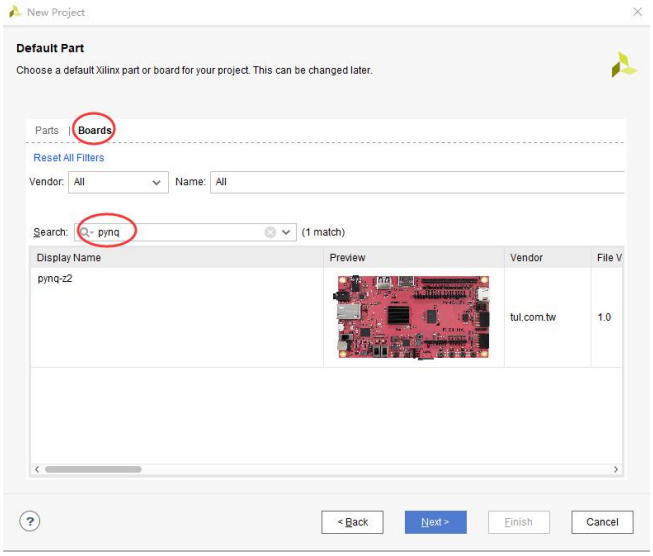
1. 添加板卡文件。由于本次实验采用的是 PYNQ Z2 板子，Vivado 中默认库中是没有此板子的型号的，所以第一次打开时，需要先添加板卡文件。（只要没重置电脑，后续就不需要再添加了）

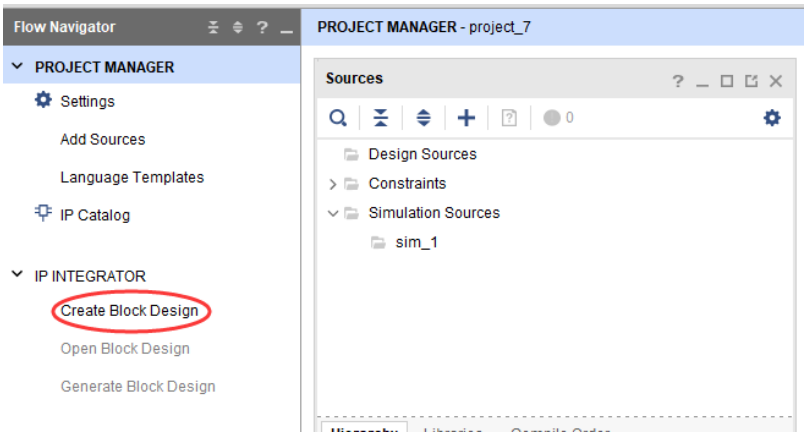
操作：首先找到 Vivado 的安装目录，将 pynq-z2 文件夹放入到“PATH/data/boards/board_parts/zynq/”，如图

2. 打开 Vivado，依次选择 Create Project > Next ,修改项目名称和项目位置，点击 Next，选择 RTL Project，并打勾 Do not specify sources at this time>Next,点击上面的 Boards，如图



