实验四 基于HLS的程序设计及PYNQ-Z1的Overlay设计

[一、 Vivado HLS（高级综合）流程及示例 2](#_Toc72173932)

[1. HLS高级综合基础 2](#_Toc72173933)

[2. HLS基础优化指令 3](#_Toc72173934)

[二、 如何通过HLS定义PYNQ-Z1 Overlay（示例） 6](#_Toc72173935)

[三、 矩阵向量乘HLS优化思路 13](#_Toc72173936)

[四、 实验内容 15](#_Toc72173937)

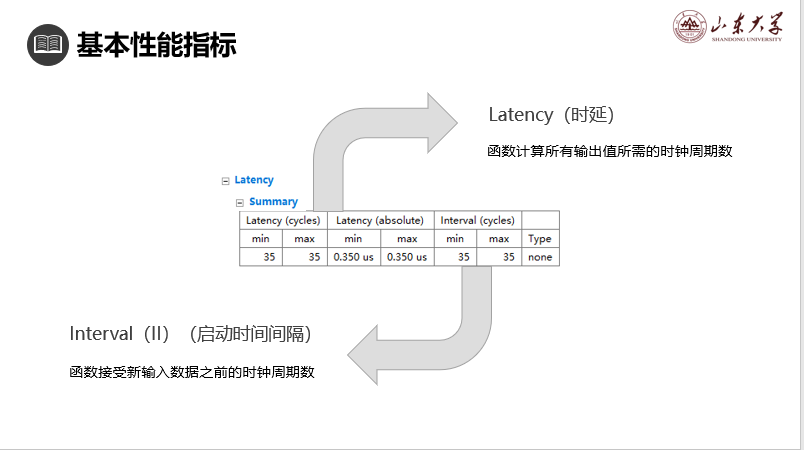
[五、 考察重点 15](#_Toc72173938)

[六、 实验要求 15](#_Toc72173939)

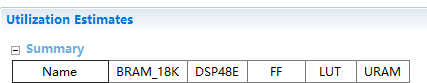
1. Vivado HLS（高级综合）流程及示例
2. HLS高级综合基础

仅介绍部分指标及基本流程，具体见附件中ug902（最好直接看原版）第一章

**性能指标：**Latency、Interval(II)，见图。



**资源指标：**



BRAM\_18K: 即block ram，18K为其存储空间大小，FPGA中的存储单元。

DSP48E: FPGA中预配好的高速运算单元。

FF: 触发器，FPGA的存储单元，控制FPGA的时序逻辑。

LUT: 查找表，FPGA的存储单元，控制FPGA的组合逻辑。

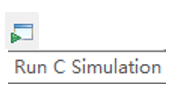
URAM: Ultrascale+系列FPGA中的存储单元，PYNQ-Z1中该列指标为0。

**存储单元性能比较**（查找表和触发器类似）**：**



**基本流程：**

C仿真 C综合 C/RTL联合仿真 导出RTL（IP核）

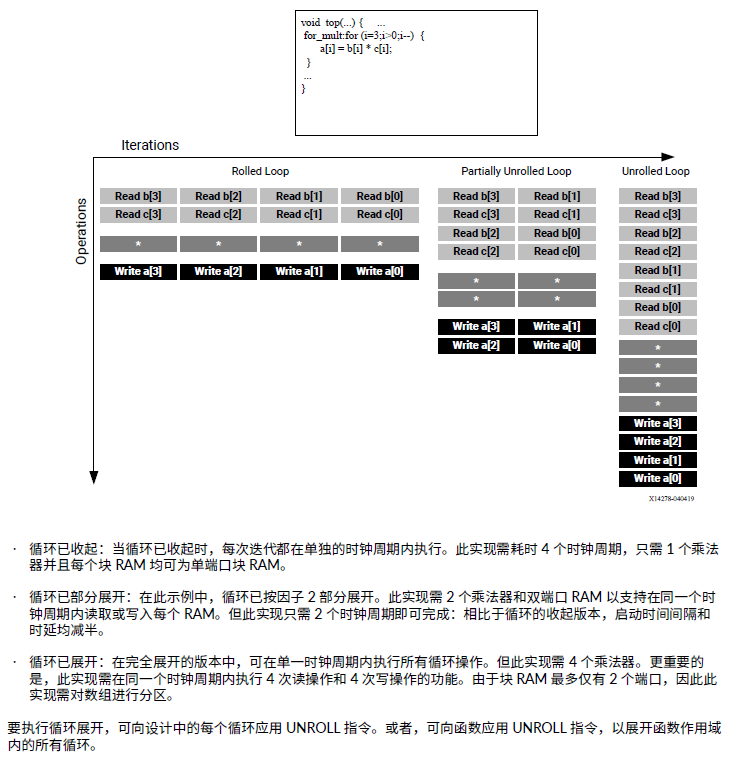




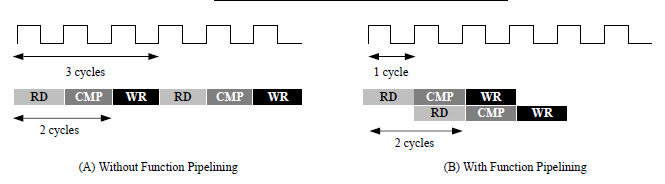
1. HLS基础优化指令

**① UNROLL（循环展开）：**

unroll是将for循环内的代码展开成8份，也相当于使用8倍的资源去实现这个结构。factor指定的unroll份数必须是可以被循环次数整除的数。factor也可以省略，default时unroll份数就是循环次数。

****

**② PIPELINE（流水线）：**

****

仅应用于指定区域，而不应用于下层层级，但下层层级内的所有循环都会自动展开。指定函数的下层层级内的所有子函数都必须单独PIPELINE。如果将子函数PIPELINE，其上层的已PIPELINE的函数即可充分发挥PIPELINE的性能；反之如果已PIPELINE的顶层函数下层的任意子函数不采用PIPELINE，则可能限制PIPELINE的性能。

