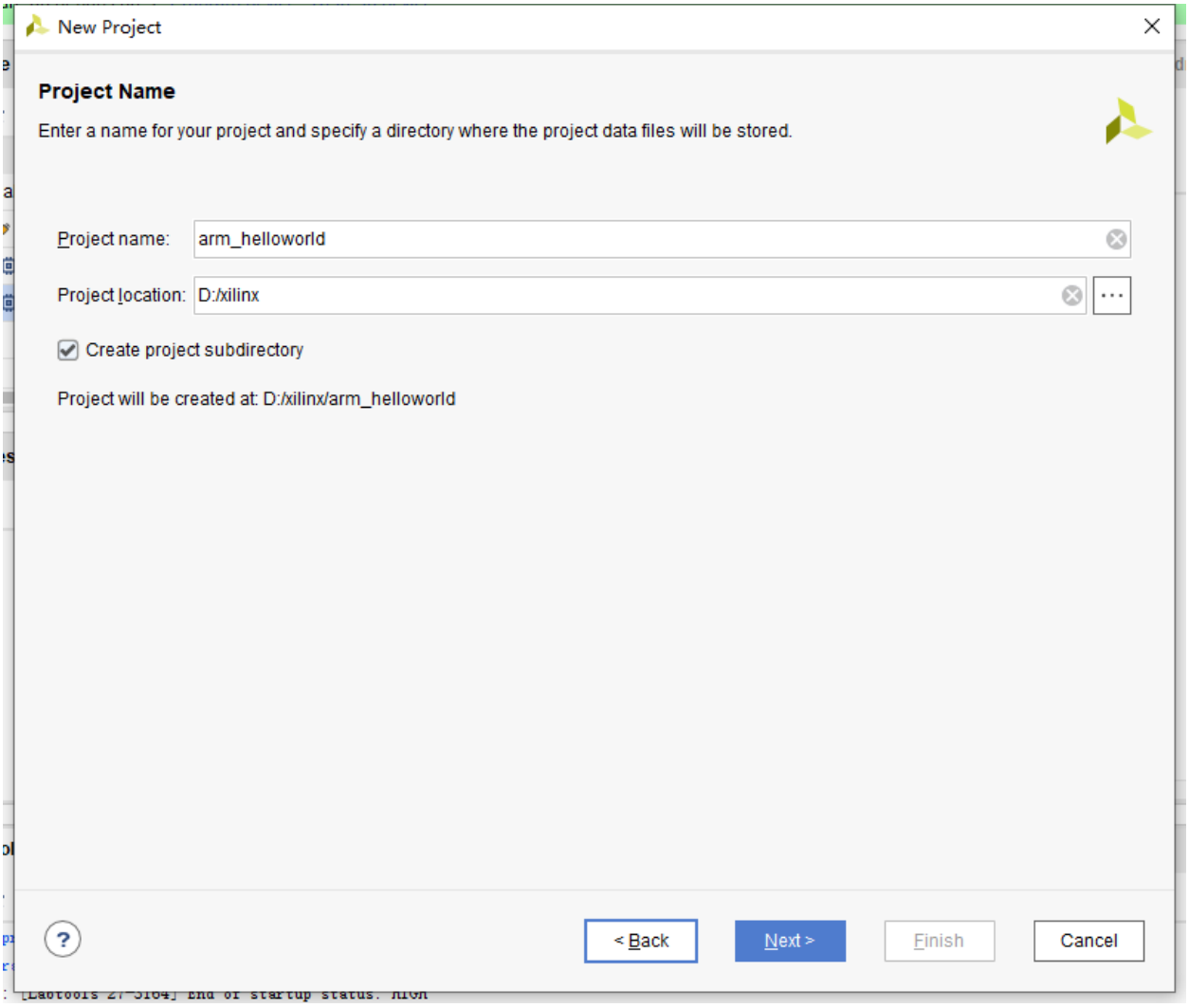
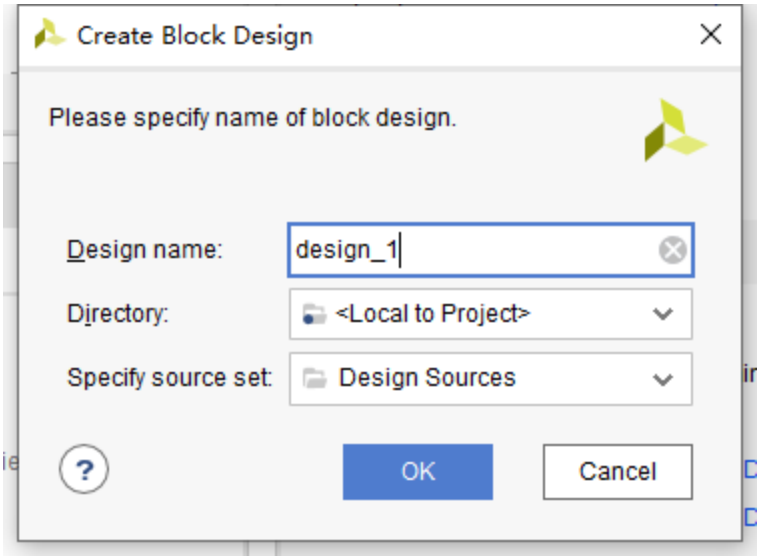
[L07]ARM Hello World

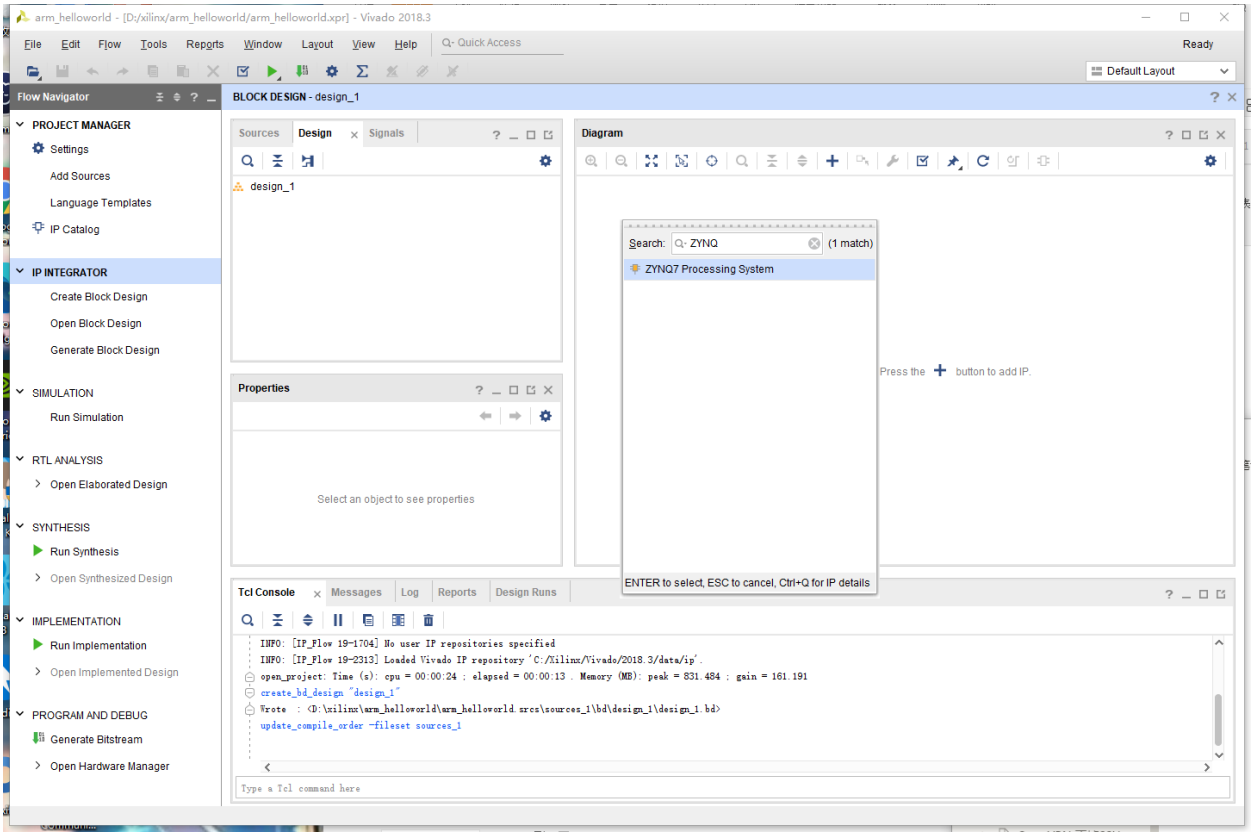
之前说了那么多PL端的,也该说说PS端了,实际上这才是ZYNQ精髓,FPGA其实是这个芯片里打杂的那个,很多教程提到如何Verilog实现串口怎样怎样,费资源不说,还麻烦,不如ARM就printf一下,多双开,现在就开始.不过还是要新建一个FPGA工程.