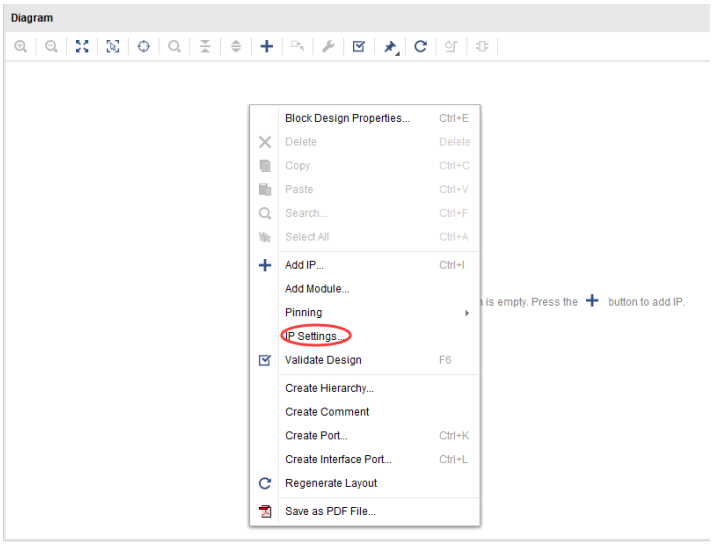
点击 pynq-z2>Next>Finish。

3.点击左侧 IP INTERATOR/Create Block Design

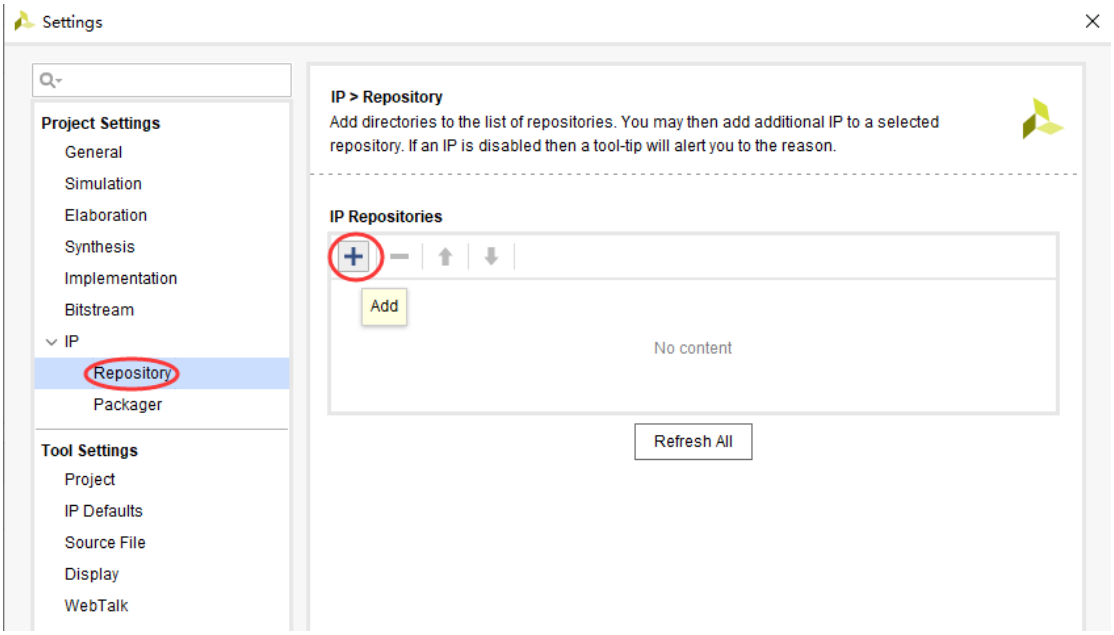


修改模块名称，点击 OK。

3. 添加 IP 核。右键 Diagram 空白处，点击 IP setting。



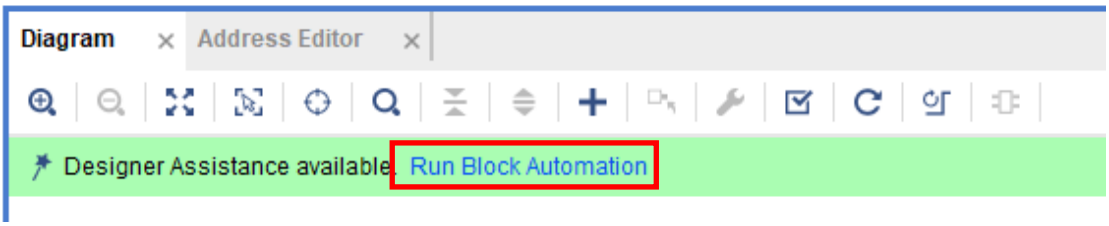
点击 IP/Repository/Add



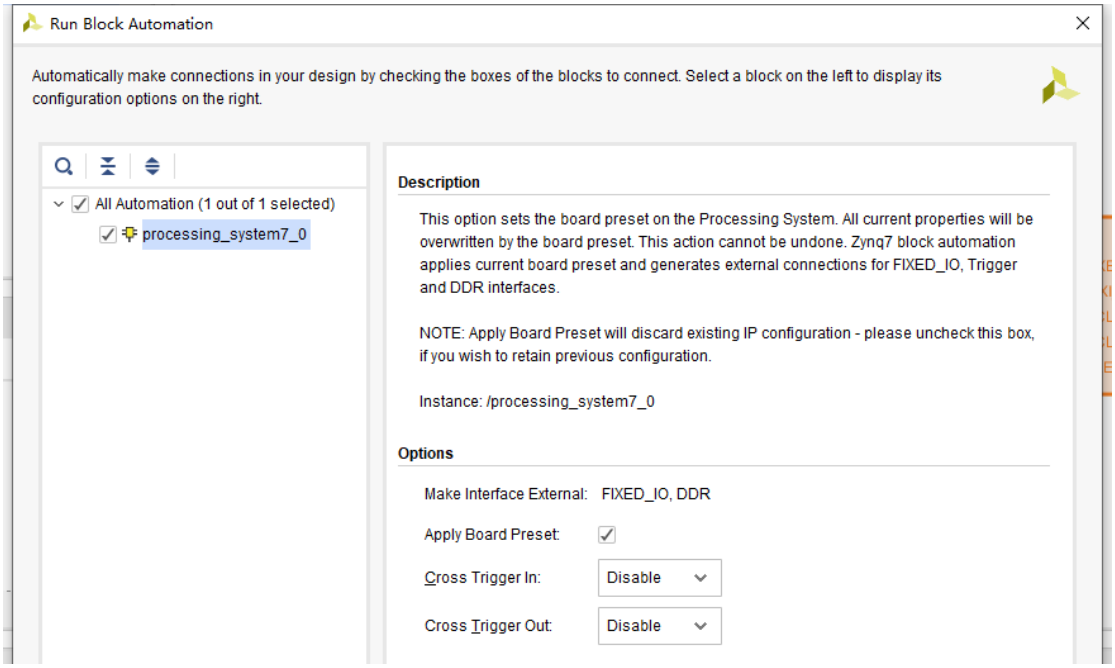
选中 PATH/ vivado-library-master（其中 PATH 为 vivado-library-master 的存储路径），点击 Select> OK>Apply>OK

4. 右键 Diagram 空白处，Add IP，搜索 zynq7，双击 ZYNQ7 Processing System 添加 IP。

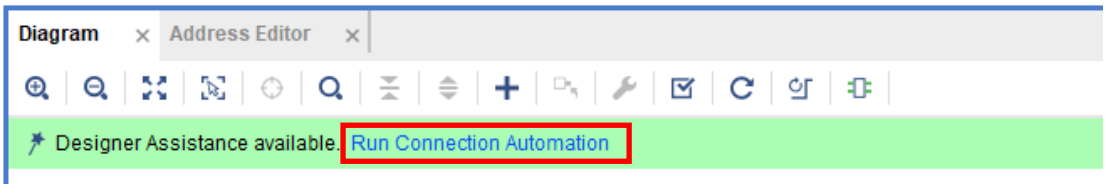
5. 点击上方的 Run Block Automation



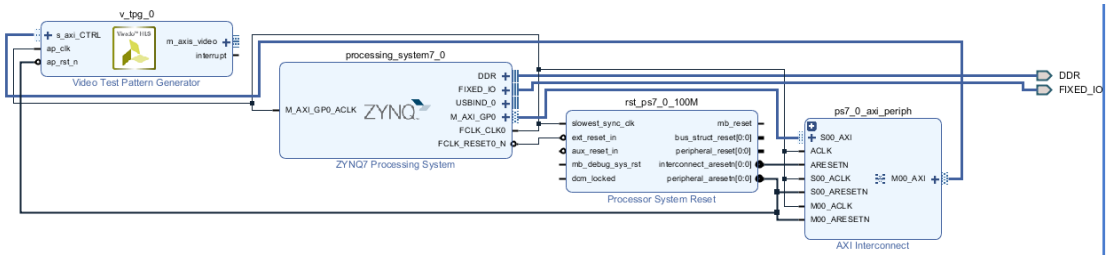
6.1 保证 Apply Board Preset 勾选，并点击 OK 确认



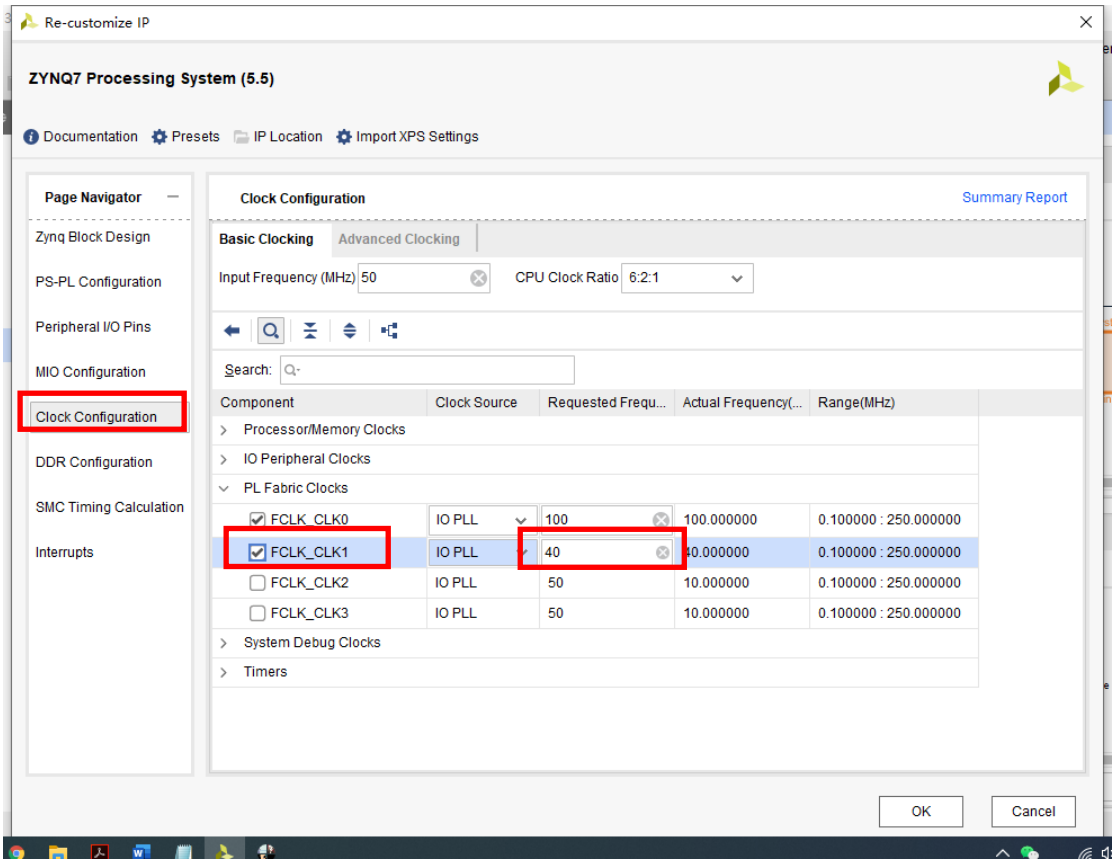
6. 右键 Diagram 空白处，Add IP，搜索 Video Test Pattern Generator，双击 Video Test Pattern Generator 添加 IP。
7. 点击上方的 Run Connection Automation