对于函数，PIPELINE将永久运行，永不停止；对于循环，PIPELINE将持续执行直至循环的所有迭代完成为止。

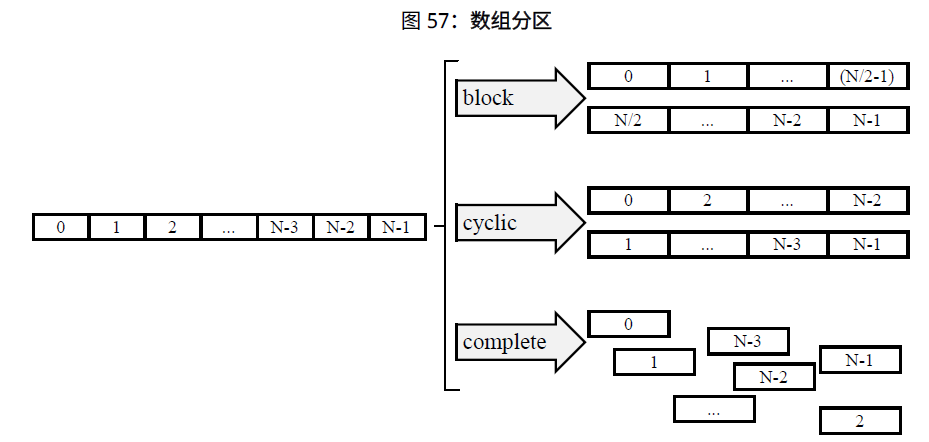
**③ PARTITION（数组分割）：**

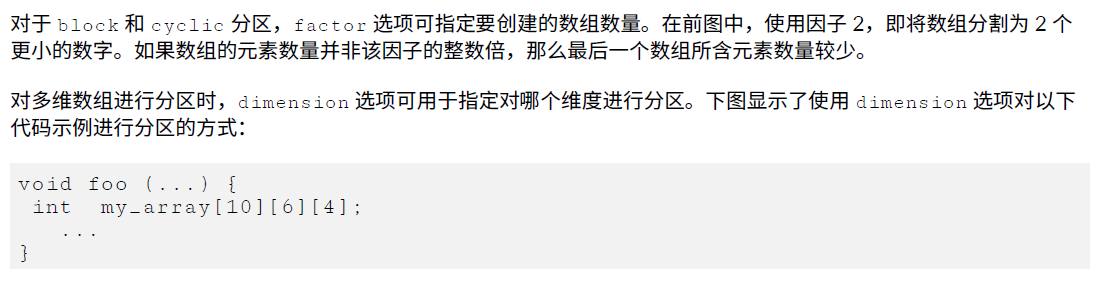
三种分区方式：

block：将原始数组分割为原始数组的连续元素块（大小相同）

cyclic：将原始数组分割成多个大小相同的块，这些块交织成原始数组的元素

complete：将数组按其独立元素进行拆分。对应于将内存解析为寄存器

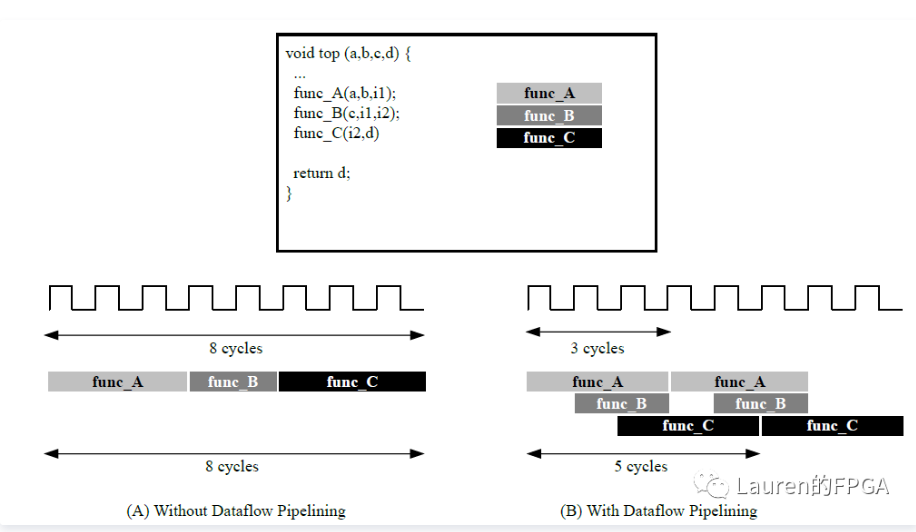






**④ DATAFLOW（数据流）：**

可以从函数和循环中构造出单独的流水线阶段（通常称为进程）。

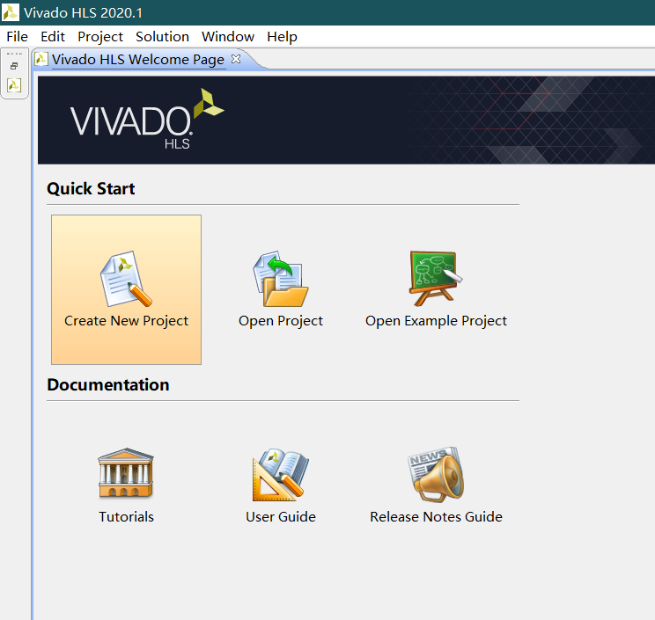
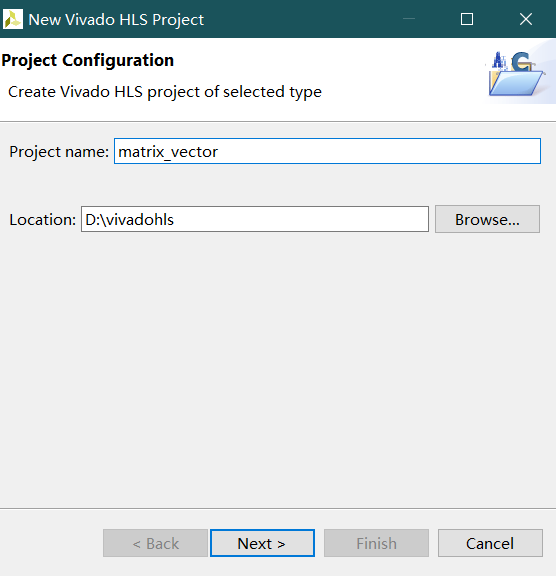
****

dataflow指令和pipeline指令都生成能够流水线执行的电路。关键的区别在于任务流水的粒度不一样。pipeline 指令构造了一个在循环级别上有效的流水线化的体系结构，由指令中的II所决定。运算符是静态调度的，如果II大于1，则可以在同一运算符上共享运算。dataflow 指令构造了一种体系结构，该体系结构可以有效地进行流水线操作，以获取（可能未知的）时钟周期数的操作，例如在数据块上运行循环的行为。

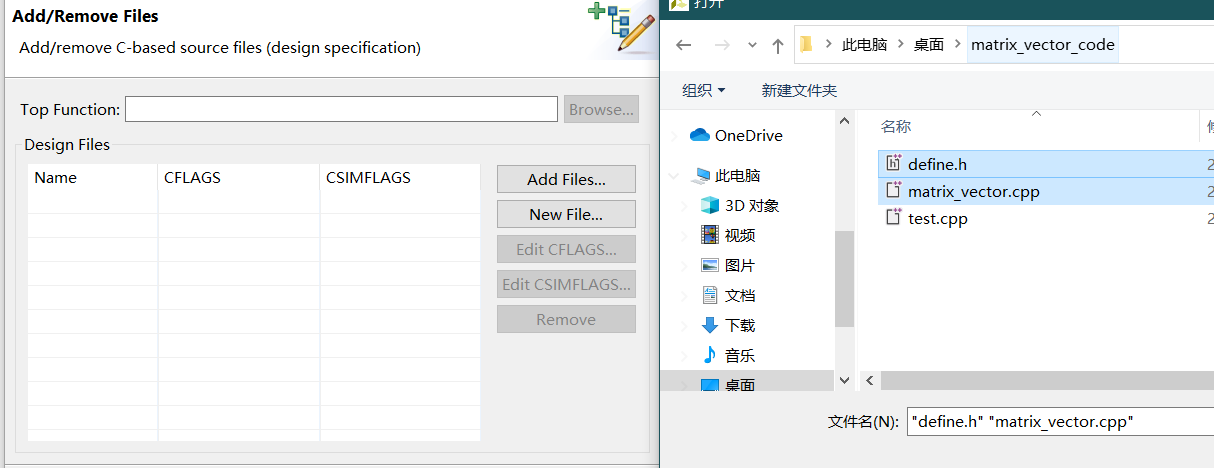
1. 如何通过HLS定义PYNQ-Z1 Overlay（示例）

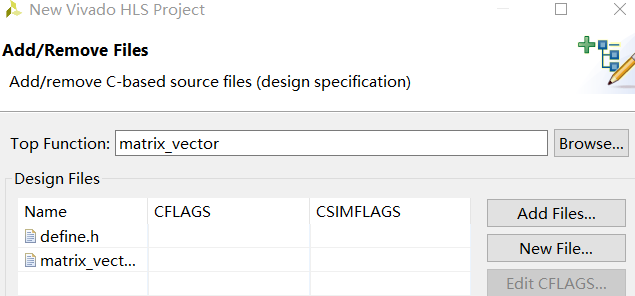
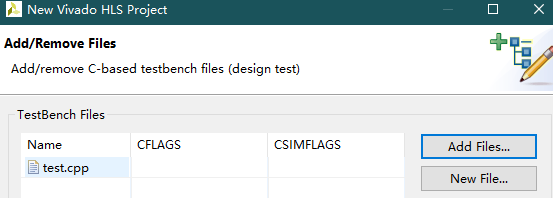
以矩阵向量乘法为例，下面给出其在PYNQ-Z1上的完整优化流程。