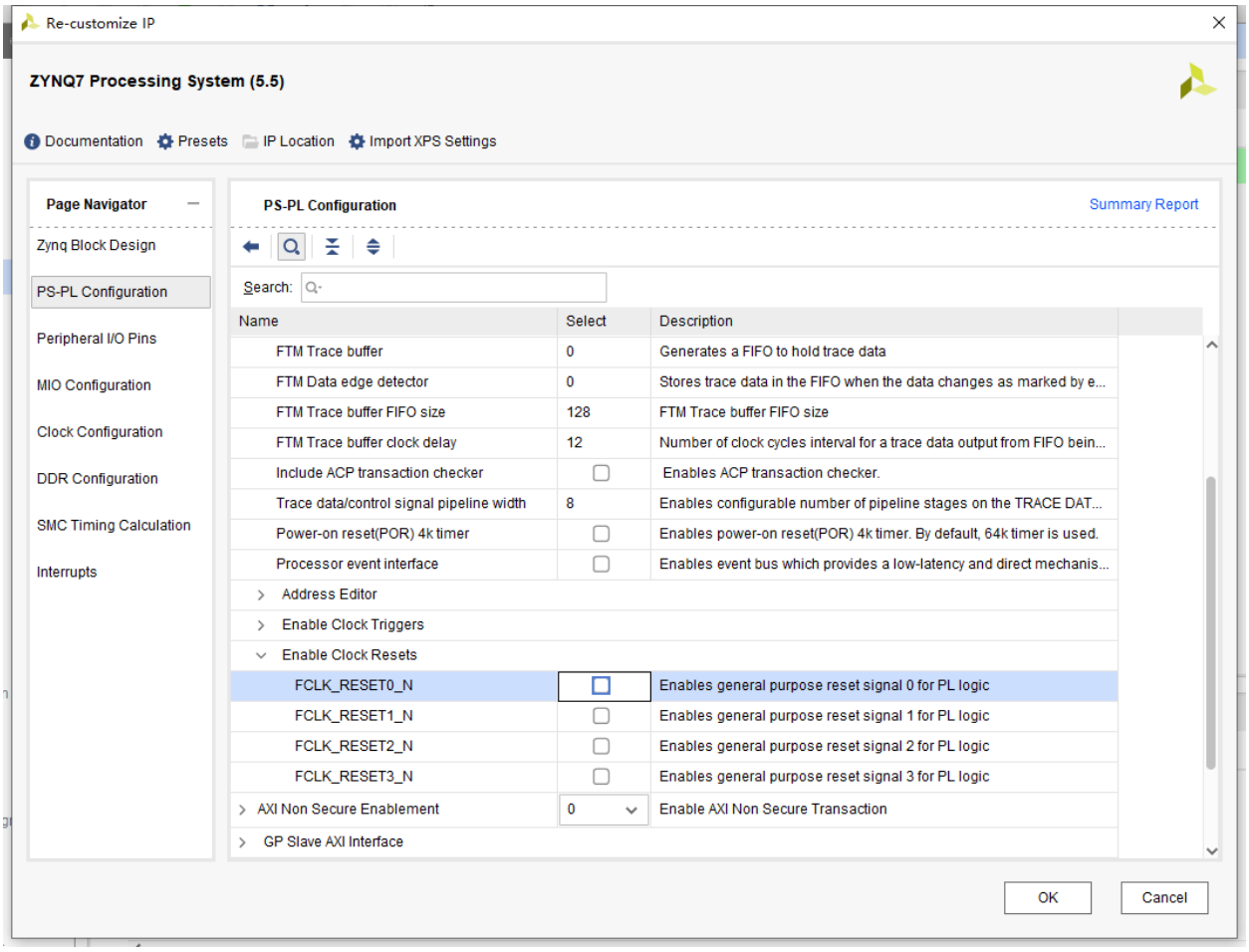
由于这里不写PL逻辑,所以直接Create Block Design,文件名随便,我就保持默认了.



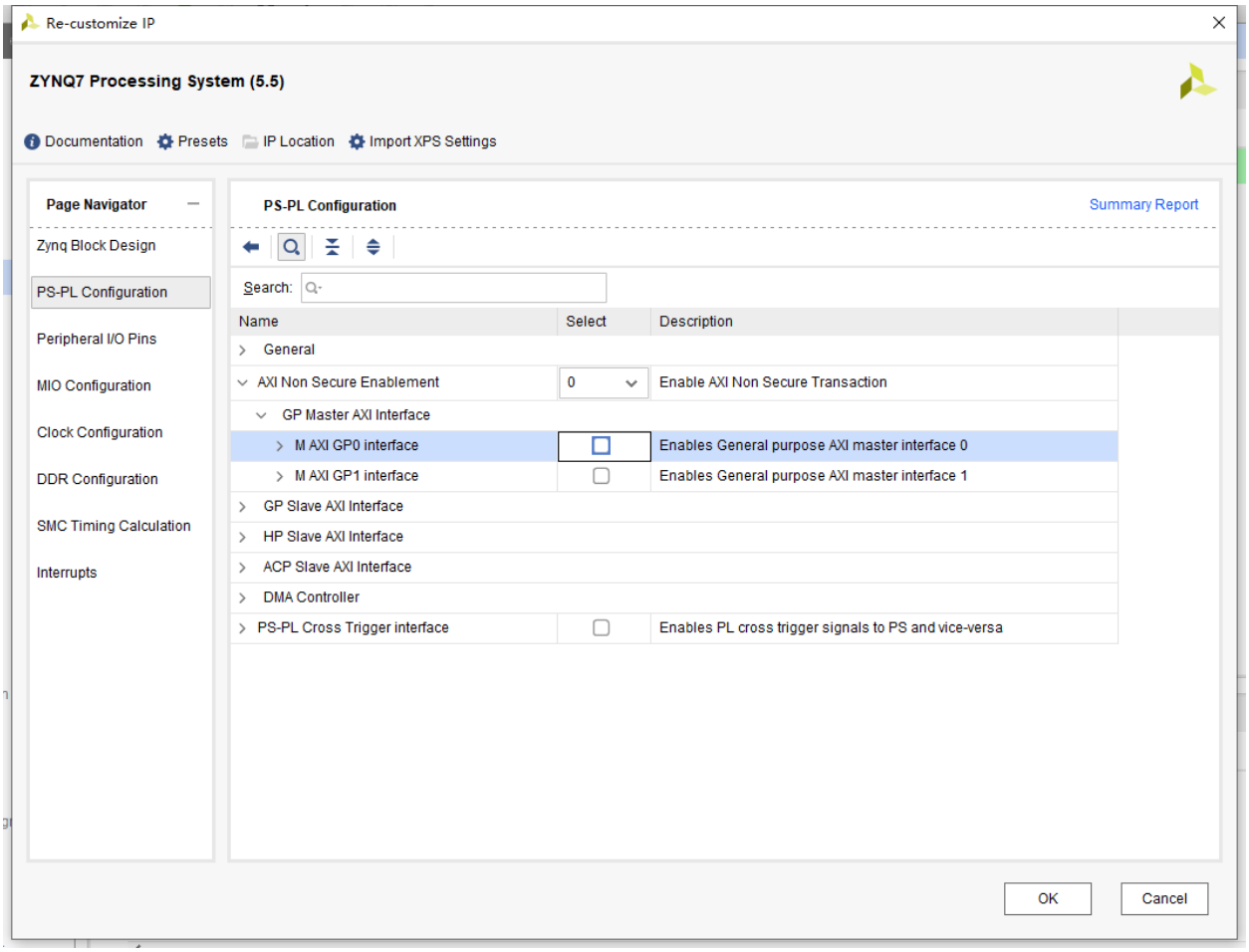
拖一个ZYNQ7 Processing System出来.