最终如下



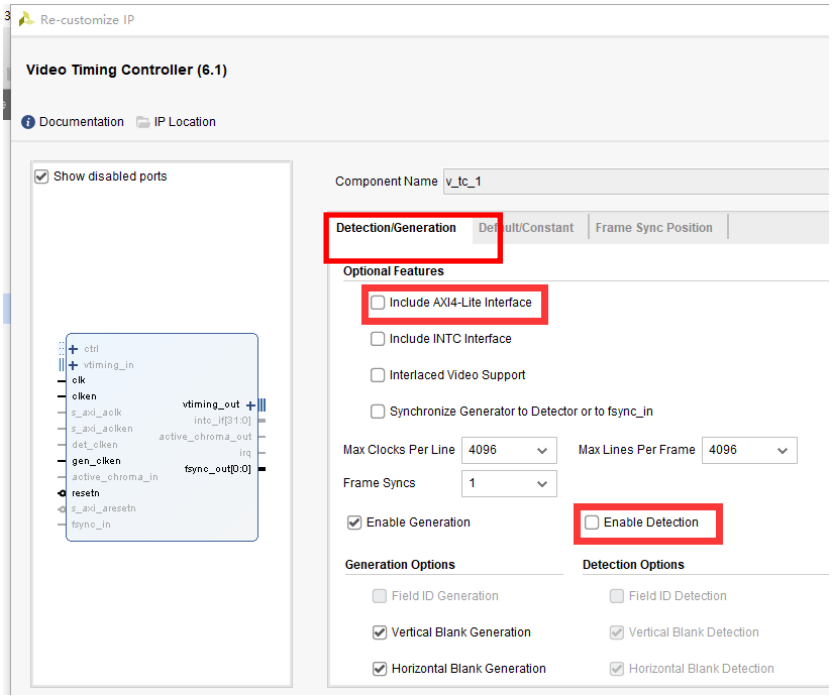
8. 双击之前添加的 ZYNQ7 Processing System 模块进行配置
   8.1 点击左侧 Clock Configuration>PL Fabric Clocks，勾选 FCLK_CLK1，将其 Request Frequency 设置为 40MHz



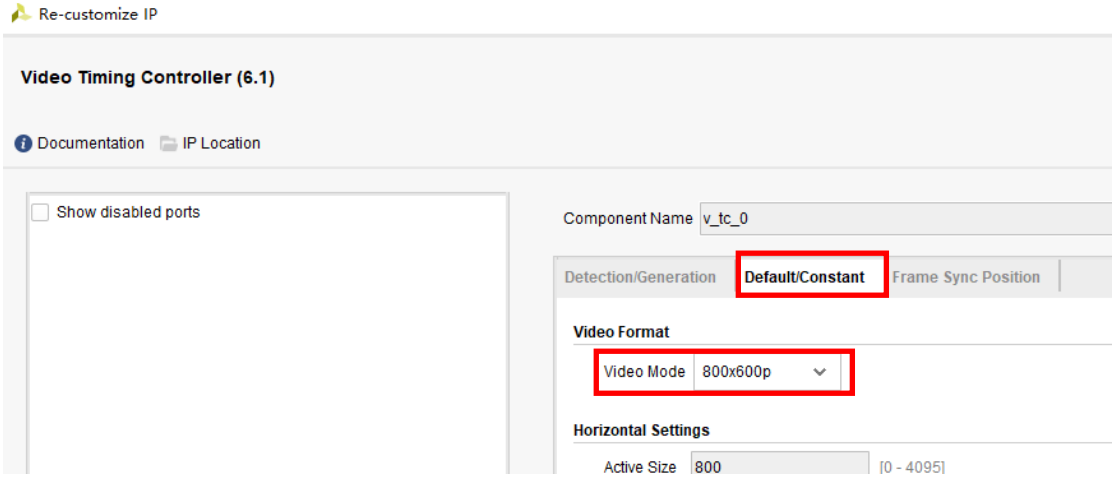
8.2 点击 OK 完成配置

9. 右键 Diagram 空白处，Add IP，搜索 Video Timing Controller，双击 Video Timing Controller 添加 IP。
10. 双击之前添加的 Video Timing Controller 模块进行配置

    10.1 在 Detection/Generation 栏目下，取消勾选 Include AXI4-Lite Interface 和 Enable Detection

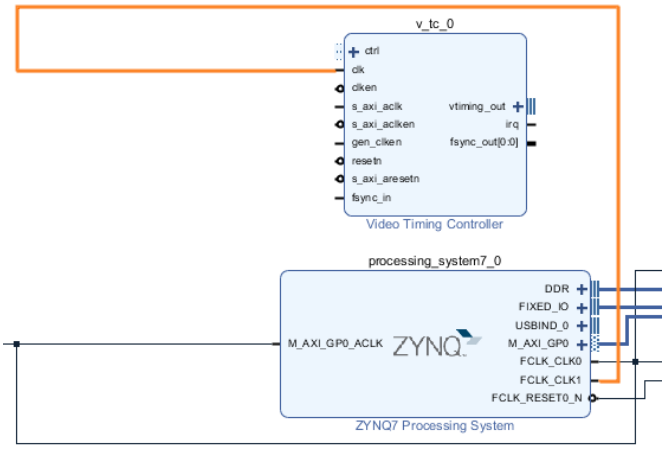


10.2 在 Default/Constant 栏目下，设置 Video Mode 为 800×600p



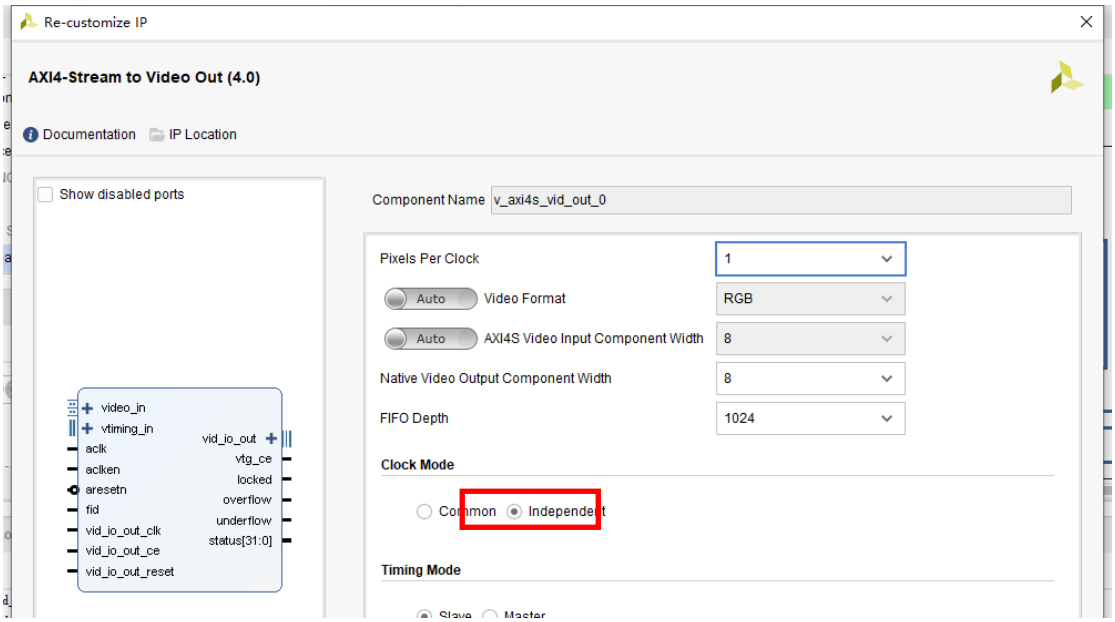
10.3 点击 OK 完成配置

11. 将 Video Timing Controller 的 clk 输入端子和 ZYNQ7 Processing System 的 FLCK_CLK1 输出端子连接起来。