**①打开Vivado HLS，建立工程。**

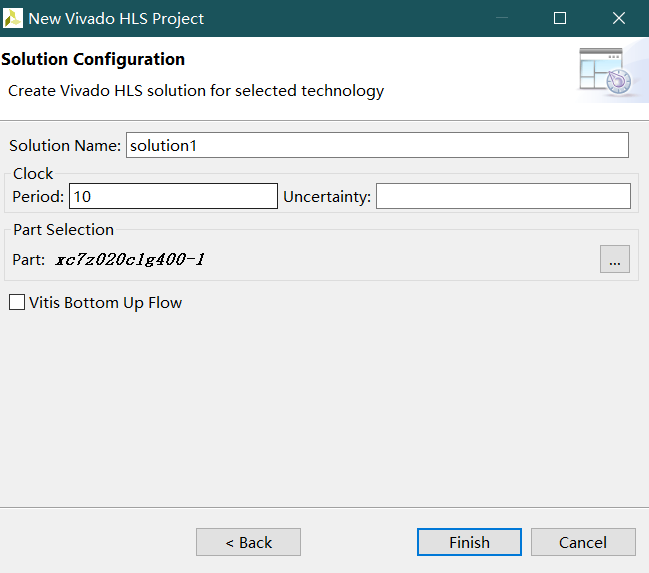
 

**②导入矩阵向量乘法的顶层函数和测试文件。**

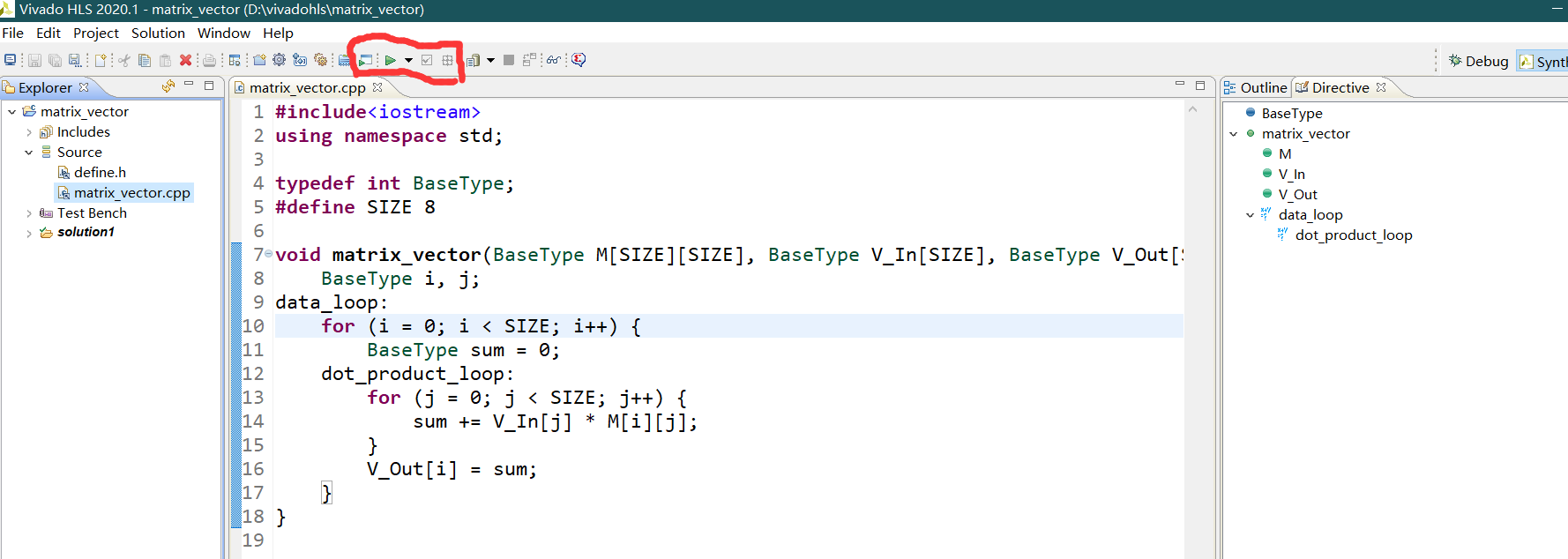


**③新建solution，选取PYNQ-Z1对应的part:xc7z020clg400-1。**

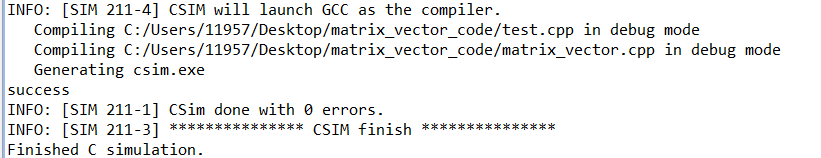


**④进入界面后，打开matrix\_vector顶层函数文件，便可对该函数进行优化设计。**

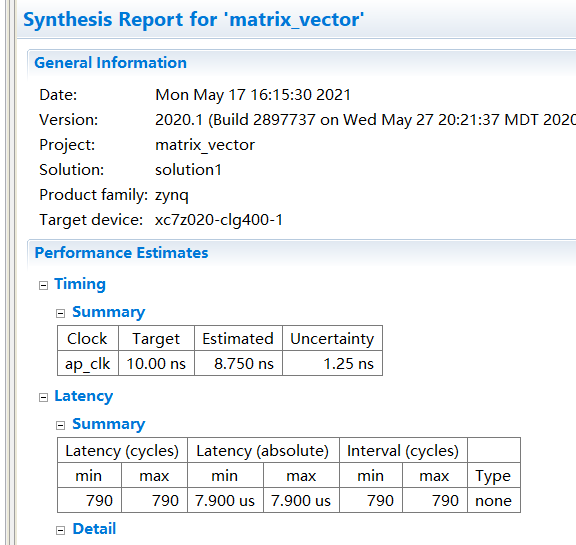
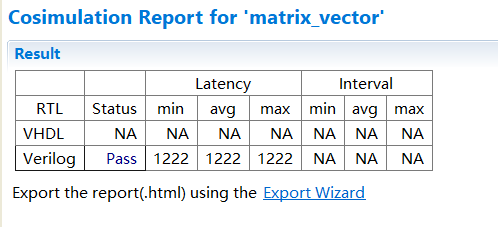