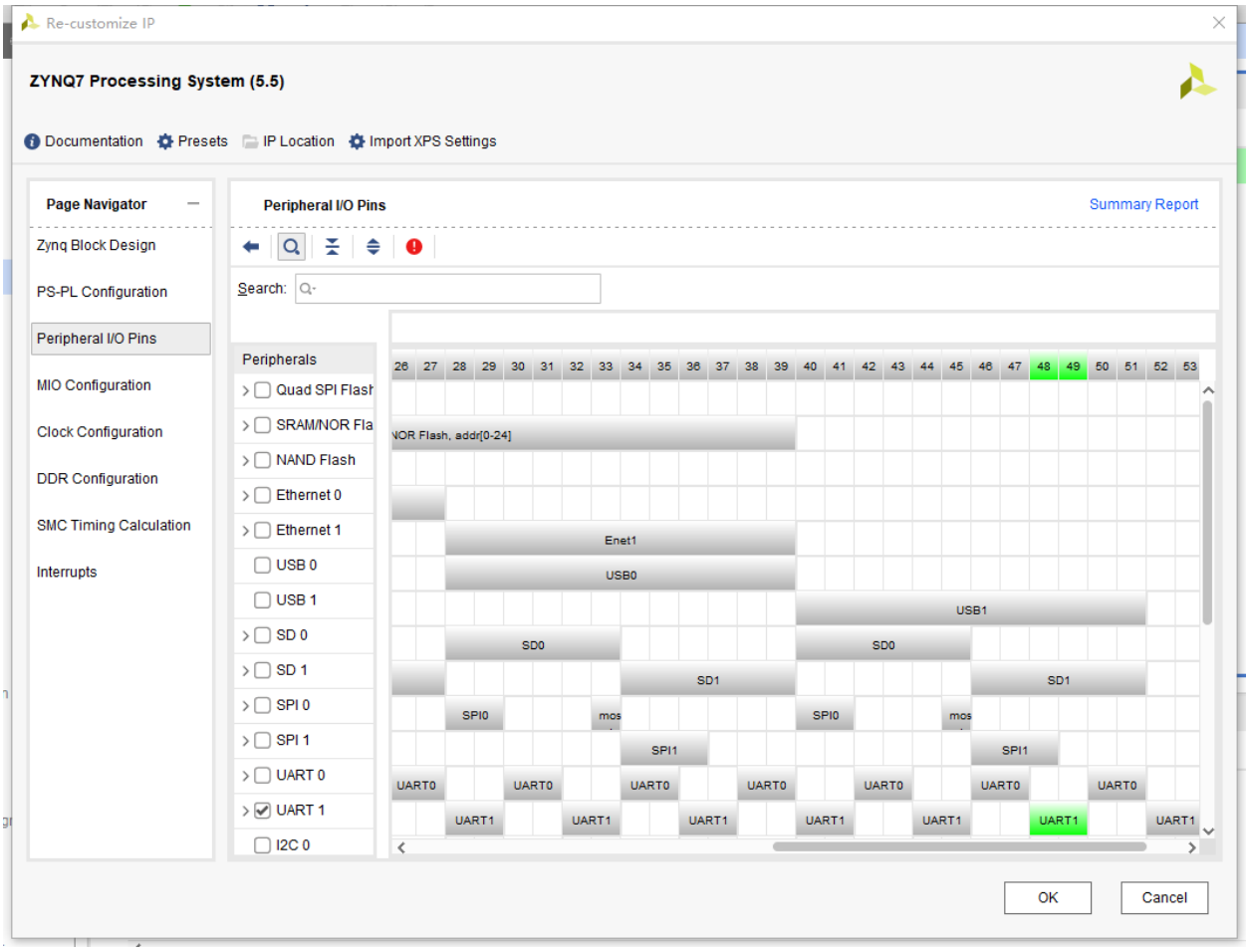
进入这个块的PS-PL Configuration→General→Enable Clock Resets,取消全部复位信号,因为我们暂时用不到PL部分.



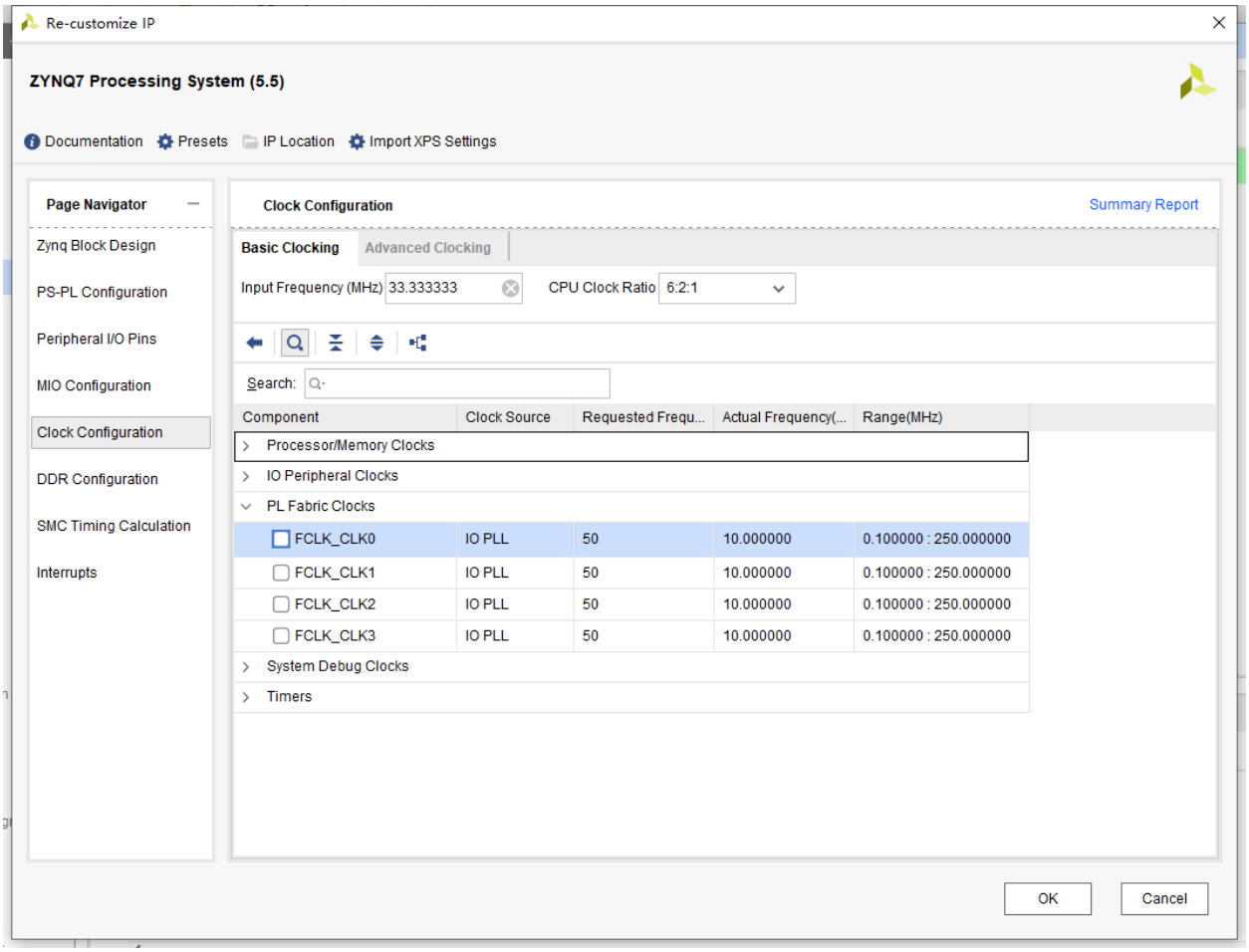
然后在PS-PL Configuration→AXI Non Secure Enablement→GP Master AXI Interface中取消M AXI GP0 Interface,那是因为我们也没链接到其他交错总线.