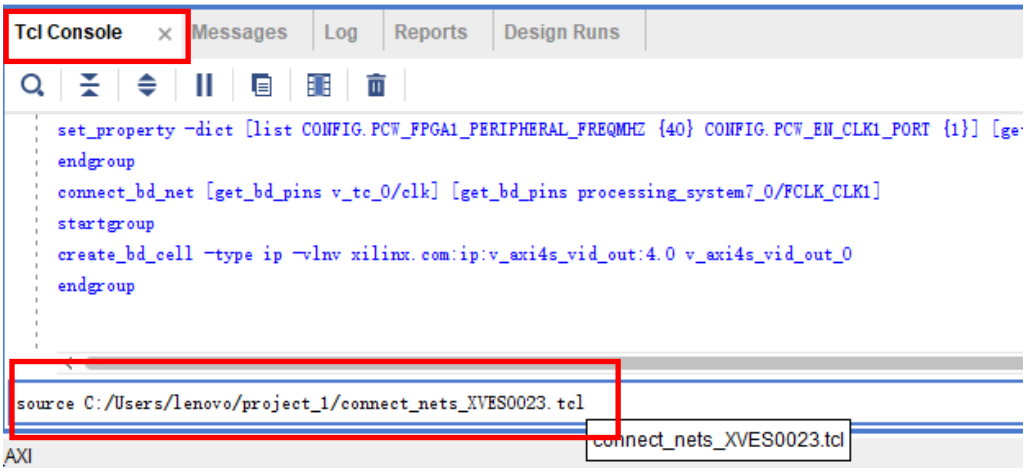


12. 右键 Diagram 空白处，Add IP，搜索 AXI4-Stream to Video Out，双击 AXI4-Stream to Video Out 添加 IP。
13. 双击之前添加的 AXI4-Stream to Video Out 模块进行配置

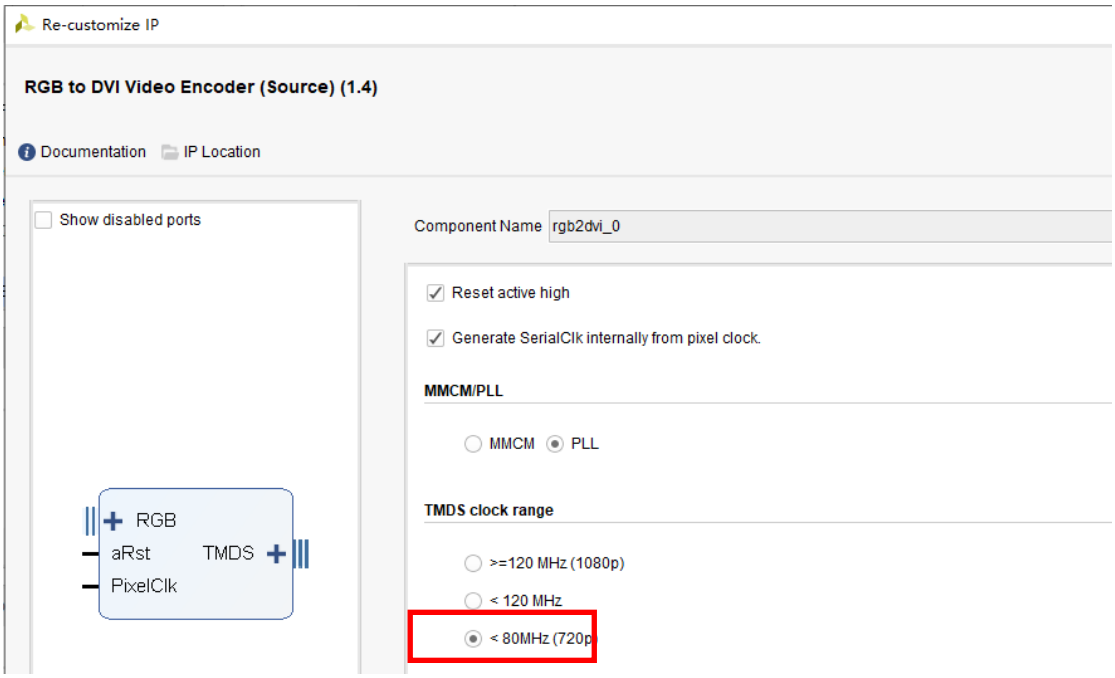
13.1 将 Clock Mode 改为 Independent



14. 在下方 Tcl Console 中，输入命令
source C:/Users/lenovo/project_1/connect_nets_XVES0023.tcl
（命令格式：source /PATH，其中/PATH 为 XVES0023.tcl 所在路径，须把“\”改为“/”）
运行 tcl 脚本 connect_nets_XVES0023.tcl【该脚本在 XVES_0023/src/tcl 中】

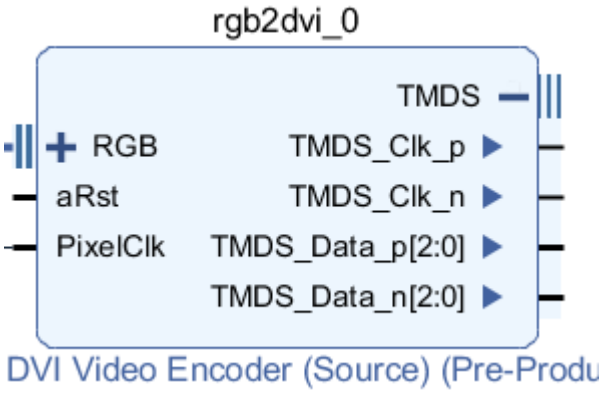


15. 右键 Diagram 空白处，Add IP，搜索 RGB to DVI Video Encoder，双击 RGB to DVI Video Encoder 添加 IP。
16. 双击之前添加的 RGB to DVI Video Encoder 模块进行配置
16.1 将 TMDS clock rang 改为<80MHz(720p)



16.2 点击 OK 完成配置

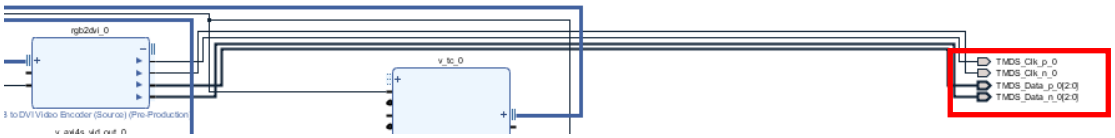
17. 将 RGB to DVI Video Encoder 的 RGB 输入端子与 AXI4-Stream to Video Out 的 vid_io_out 输出端子连接。
18. 将 RGB to DVI Video Encoder 的 PixelClk 输入端子与 ZYNQ7 Processing System 的 FLCK_CLK1 输出端子连接。
19. 左键展开 RGB to DVI Video Encoder 的 TMDS 输出端子