**⑤进行HLS的基本流程:C仿真、C综合、C\RTL联合仿真、导出IP核，所有操作只需要要点击上述按钮，其他选项默认即可。结果如下。**

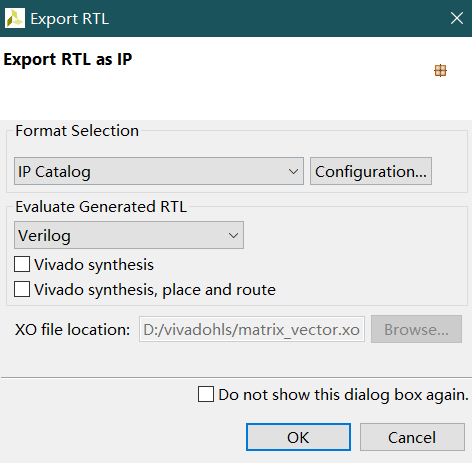
C仿真结果：



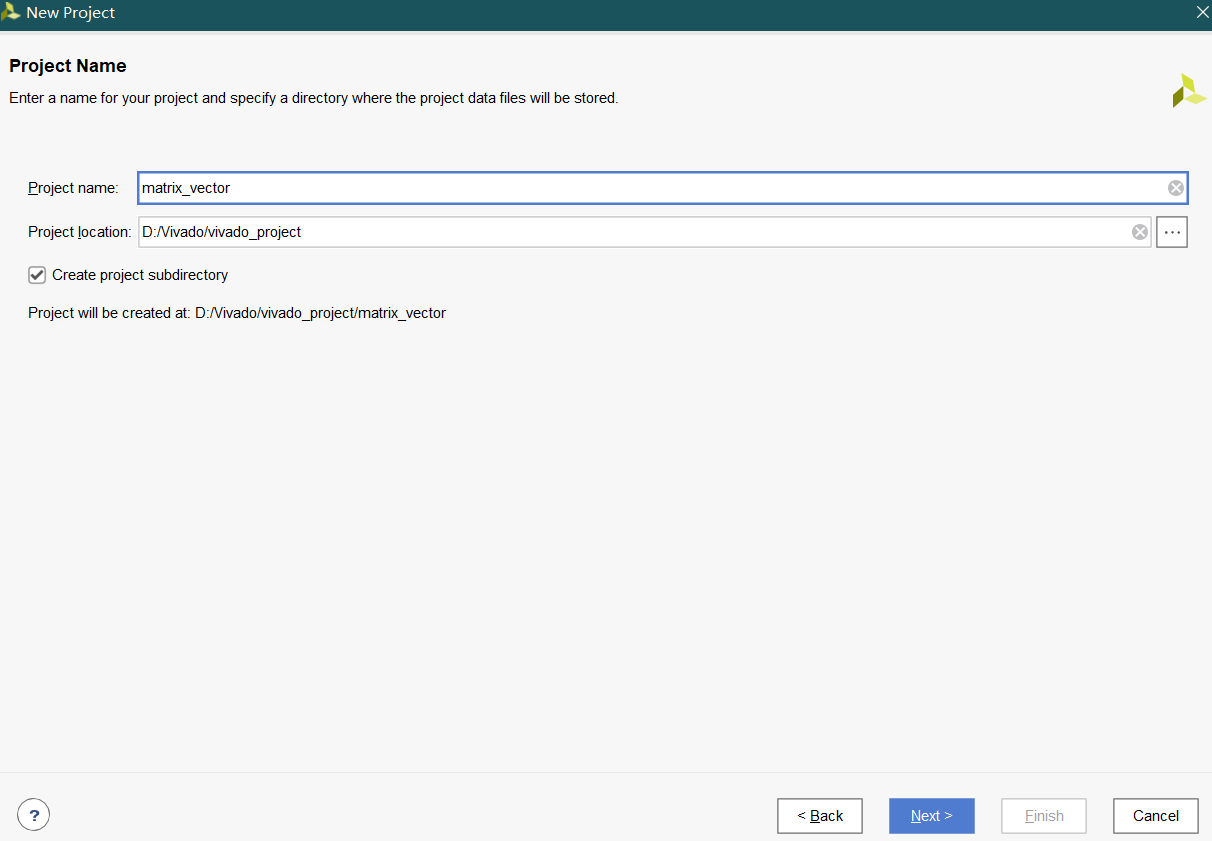
C综合和C\RTL联合仿真结果：

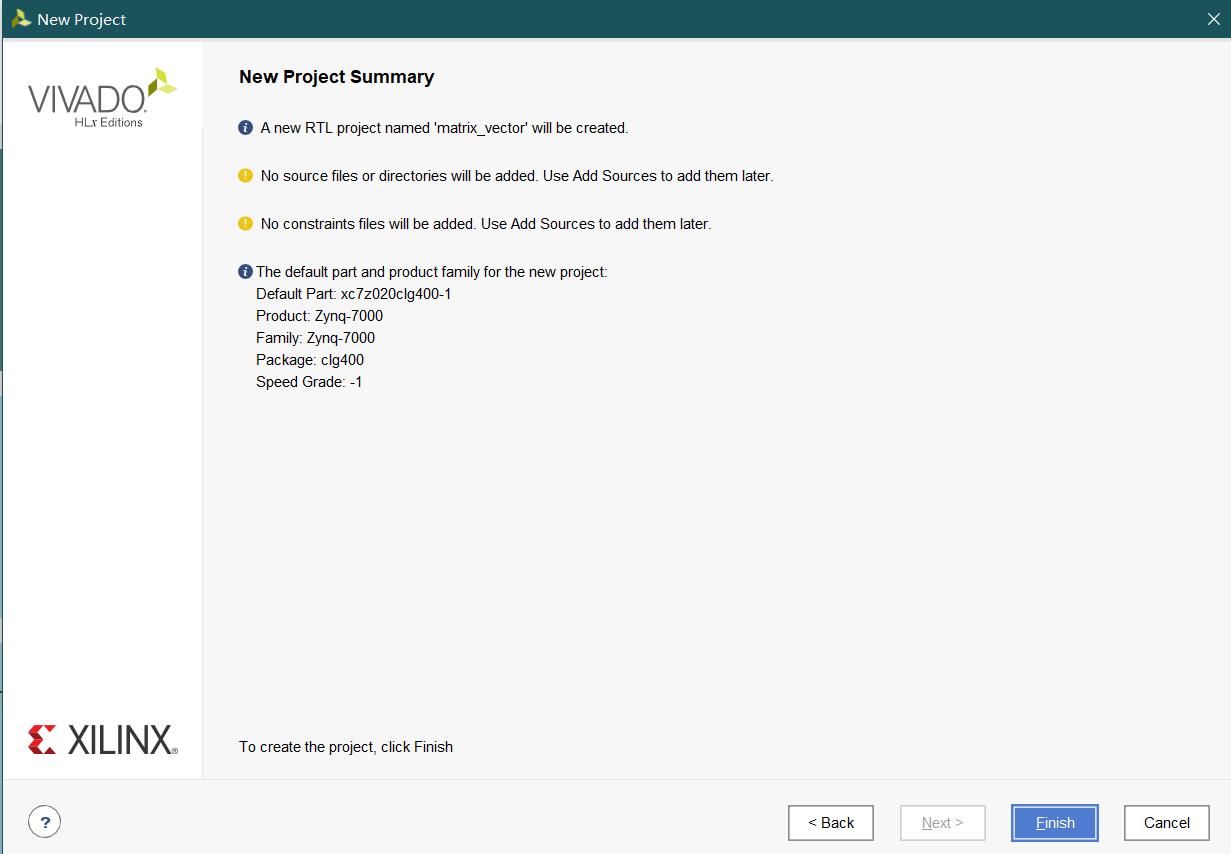
导出IP核（点击OK后等待几秒钟即可）：



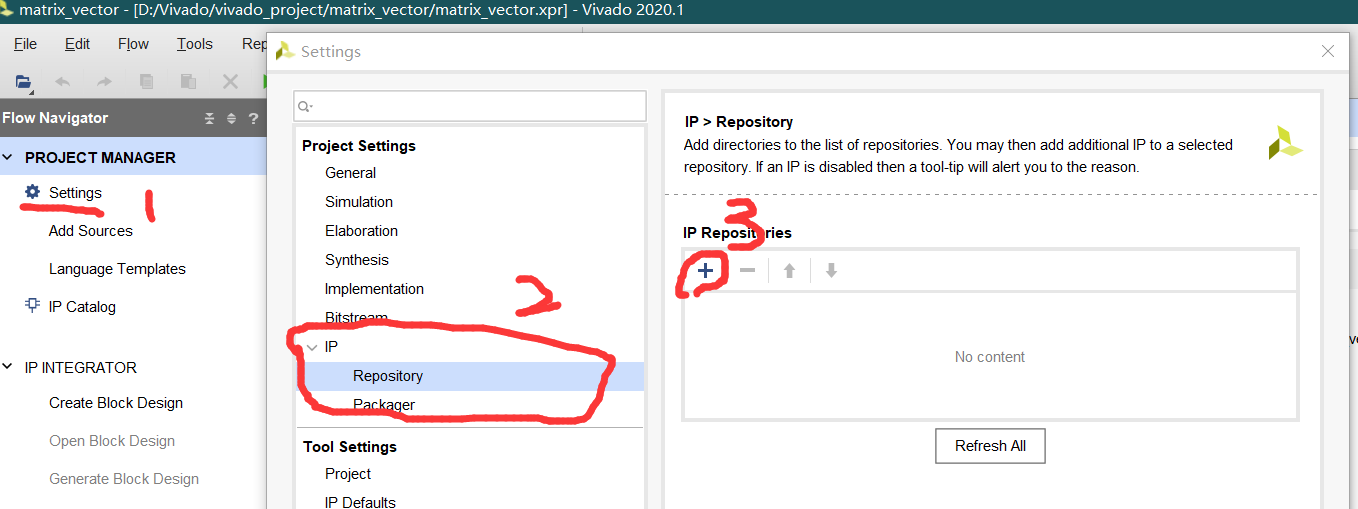
**⑥打开Vivado软件，建立新工程并命名，后面在选part之前一直点next就行。**