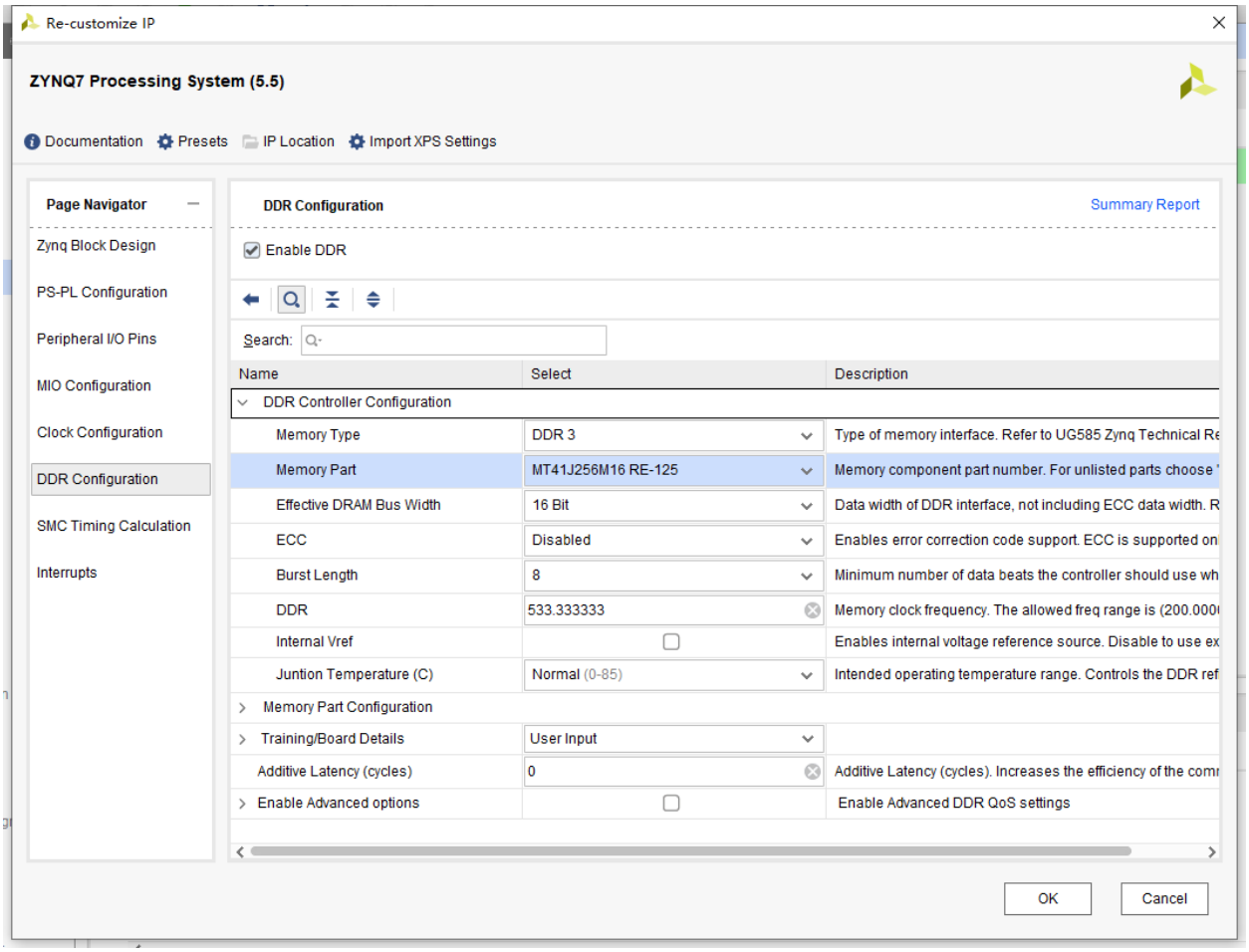
在Peripheral I/O Pins选择UART1并选择相应的IO.



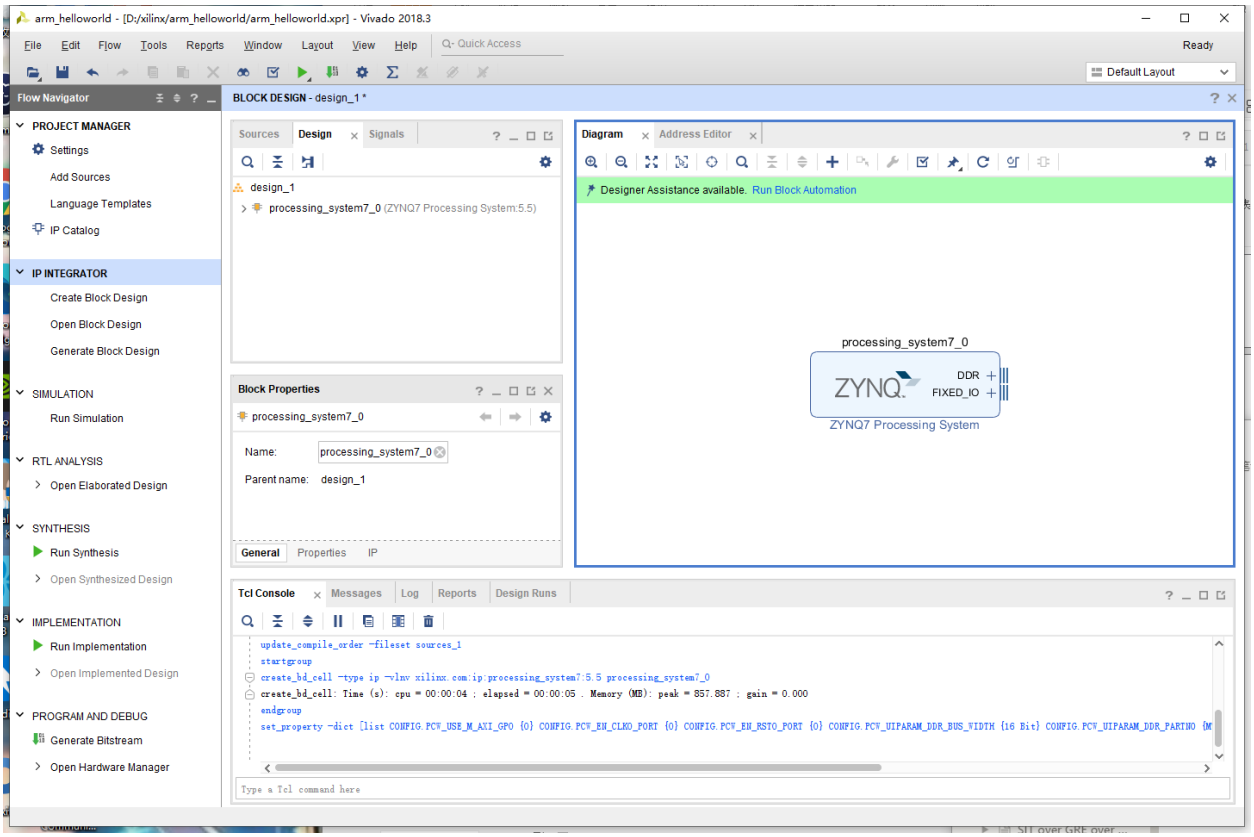
到Clock Configuration→PL Fabric Clocks中取消4个对外时钟,他是由ARM给FPGA提供时钟用的,这里PL都用不上.



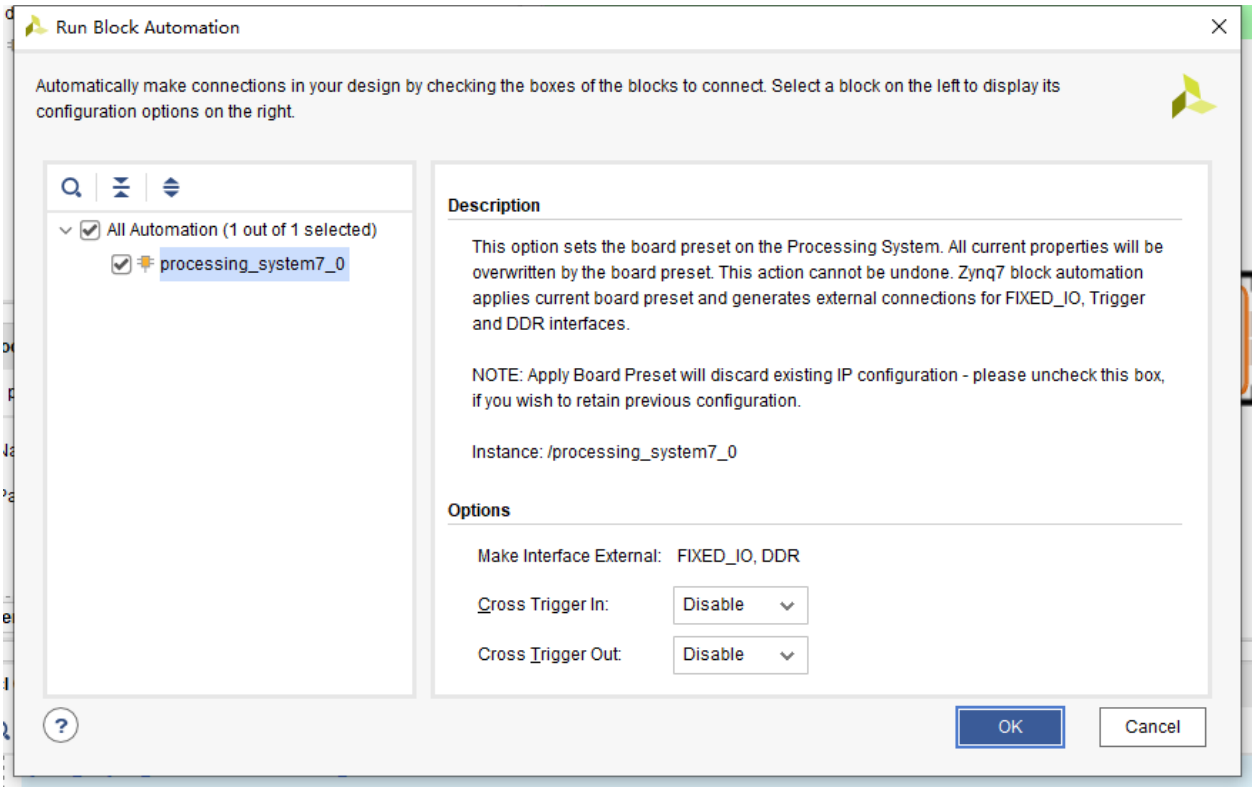
到DDR Configuration配置,这个之前的教程也配置过,不多说了.(倒数第二个就是我手上板子的DRAM)