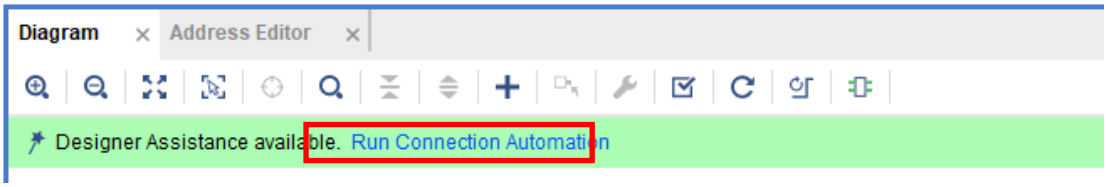
20. 对每个端子生成外部管脚

点击 TMDS_Clk_p 的名称，按住 Ctrl+T 生成管脚（注意是仅 TMDS_Clk_p 引脚变橙色才正确，如果是整个模块都变橙色说明你选中的是整个模块，需要重选，如果太小了可以 Ctrl+鼠标滚轮控制缩放）。

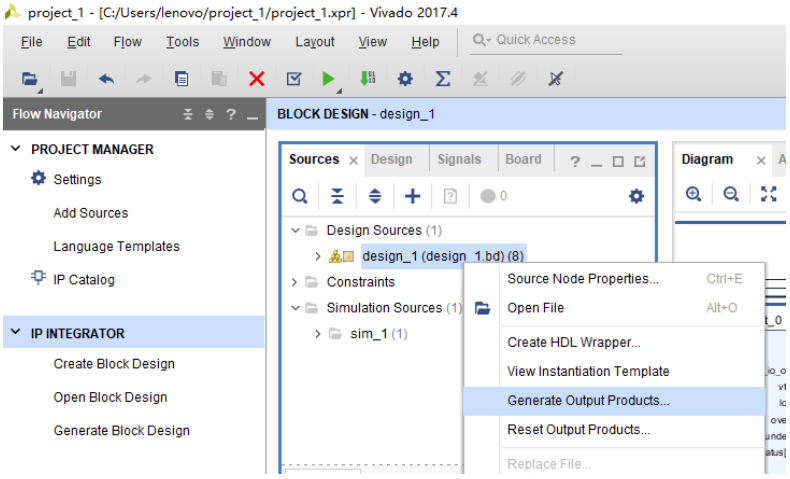
TMDS_Clk_n、TMDS_Data_p[2:0]、TMDS_Data_n[2:0]与之同理。



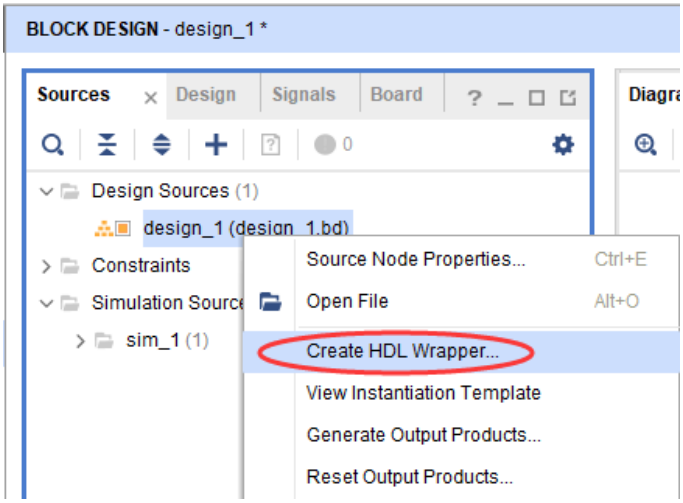
21. 再点击一次上方的 Run Connection Automation



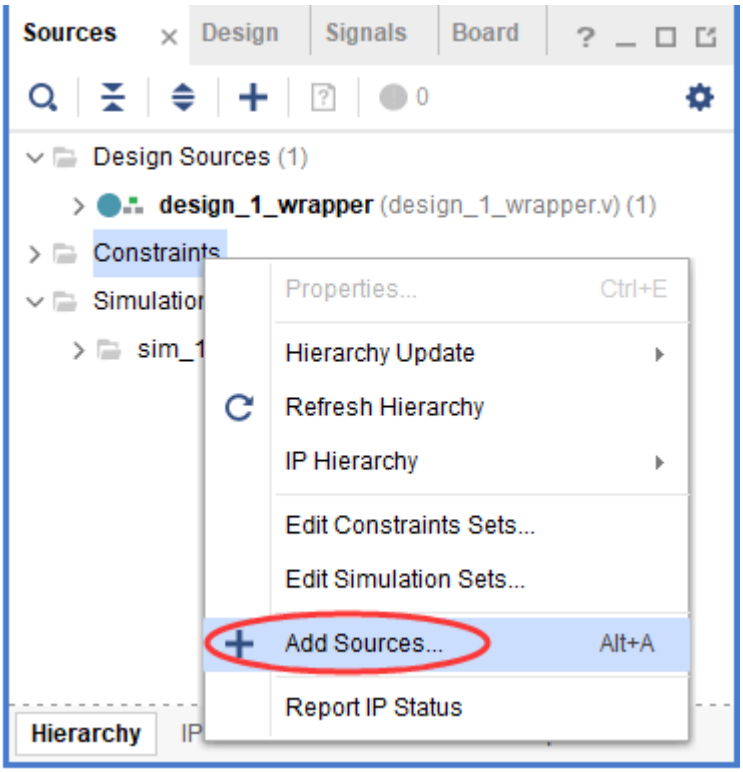
22. 点击上方验证按钮，若出现成功，则完成验证
23. 右键左侧的 Sources>Design Sources>design_1，选择 Generate Output Products



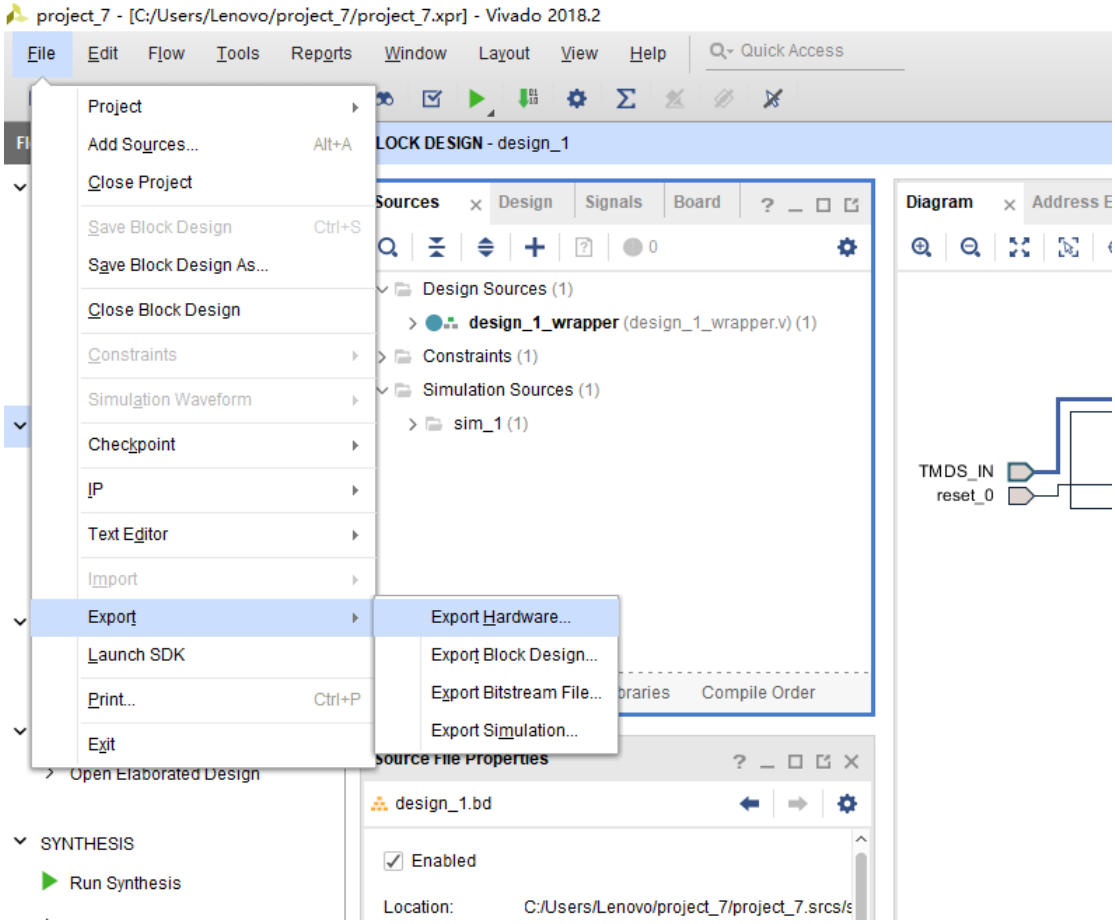
24. 右键左侧 Sources/Design Sources/design>Create HDL Wrapper