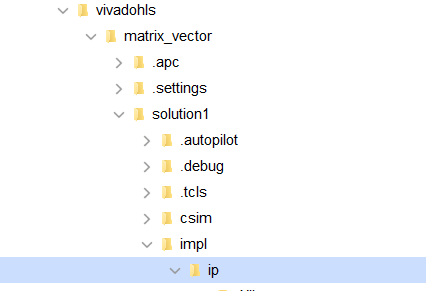
**⑦选择和Vivado HLS中一样的part:xc7z020clg400-1，最终界面应该如下所示。**



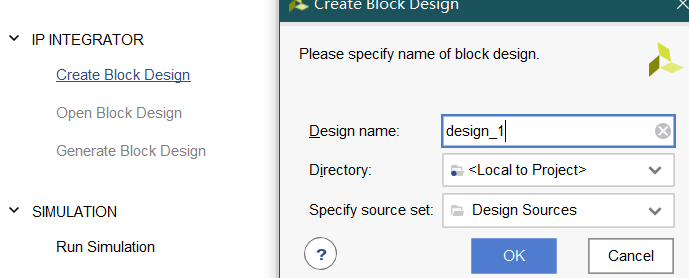
**⑧进入Vivado界面后，按下图顺序在设置中选取刚刚HLS导出的IP。**



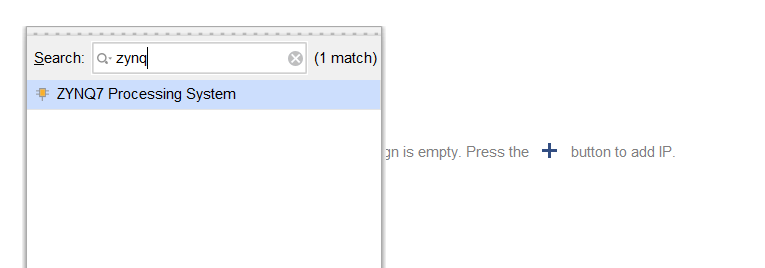
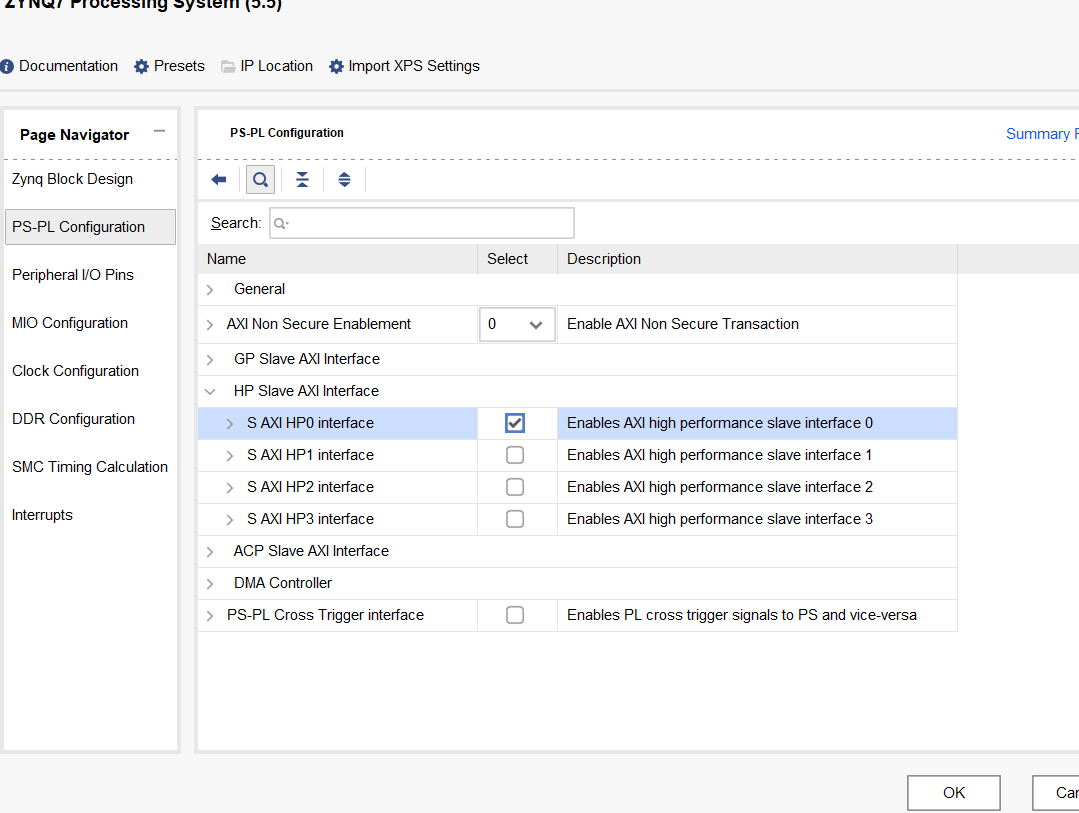
导出的IP对应文件夹在HLS项目的文件夹中，如图：



**⑨新建一个Block Design：**

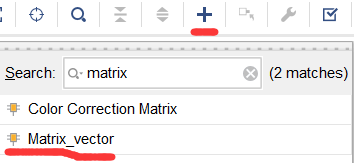


**⑩点击空白部分中间的＋号，查找ZYNQ7 Processing System导入Design中，然后双击刚刚导入的ZYNQ7，在PS-PL Configuration中将S AXI HP0 interface打勾。完成后，在上方绿色的提示中点击Run Block Automation进行自动连线。**

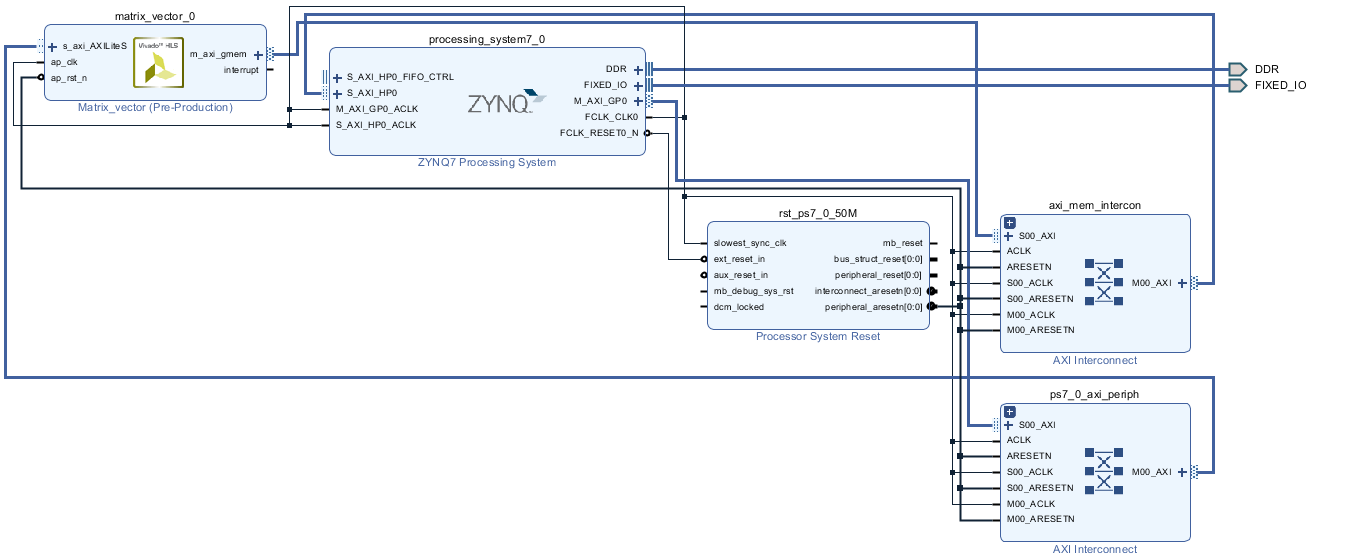
 

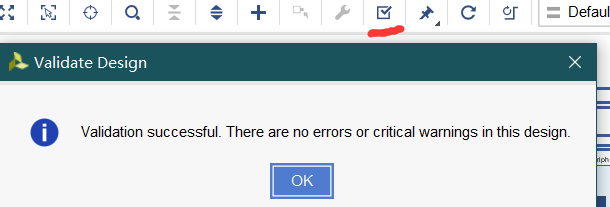
C:\Users\DELL\Documents\Tencent Files\1195750722\Image\C2C\6T1(15WF4OBV3DI`(KM3{NJ.png