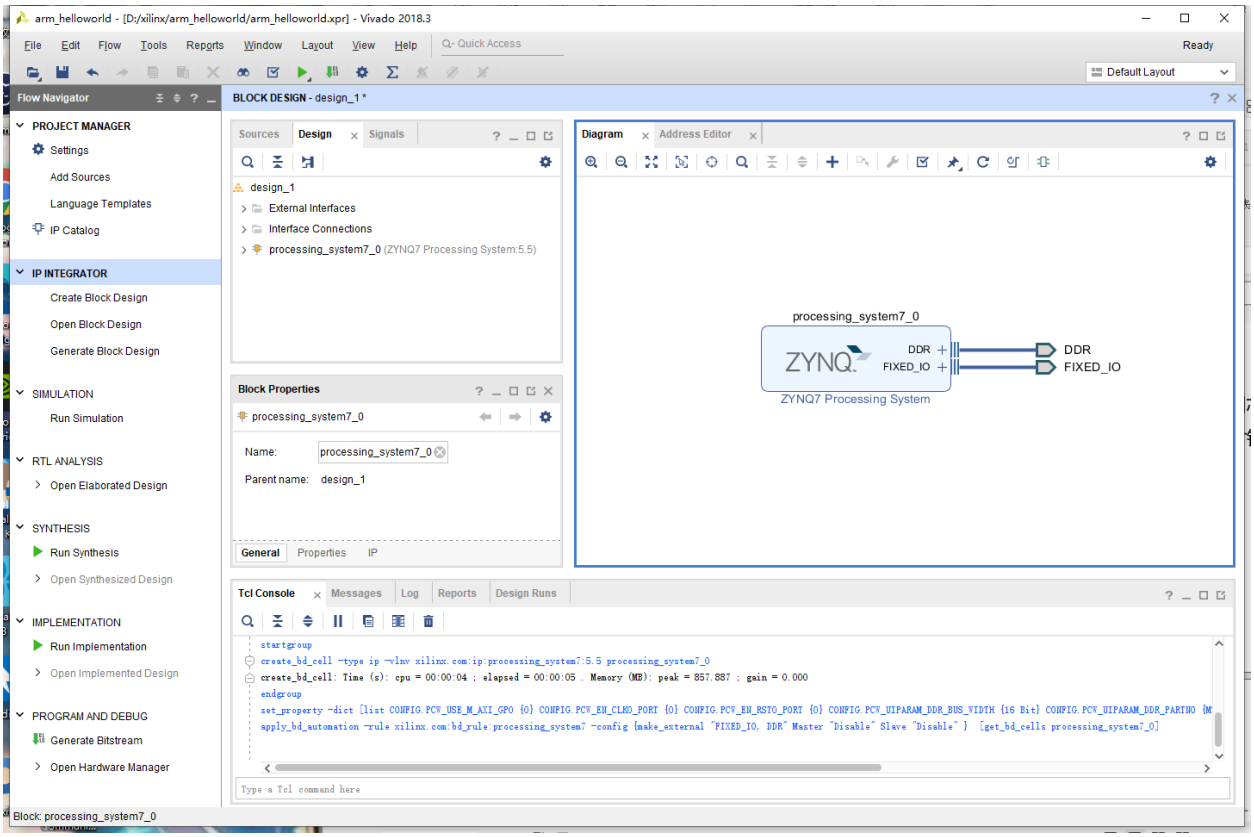
最后点OK就行了,现在提示Run Block Automation.



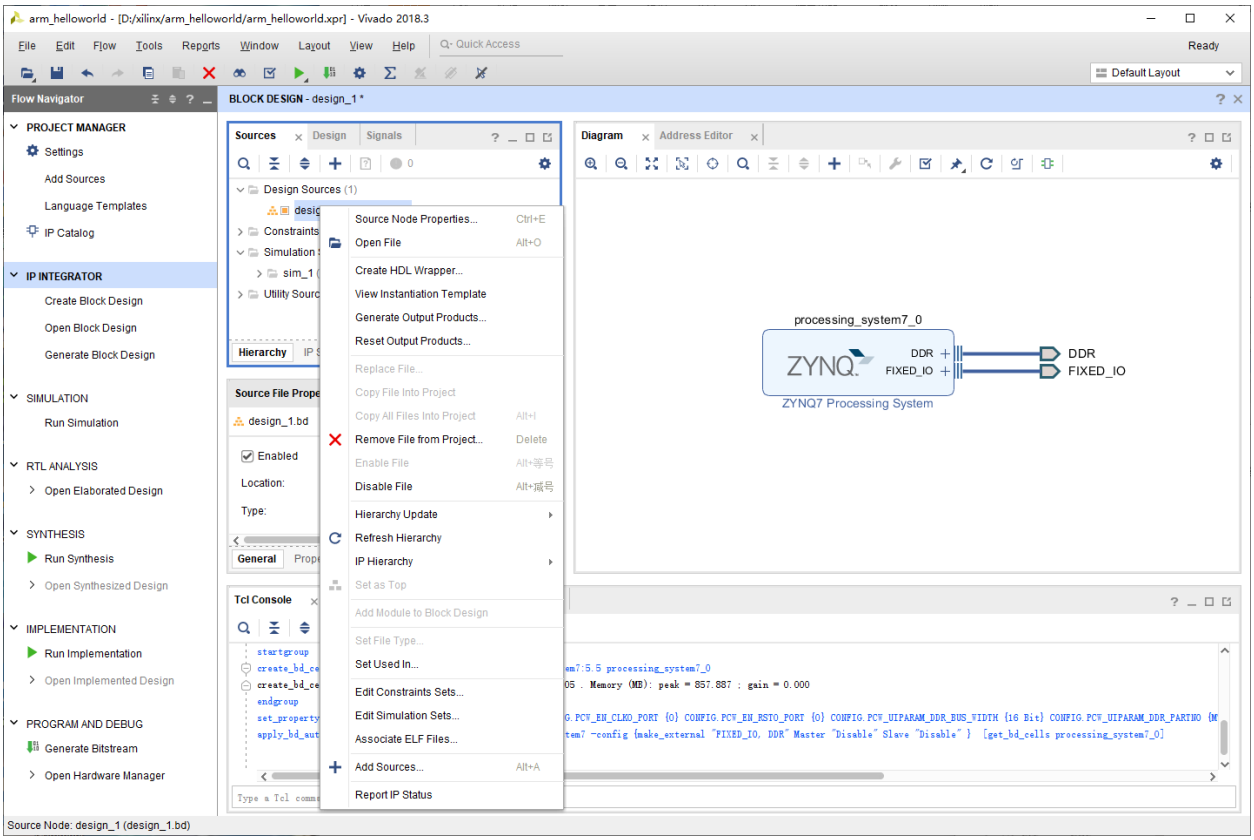
Auto这事情应该人人都喜欢,所以干吧.



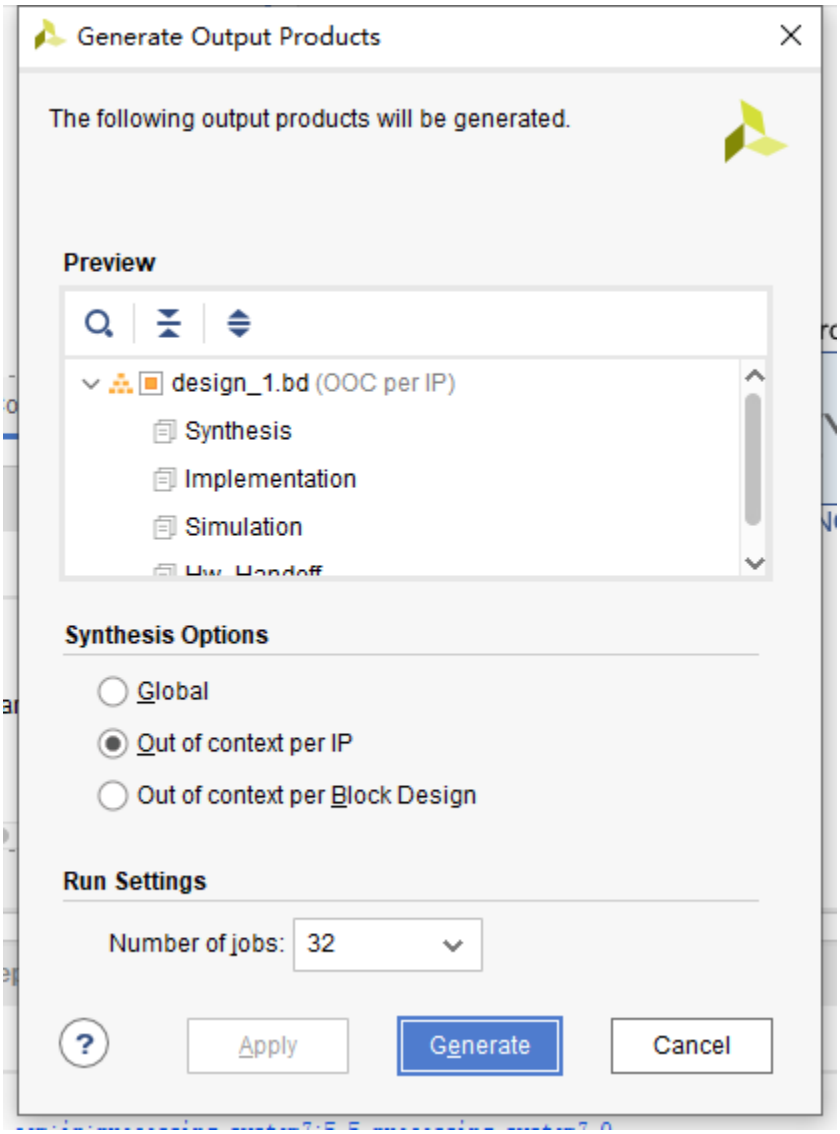
最后DDR信号就出来了.