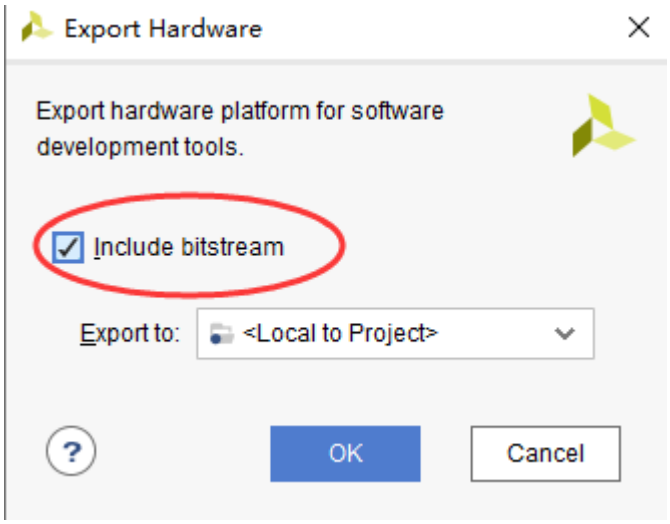
25. 右键 Sources/Constraints>Add Sources，添加约束文件。



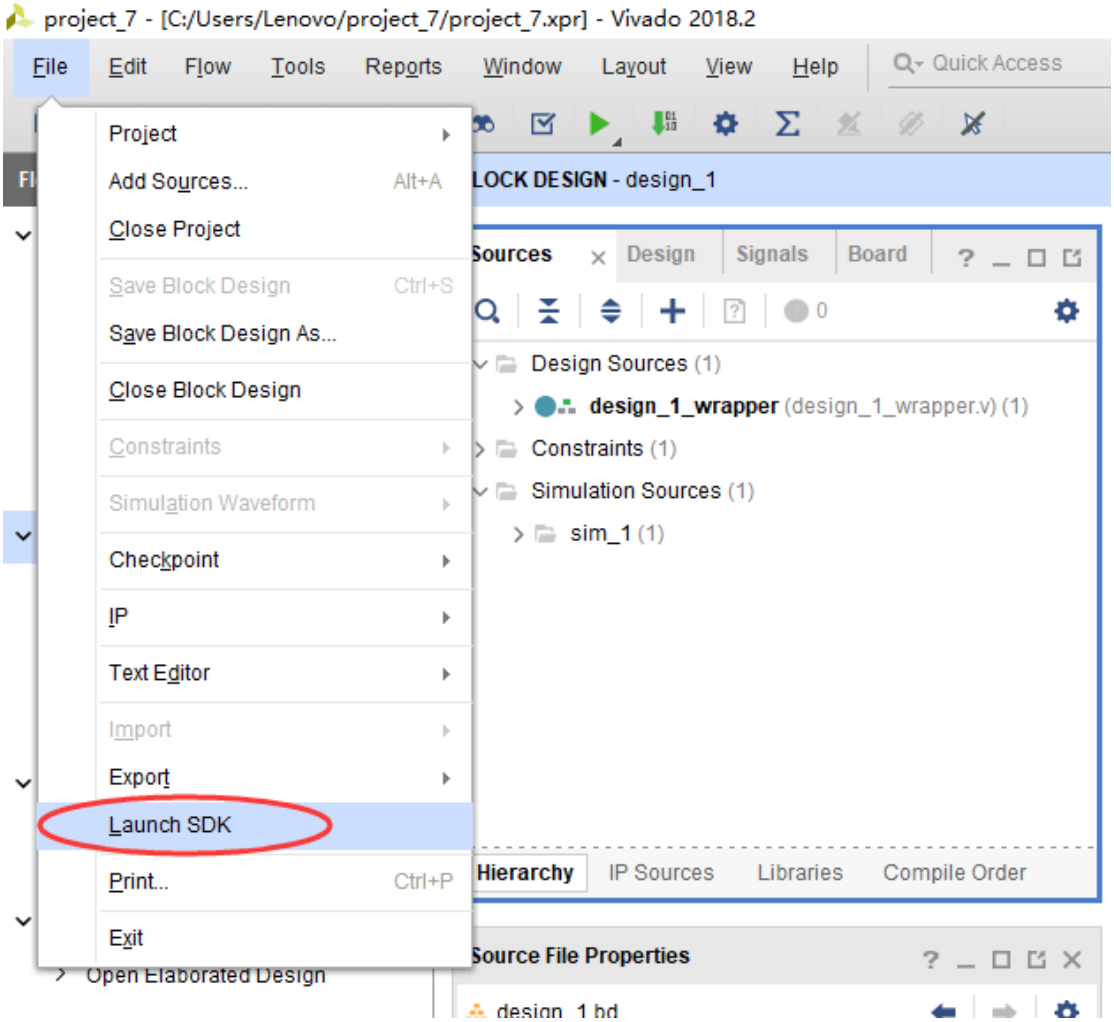
26. 选择 Add or create constraints，点击 Next，点击 Add Files，选择 XVES_0023\src\proj\src\constr 中的 pynq-z2_v1.0.xdc，点击 OK，点击 Finish，完成添加约束文件。
27. 点击左侧的 PROGRAM AND DEBUG/Generate Bitstream，点击 OK>OK,进行生成比特流。（这一步耗费的时间比较长，可以看右上角完全 ready 了，再进行下一步）
28. 跳出生成比特流成功的对话框后，点击 Cancel。点击左上角的 File/Export/Export Hardware