**⑪点击上方的＋号添加刚刚在HLS中设计的IP核，IP核名字为顶层函数的名字（这里为matrix\_vector），然后继续点击Run Block Automation进行两次自动连线，如下图所示。**

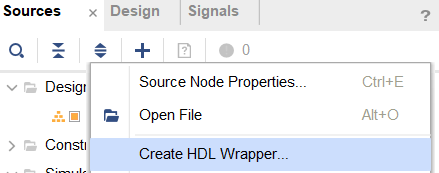


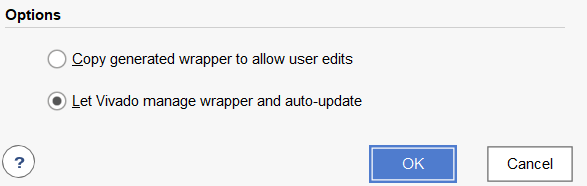
**⑫自动连线中将能打勾的全部打勾，最终得到类似如图所示的Design，点击位于＋号右方的Validate Design按钮**C:\Users\DELL\Documents\Tencent Files\1195750722\Image\C2C\$)P0}47IB5I]XPL5@5TXP1I.png **(或者按F6)，验证成功即表示设计完成。**



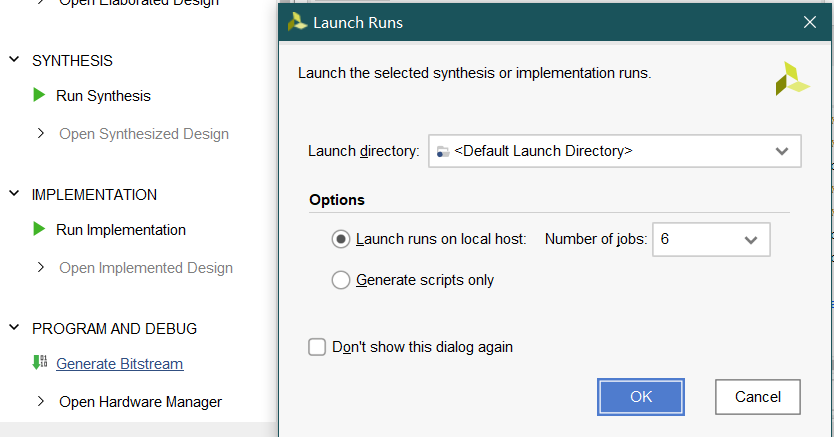


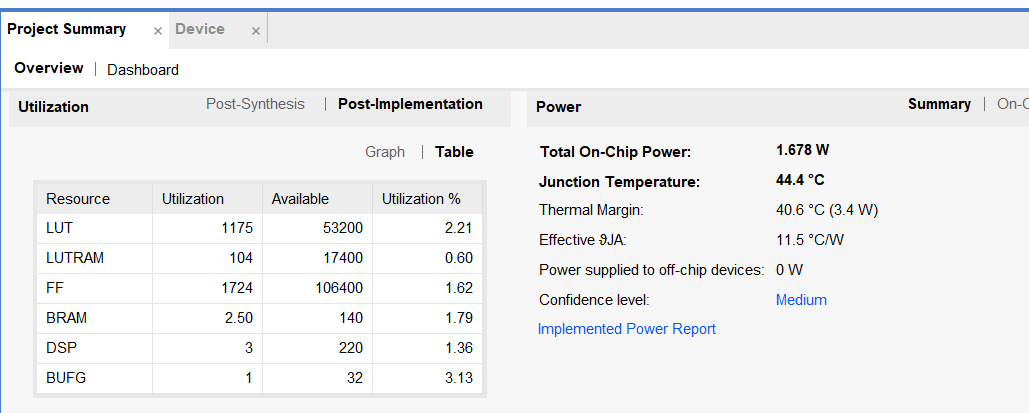
**⑬右击Design Sources中的design\_1,点击Create HDL Wrapper进行封装。**





**⑭点击左下方Generate Bitstream导出比特流，会消耗几分钟时间，耐心等待。成功后，选择Open Implemented Design，在Project Summary中可以看到Implement后的实际的资源消耗情况，如图中所示。**