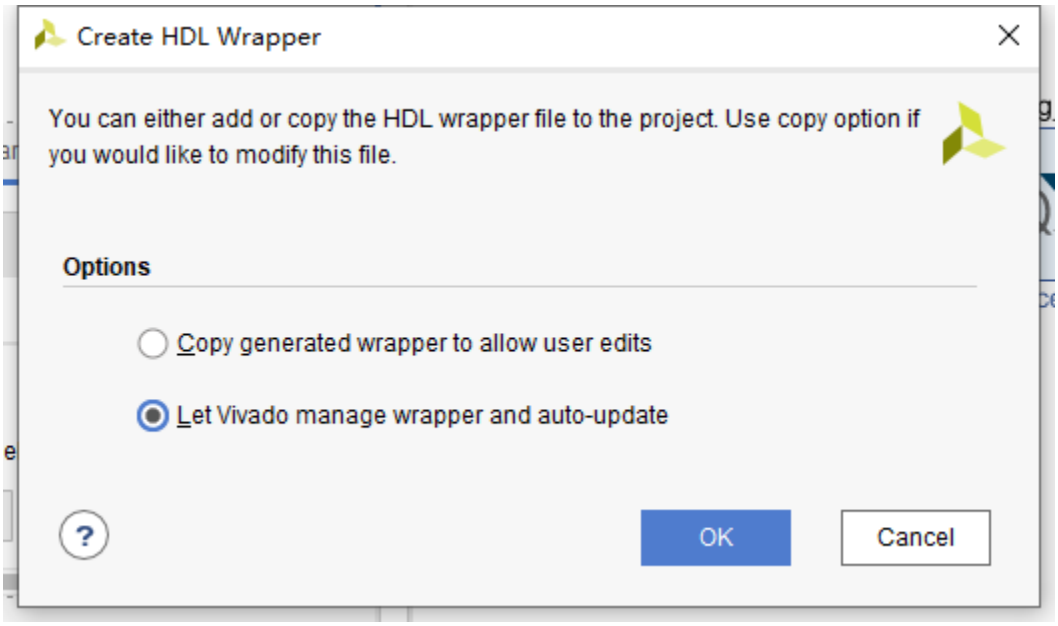
回到源码,右键Generate Outpu Products...


arm_helloworld - [D:/xilinx/arm_helloworld/arm_helloworld.xpr] - Vivado 2018.3
File Edit Flow Tools Reports Window Layout View Help
Quick Access
Ready
Default Layout
Flow Navigator
PROJECT MANAGER
Settings
Add Sources
Language Templates
IP Catalog
IP INTEGRATOR
Create Block Design
Open Block Design
Generate Block Design
SIMULATION
Run Simulation
RTL ANALYSIS
Open Elaborated Design
SYNTHESIS
Run Synthesis
Open Synthesized Design
IMPLEMENTATION
Run Implementation
Open Implemented Design
PROGRAM AND DEBUG
Generate Bitstream
Open Hardware Manager
BLOCK DESIGN - design_1 *
Sources
Design
Signals
Design Sources (1)
Constraints
Simulation
Utility Sources
Hierarchy
Source File Properties
design_1.bd
Enabled
Location:
Type:
General
Source Node Properties... Ctrl+E
Open File Alt+O
Create HDL Wrapper...
View Instantiation Template
Generate Output Products...
Reset Output Products...
Replace File...
Copy File Into Project
Copy All Files Into Project Alt+I
Remove File from Project... Delete
Enable File Alt+等号
Disable File Alt+减号
Hierarchy Update
Refresh Hierarchy
IP Hierarchy
Set as Top
Add Module to Block Design
Set File Type...
Set Used In...
Edit Constraints Sets...
Edit Simulation Sets...
Associate ELF Files...
Add Sources... Alt+A
Report IP Status
Diagram
Address Editor
processing_system7_0
ZYNQ
DDR
FIXED_IO
ZYNQ7 Processing System
Tcl Console
Source Node: design_1 (design_1.bd)

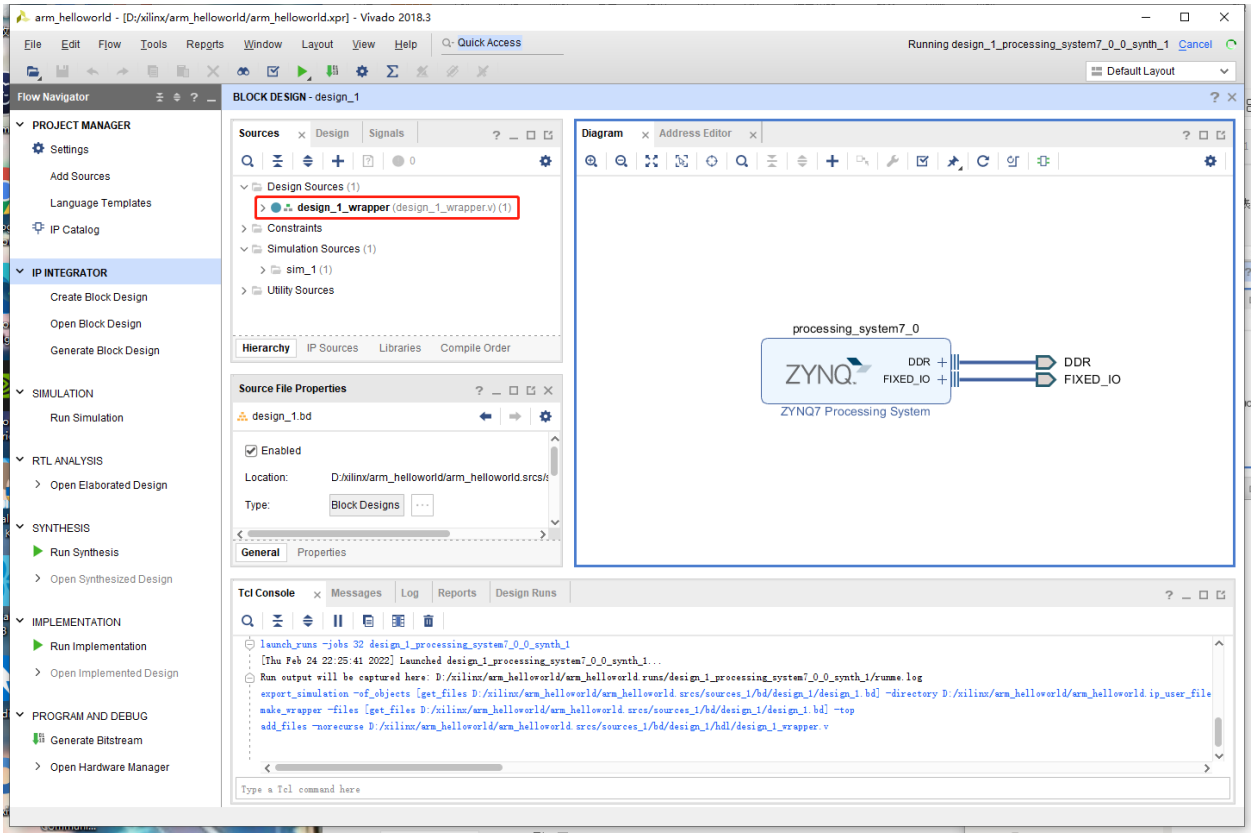
默认配置就可以.