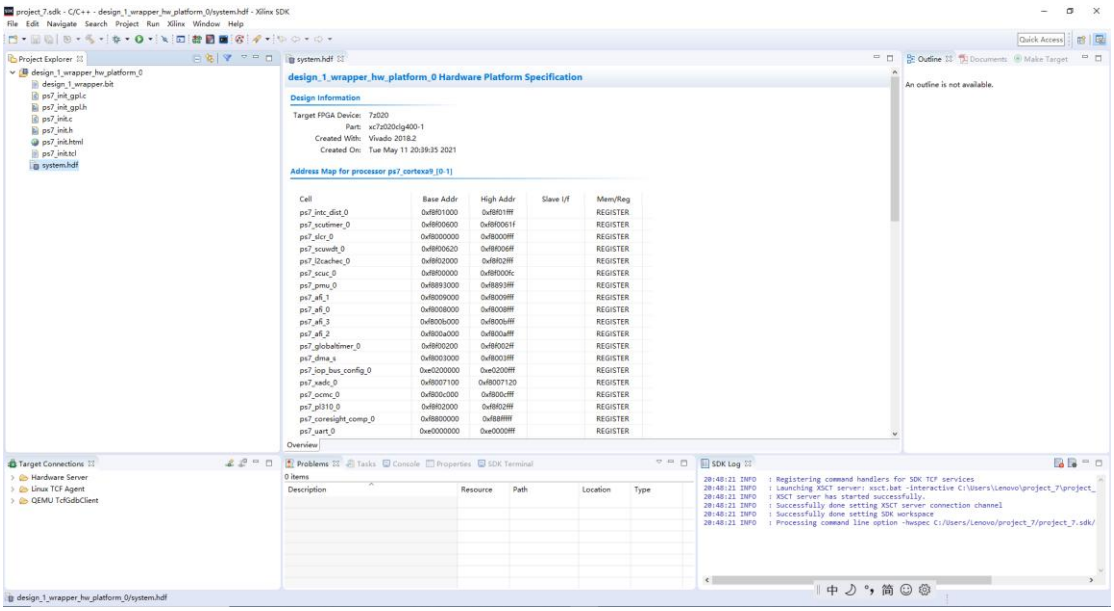
29. 勾选 Include bitstream，点击 OK



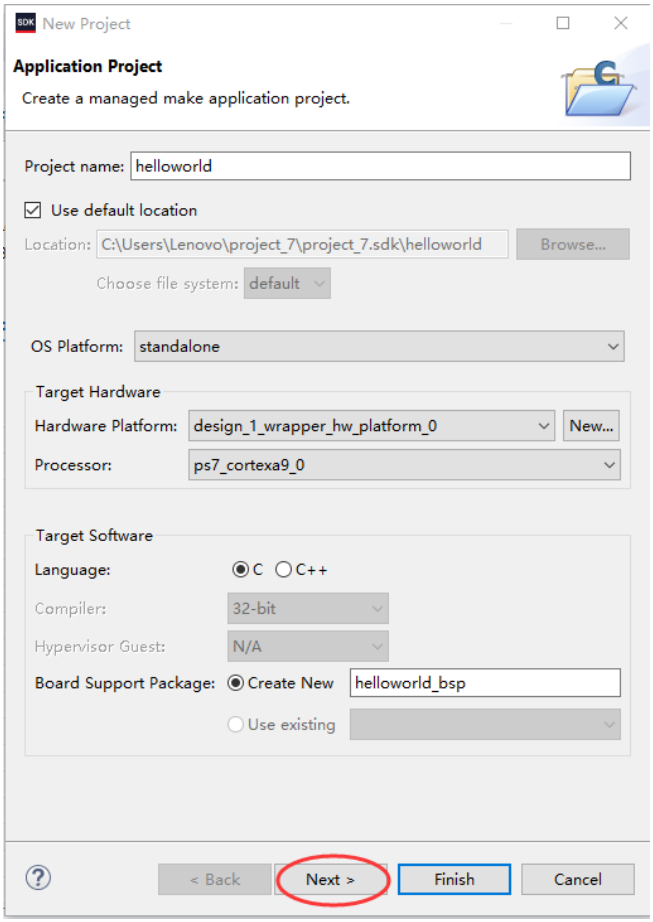
30. 点击 File/Launch SDK，对话框选择 OK。（这一步需要启动另一个软件 SDK，需要一点时间）