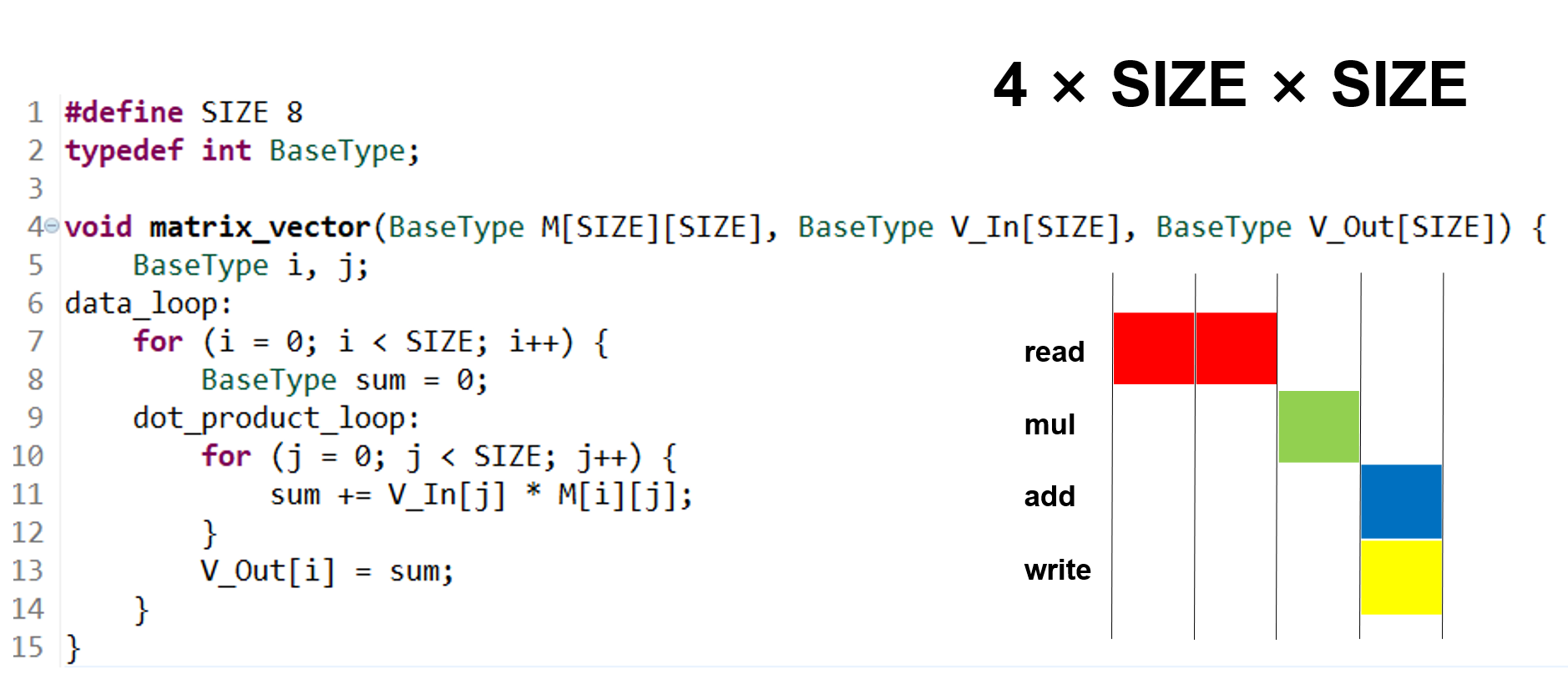




**本次实验仅要求到⑭步，不要求上板测试实际运行时间，完整的PYNQ-Z1上板流程请参考附录。**

1. 矩阵向量乘HLS优化思路

下面是矩阵向量乘的c程序，其中M为8\*8的矩阵，V\_In为含有8个元素的初始向量，V\_Out为执行乘法后得到的含有8个元素的向量。不加任何directive指令，该程序的每一个内层循环dot\_product\_loop的运行流程如下图右方所示（可以在Vivado HLS综合后点击右上方的Analysis自行观察）。对于每一次循环，至少需要4个时钟周期，而内层循环dot\_product\_loop会循环8次，外层所以对于整个程序，至少需要4\*8\*8个cycles。（这里说至少是因为没有考虑存储i,j等变量的时间）。可以通过c综合报告观察资源和性能。



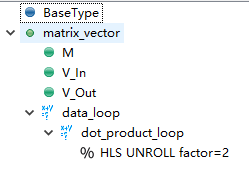
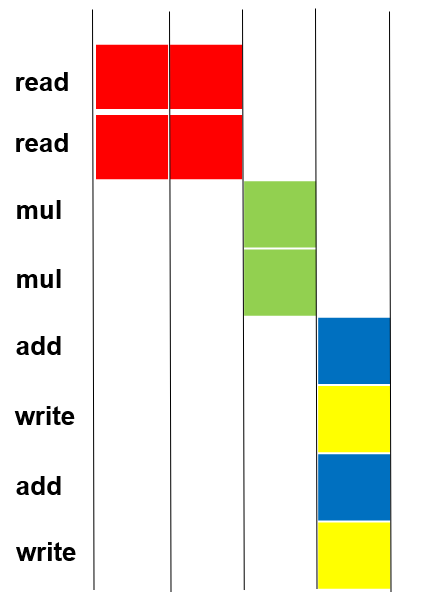
为了进行优化，我们可以建立新的Solution，建立新的Solution的方法是点击上方的New Solution按钮 并给新的Solution命名。

然后在右边的Directive中找到内部循环dot\_product\_loop（如果打开顶层函数时没有自动打开可以点击上方Windows->Show View->Directive），右键**添加优化指令UNROLL**,将**factor设为2**，选择**Directive File**（其中Source File指把优化指令写入代码中，Directive File指把优化指令单独写入Directive文件中。使用Source File的好处是方便代码的移植，而一般在不同的Solution进行比较时使用Directive File，这样每个Solution中的优化指令不会相互影响。）。

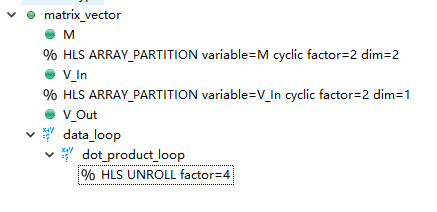
这时那么就相当于将内部的8次循环变成了4次循环，每一次循环执行两次乘法和加法，资源的使用量就会加倍，那么运行的流程如下边右图所示，这时整个程序的时间就会缩短为4\*4\*8个cycles。可以通过Compare Reports按钮 比较不同Solution的资源和性能。

**4 × SIZE/2 × SIZE**



而当我们继续增大UNROLL的factor时会从c综合报告中发现，latency并没有成倍的增长，而且资源（如DSP）的使用也并没有加倍，这是因为**BRAM的端口数量限制**。数组在FPGA内一般会自动被映射成双端口的BRAM，而双端口就意味着只能同时执行两个读或写操作。当UNROLL factor=2时，分别同时读取了V\_In和M的两个元素，因为V\_In和M分开存储在不同的BRAM上，所以能够支持同时对这两个元素进行两次读取。而当factor=4时，需要同时读取4个元素，由于端口量的限制无法做到循环展开4次，此时则需要使用**ARRAY PARTITION**指令对数组进行分割，将数组分开存储在不同的BRAM上。



因为对内部循环dot\_product\_loop展开后，同时顺序访问M不同的列，所以这里使用了cylic对M的第二维和V\_In进行分割（cyclic原理见第2小节），将M和V\_In存储在4个BRAM中，这样便能满足UNROLL factor=4时需要的4\*2=8个同时访问的端口数。此时便可以达到循环展开4次的最佳性能。可以通过综合报告观察资源的使用情况以及性能。

可以考虑将factor增大，或对外层循环data\_loop进行展开，或使用PIPELINE进行流水线操作（原理见第2小节）将此程序继续优化至最佳。

1. 实验内容

**基础任务：**

使用HLS建立工程，将已给出的SIZE=8的矩阵向量乘程序进行优化，将c综合报告（综合会自动弹出，也可在左侧Solution->syn->report打开）中的**Latency和Interval降至200以下**，在HLS优化过程中分析添加不同的优化指令后C综合报告中的资源和性能的变化情况，仿真，导出IP核。

**附加题：**

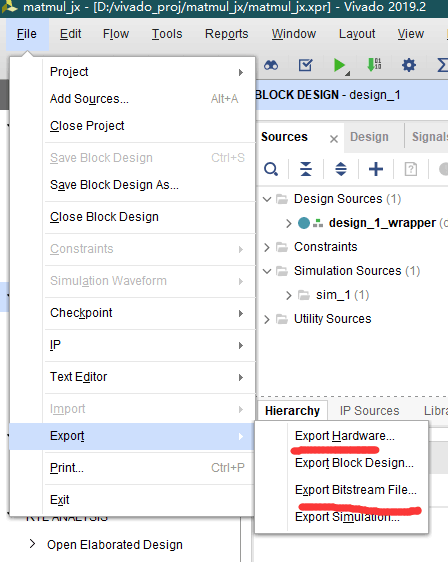
1. 将SIZE增大，改为100、1000甚至更大，再对程序进行优化，讨论是否还可以用之前的优化方式，分析原因。
2. 用同样的流程对矩阵的乘法（或其他案例，自选）进行优化加速。
3. 考察重点
4. HLS编程优化的基本方法。
5. 使用FPGA进行加速的流程。
6. 实验要求
7. **不超过2人一组完成。**
8. **有余力的同学可以自行研究附加问题。**
9. **提交形式为.zip压缩文件，命名方式为“张三-李四实验四报告.pdf”（命名方式不正确会扣分）。压缩包中需中包括修改后的代码文件和实验报告文件，实验报告格式为word或pdf，并在报告开头写上全组成员名字学号。发送到山大网盘：**

[**https://icloud.qd.sdu.edu.cn:7777/link/37124C7BE0BB351702AA5F5E2F94C973**](https://icloud.qd.sdu.edu.cn:7777/link/37124C7BE0BB351702AA5F5E2F94C973%20)

1. **提交报告的截止时间为5月30日晚上12点前。晚交的报告将会被扣分。**
2. **所有雷同的实验报告（包括和之前高年级提交的实验报告雷同），本次实验不得分。同一学生本学期出现2次及以上雷同的实验报告，本学期实验为0分。**
3. 附录