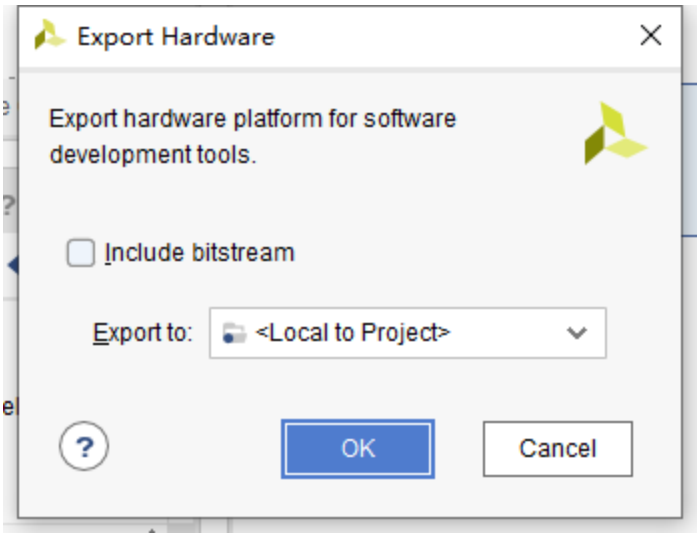
然后继续右键Create HDL Wrapper.


Create HDL Wrapper
You can either add or copy the HDL wrapper file to the project. Use copy option if you would like to modify this file.
Options
Copy generated wrapper to allow user edits
Let Vivado manage wrapper and auto-update
OK
Cancel

最后顶层文件就出来了.

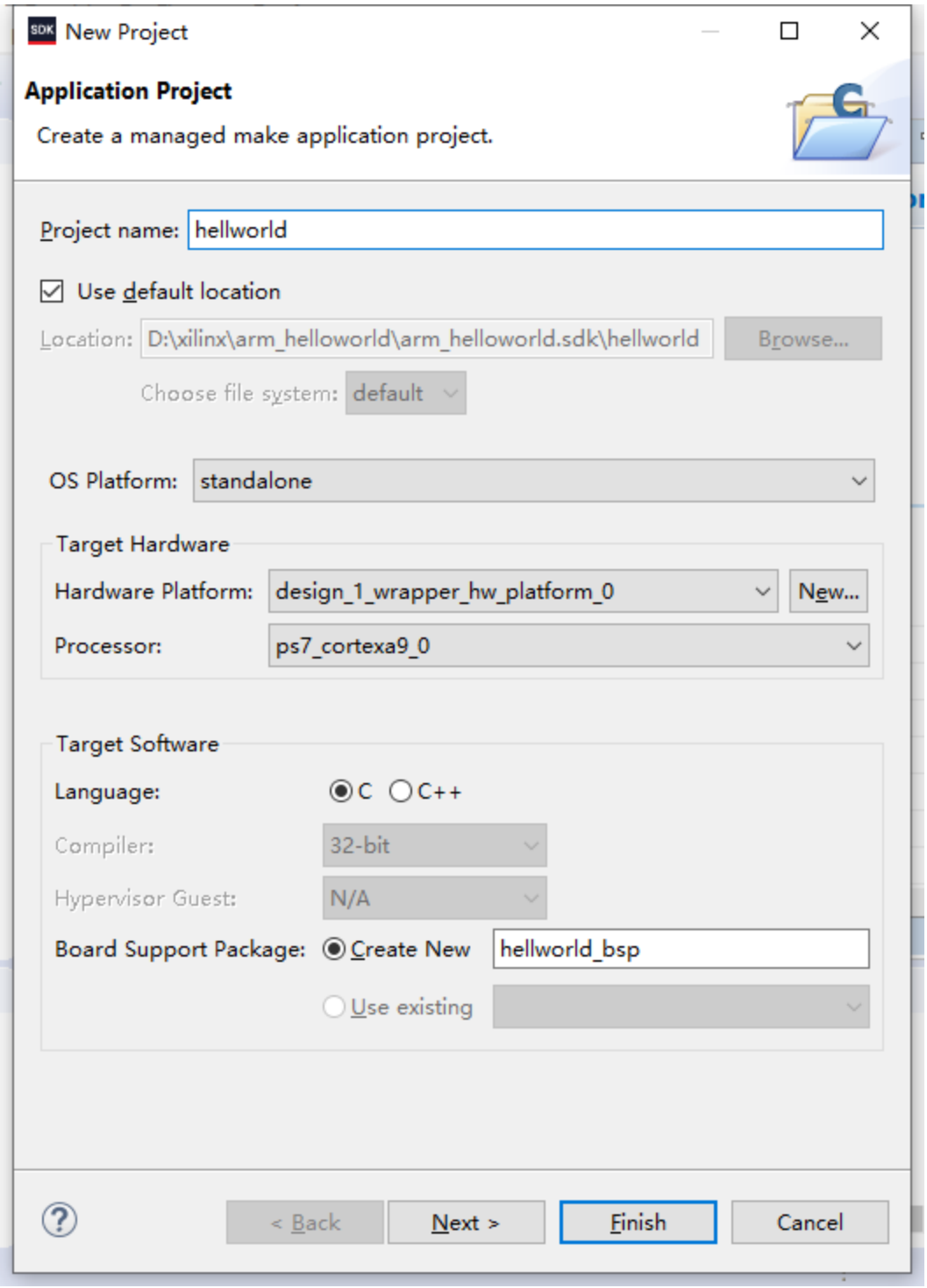

arm_helloworld - [D:/xilinx/arm_helloworld/arm_helloworld.xpr] - Vivado 2018.3
File Edit Flow Tools Reports Window Layout View Help
Quick Access
Running design_1_processing_system7_0_0_synth_1 Cancel
Default Layout
Flow Navigator
PROJECT MANAGER
Settings
Add Sources
Language Templates
IP Catalog
IP INTEGRATOR
Create Block Design
Open Block Design
Generate Block Design
SIMULATION
Run Simulation
RTL ANALYSIS
Open Elaborated Design
SYNTHESIS
Run Synthesis
Open Synthesized Design
IMPLEMENTATION
Run Implementation
Open Implemented Design
PROGRAM AND DEBUG
Generate Bitstream
Open Hardware Manager
BLOCK DESIGN - design_1
Sources
Design
Signals
Design Sources (1)
design_1_wrapper (design_1_wrapper.v) (1)
Constraints
Simulation Sources (1)
sim_1 (1)
Utility Sources
Hierarchy
IP Sources
Libraries
Compile Order
Source File Properties
design_1.bd
Enabled
Location:
Type:
Block Designs
General
Properties
Diagram
Address Editor
processing_system7_0
ZYNQ
DDR
FIXED_IO
ZYNQ7 Processing System
Tcl Console
Messages
Log
Reports
Design Runs
launch_runs -jobs 32 design_1_processing_system7_0_0_synth_1
[Thu Feb 24 22:25:41 2022] Launched design_1_processing_system7_0_0_synth_1...
Run output will be captured here: D:/xilinx/arm_helloworld/arm_helloworld.runs/design_1_processing_system7_0_0_synth_1/runme.log
export_simulation -of_objects [get_files D:/xilinx/arm_helloworld/arm_helloworld.srcs/sources_1/bd/design_1/design_1.bd] -directory D:/xilinx/arm_helloworld/arm_helloworld.ip_user_file
make_wrapper -files [get_files D:/xilinx/arm_helloworld/arm_helloworld.srcs/sources_1/bd/design_1/design_1.bd] -top
add_files -norecurse D:/xilinx/arm_helloworld/arm_helloworld.srcs/sources_1/bd/design_1/hdl/design_1_wrapper.v
Type a Tcl command here

为了确保一切没问题,简单综合一下,确定没有错误了,选择File→Export→Export Hardware,由于没有PL设计,所以可以无需包含bitstream.

