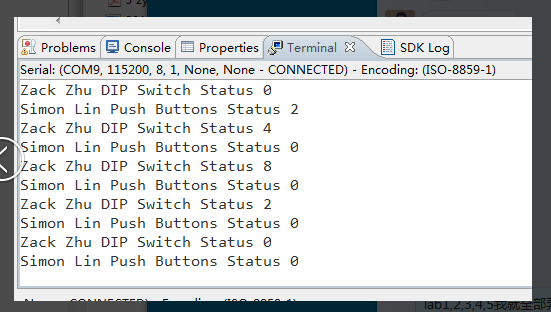
4-1.

生成bitstream,export到SDK中

5.SDK中生成TestApp Application

5-1.

生成软件平台项目



**Lab3 向system中添加Custom IP**

1.介绍：

使用Vivado IP Packager向processor system添加一个custom peripheral

创建一个AX14Lite interface 外设

2.目标：

(1)使用IP Packager创建一个custom peripheral

(2)修改IP功能

(3)添加custom peripheral到design中

(4)添加pin location constraints

(5)添加block memory到system中

3.过程：

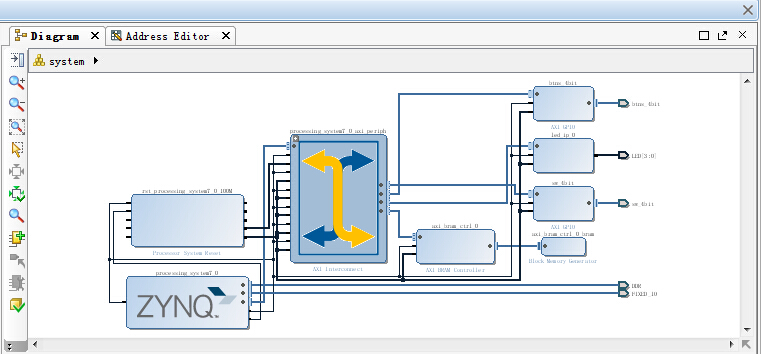
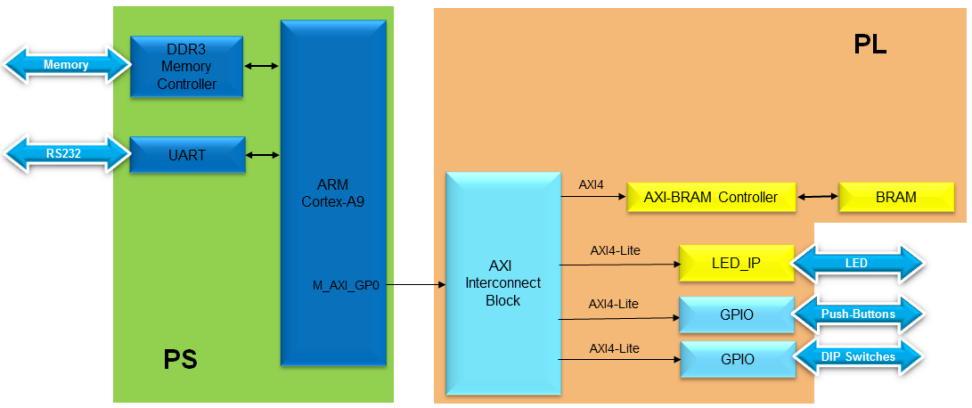
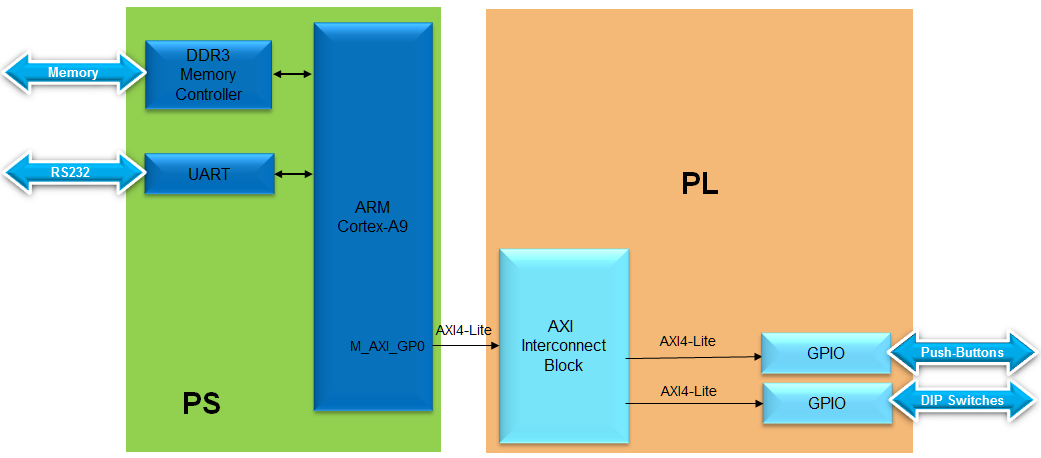
4个主要步骤：

(1)使用外设模板创建一个外设

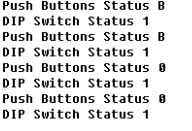
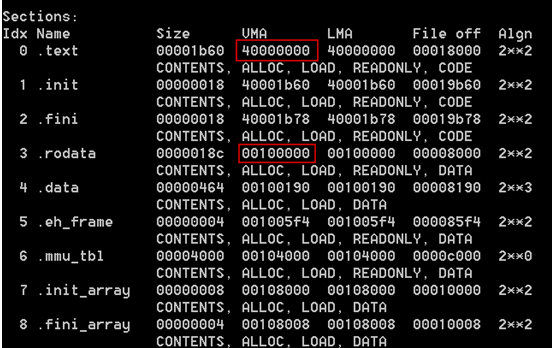
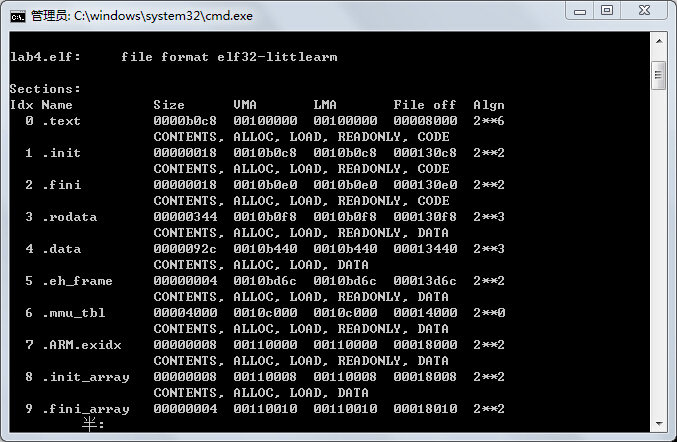
(2)使用IP Package package IP

(3)import,add,connect IP到design中

(4)add Block RAM Memory



**Lab4 Writing Basic Software Application**



**Lab5 Software Writing for Timer and Debugging**