**续“如何通过HLS定义PYNQ-Z1 Overlay（示例）”①-⑭步**

**⑮在File->Export中先后选择Export Hardware(Fixed->include bitstream)，Export Bitstream File（记住bitstream保存的位置）。**

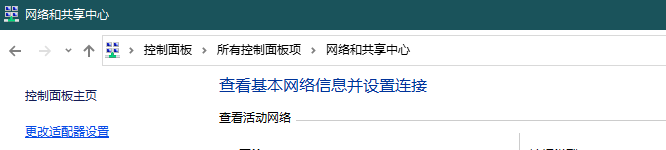


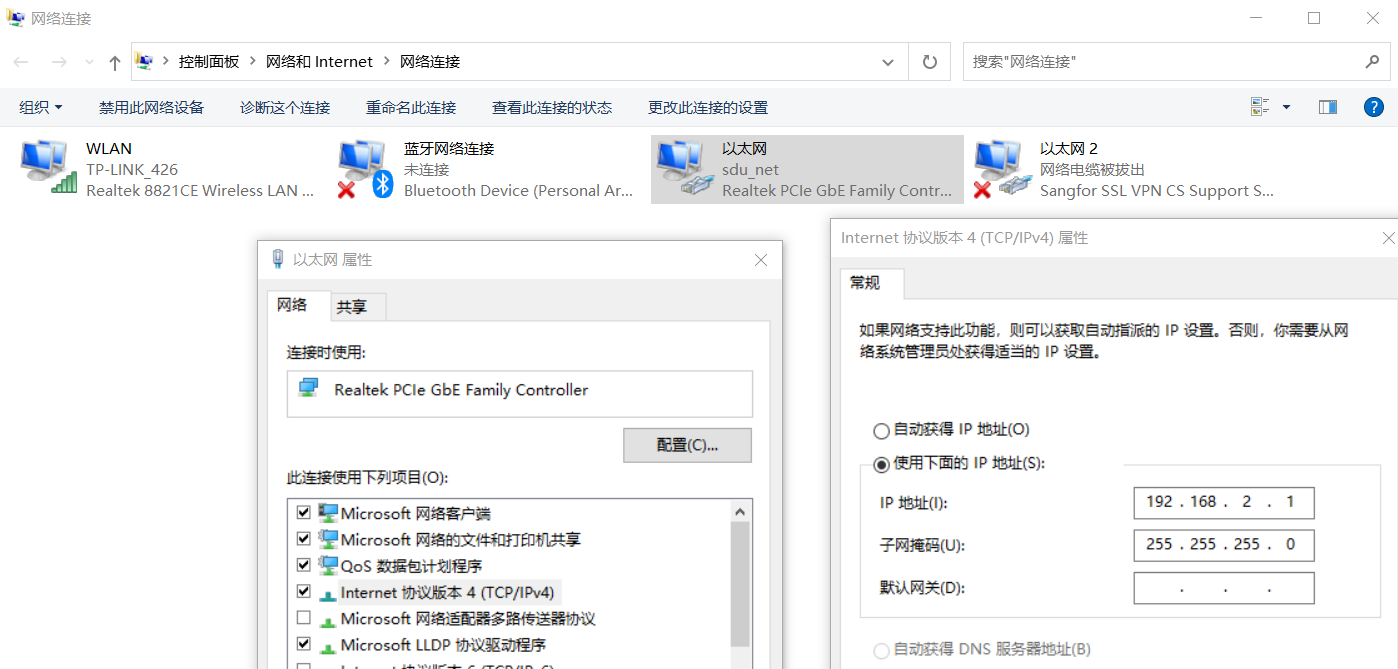
![C:\Users\DELL\Documents\Tencent Files\1195750722\Image\C2C\``E%](_GZOQ2YS8C0G2WLZQ.png](data:image/png;base64,)

**⑯顶层文件.hwh和.tcl在vivado新建的工程文件夹中可以找到，具体位置如下图。另将刚导出的Bitstream复制到此文件夹中，并将.hwh和.tcl文件改名为matrix\_vector，方便后面的操作。**

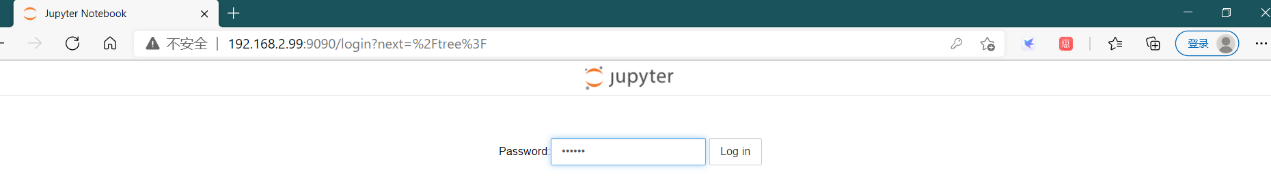


**⑰连接PYNQ-Z1板子，将USB口和网线插入笔记本电脑中即可，具体接线方式见之前给出的视频链接。打开开关，连接电脑后按下图方式配置静态IP地址，点击更改适配器选项->右击以太网->属性->Internet 协议版本 4 ->使用下面的IP地址，按图中输入。**



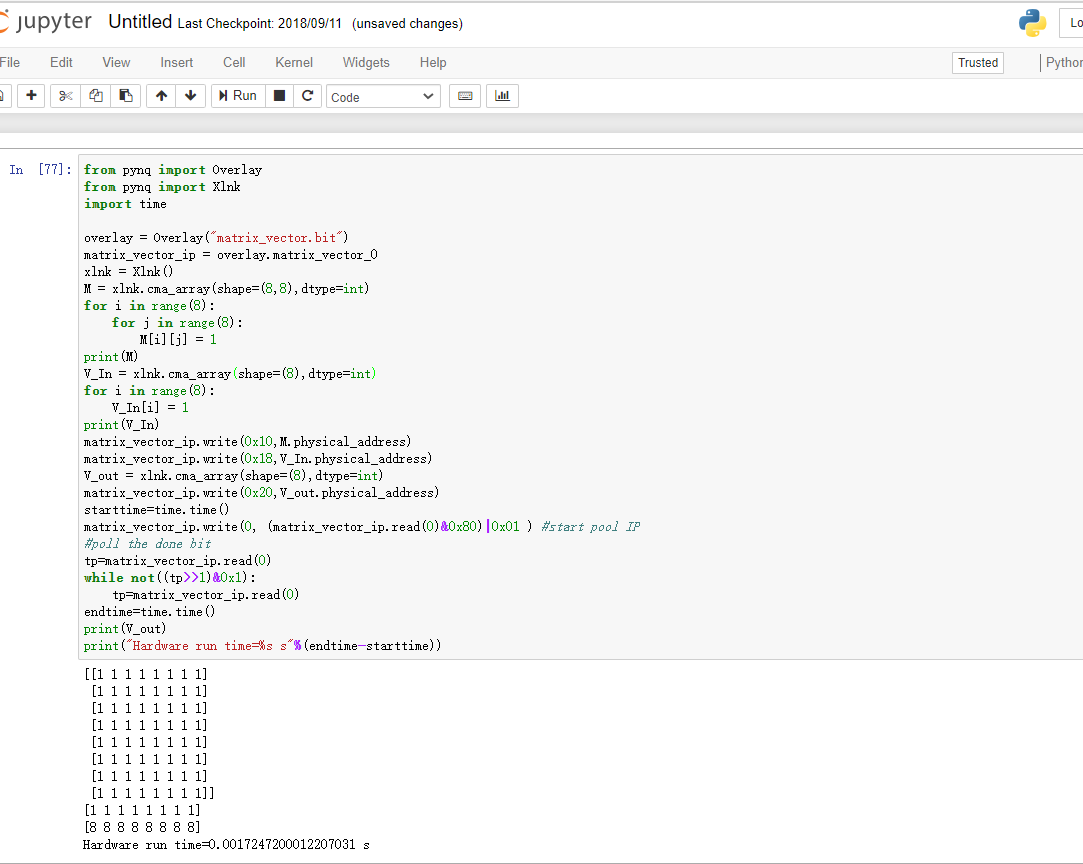


**⑱打开浏览器，访问**[**http://pynq:9090**](http://pynq:9090)**，如果网络配置正确可以进入下图的界面，用户名和密码默认都为xilinx。**



**⑲新建文件夹，将顶层文件和bitstream文件导入，并新建一个python3文件，通过python代码进行读写，调用ip核运算，运行后观察并分析结果。python文件示例见附件。**



****