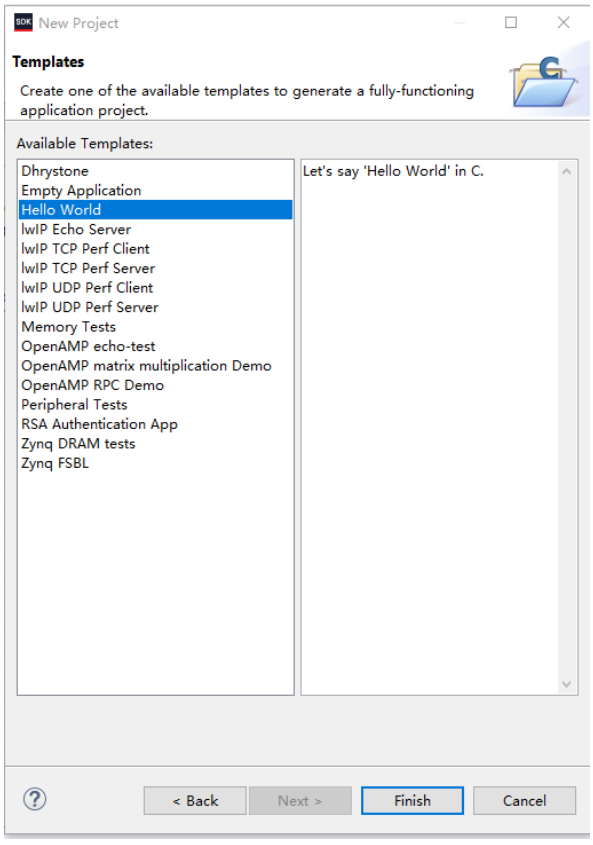
31. 成功打开 SDK 软件界面如图



32. 点击左上 File/New/Application Project，对话框中 Project name 写入 Hello world ，点击 Next。



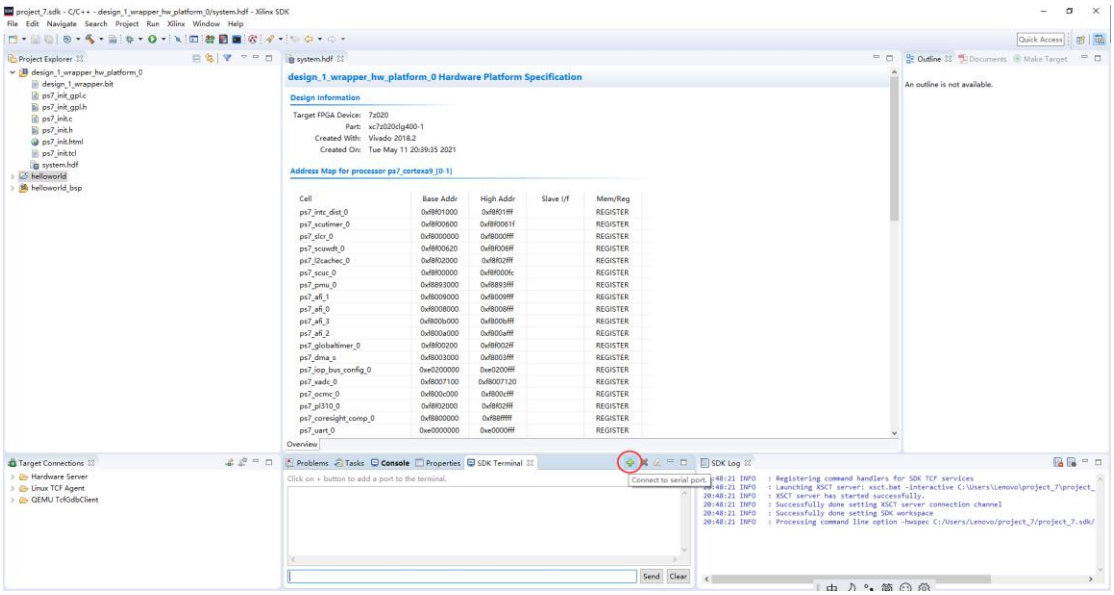
33. 选择 Hello World，点击 Finish。



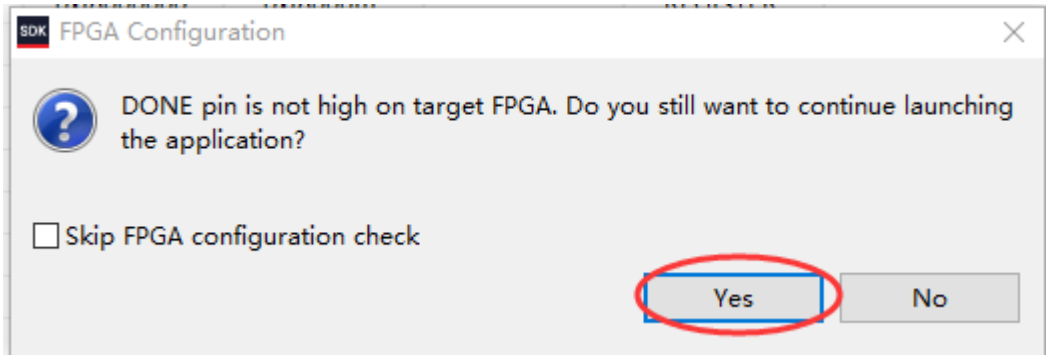
34. 将.c 文件中的代码替换如下：

```
include <stdio.h>
#include "platform.h"
#include "xil_printf.h"
#include "xv_tpg.h"

XV_tpg tpg_inst;
int Status;

int main()
{
    init_platform();

    print("Hello World\n\r");

    /* TPG Initialization */
    Status = XV_tpg_Initialize(&tpg_inst, XPAR_V_TPG_0_DEVICE_ID);
    if(Status!= XST_SUCCESS)
    {
      xil_printf("TPG configuration failed\r\n");
        return(XST_FAILURE);
    }

    // Set Resolution to 800x600
    XV_tpg_Set_height(&tpg_inst, 600);
    XV_tpg_Set_width(&tpg_inst, 800);

    // Set Color Space to RGB
    XV_tpg_Set_colorFormat(&tpg_inst, 0x0);
    //Set pattern to color bar
    XV_tpg_Set_bckgndId(&tpg_inst, XTPG_BKGND_COLOR_BARS);

    //Start the TPG
    XV_tpg_EnableAutoRestart(&tpg_inst);
    XV_tpg_Start(&tpg_inst);
    xil_printf("TPG started!\r\n");
    /* End of TPG code*/

    cleanup_platform();
    return 0;
}
```

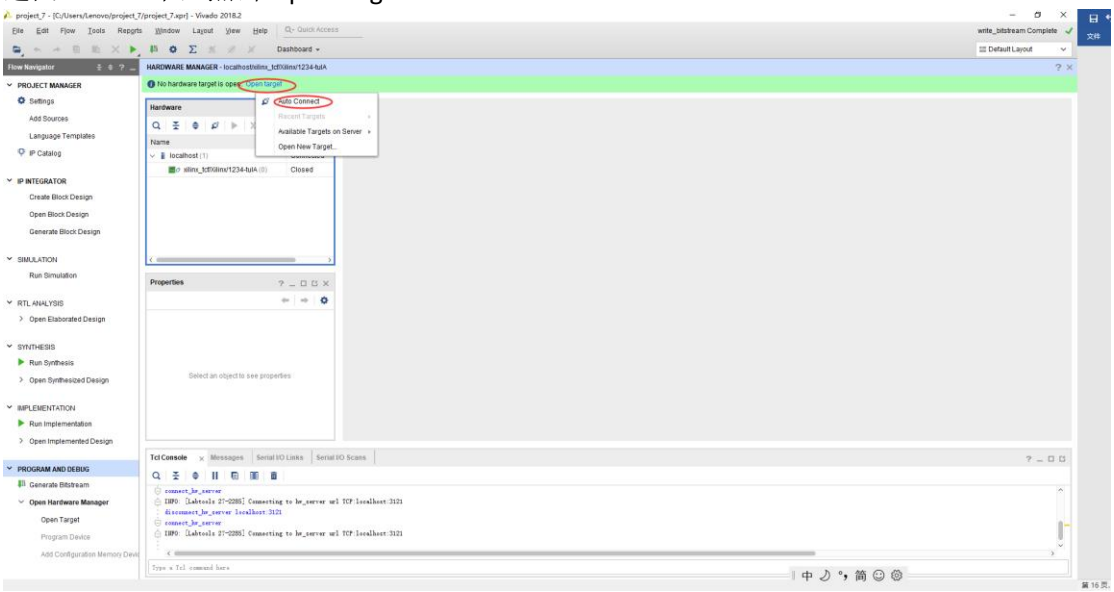
35. 返回 SDK 软件中，点击下方的 ‘+’ 按钮



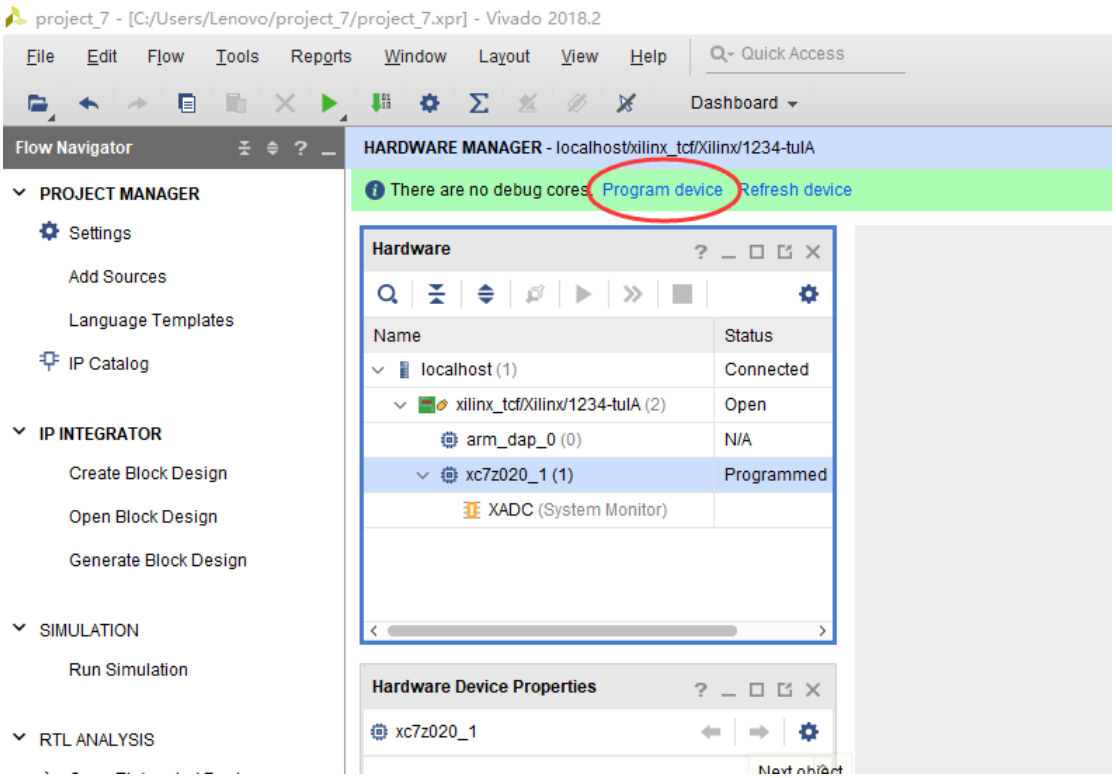
36. 对话框中 Port 选择为设备管理器中的 USB 串行接口端口，点击 OK。
37. 右键左侧的刚刚你建立的 helloworld 工程文件夹，Run As/Launch on Hardware（System Debugger）
38. 跳出的对话框选 Yes



39. 返回 Vivado 中，点击 Open target



40. 点击 Program device，点击 OK



41. 观察显示器输出，最终效果如下