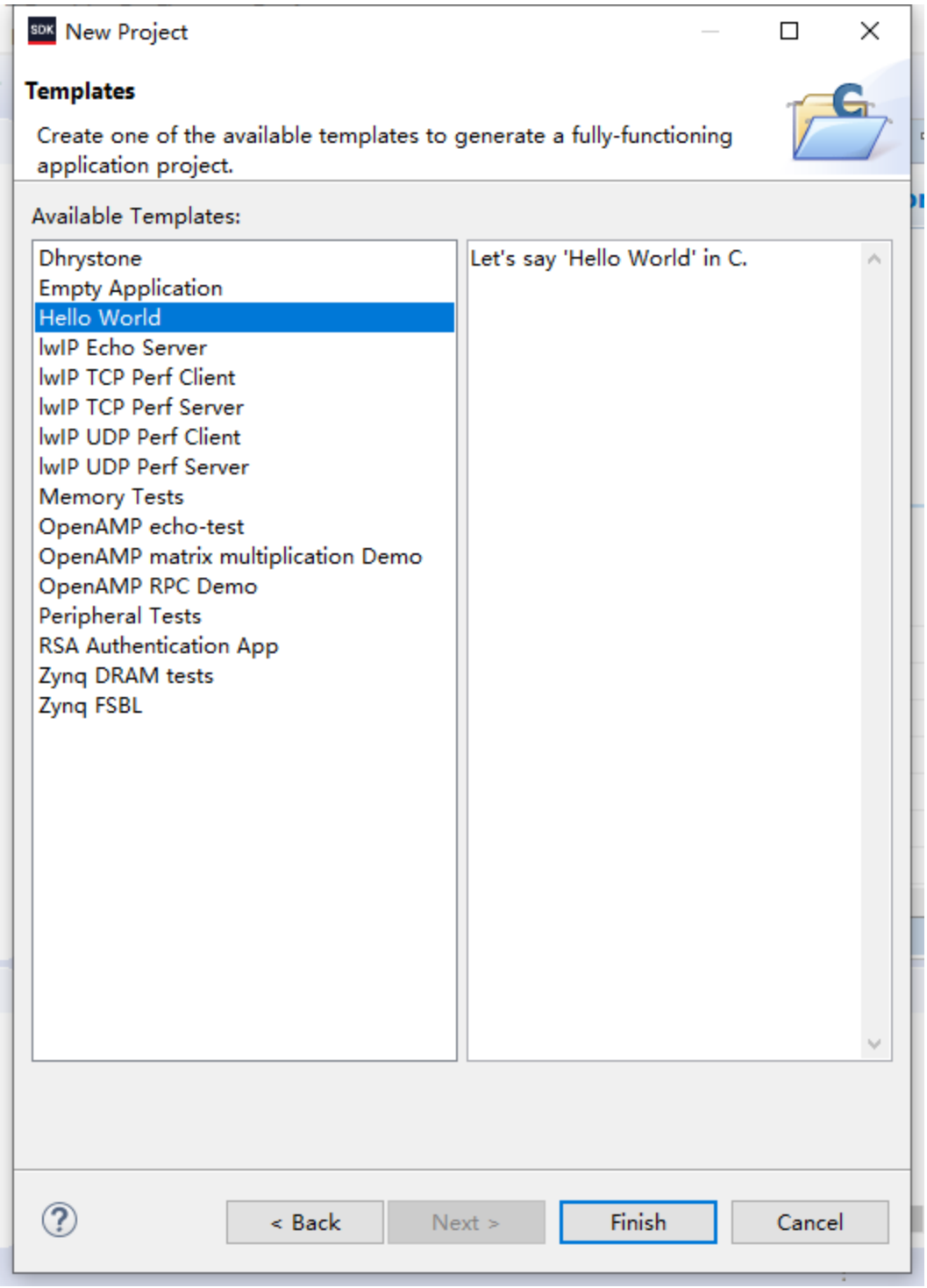


接着启动SDK(File→Launch SDK).

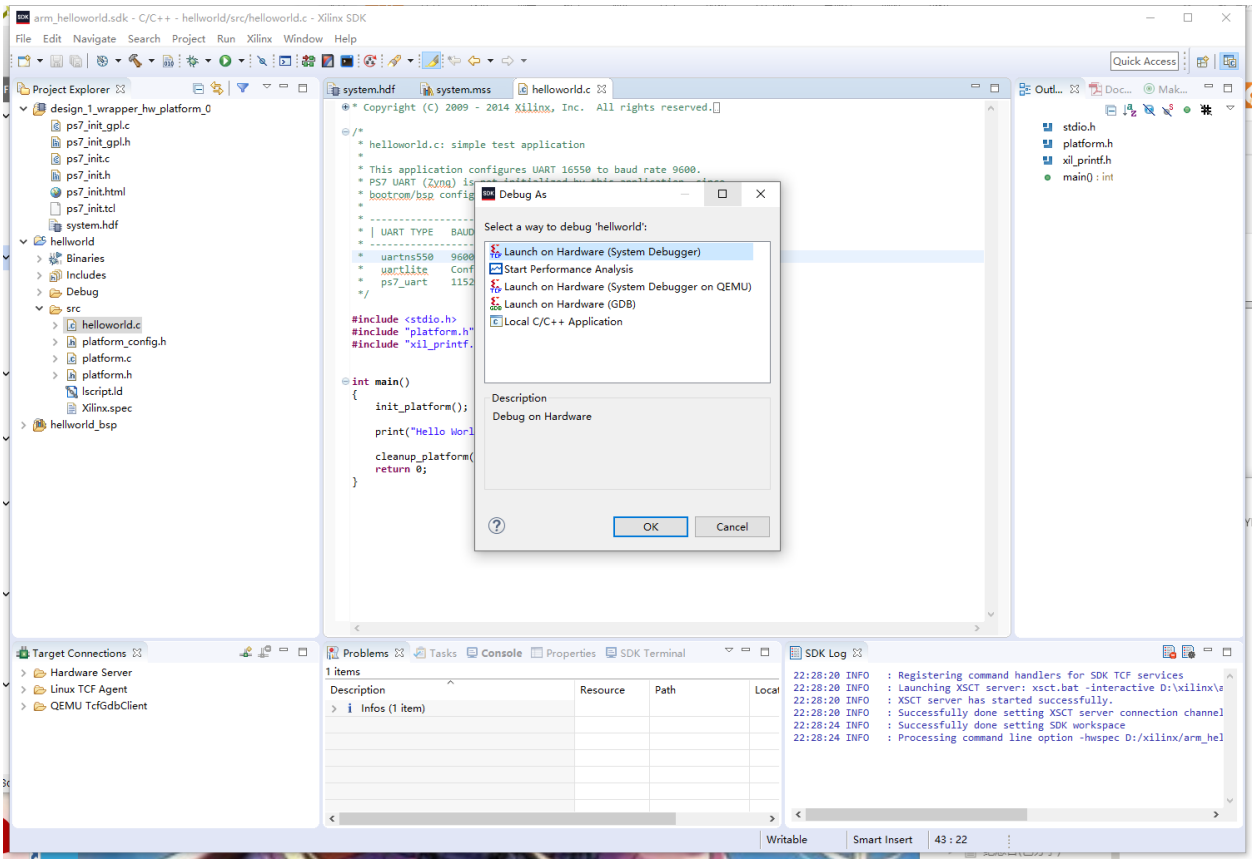
记得上次固化工程时候是怎样的,新建一个fsbl,这次不了,这次主要就是为了ARM Hello World,所以新建一个工程,名字暂且叫helloworld吧.



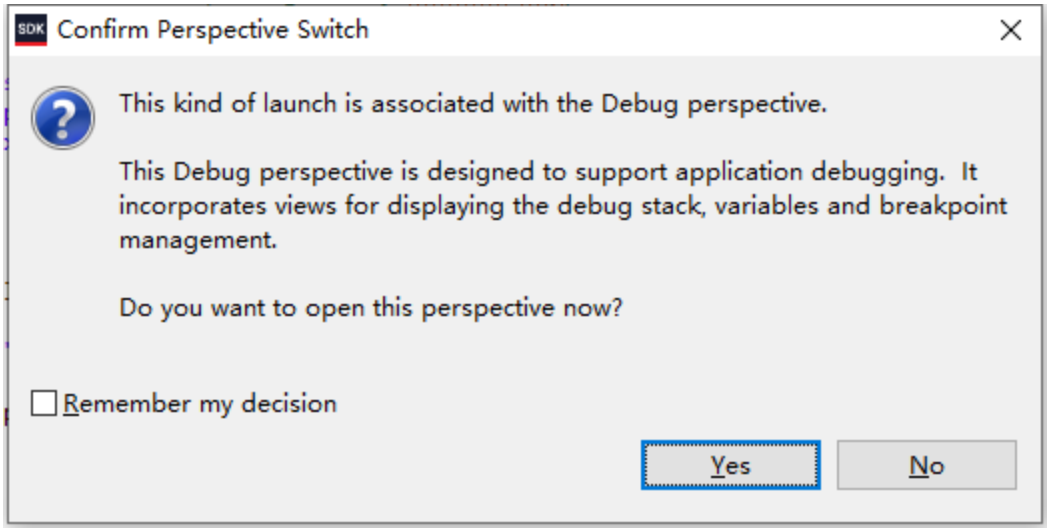
Next后就知道我要新建一个Hello World了,挺厉害的吧.


New Project
Templates
Create one of the available templates to generate a fully-functioning application project.
Available Templates:
Dhrystone
Empty Application
Hello World
lwIP Echo Server
lwIP TCP Perf Client
lwIP TCP Perf Server
lwIP UDP Perf Client
lwIP UDP Perf Server
Memory Tests
OpenAMP echo-test
OpenAMP matrix multiplication Demo
OpenAMP RPC Demo
Peripheral Tests
RSA Authentication App
Zynq DRAM tests
Zynq FSBL
Let's say 'Hello World' in C.
< Back
Next >
Finish
Cancel

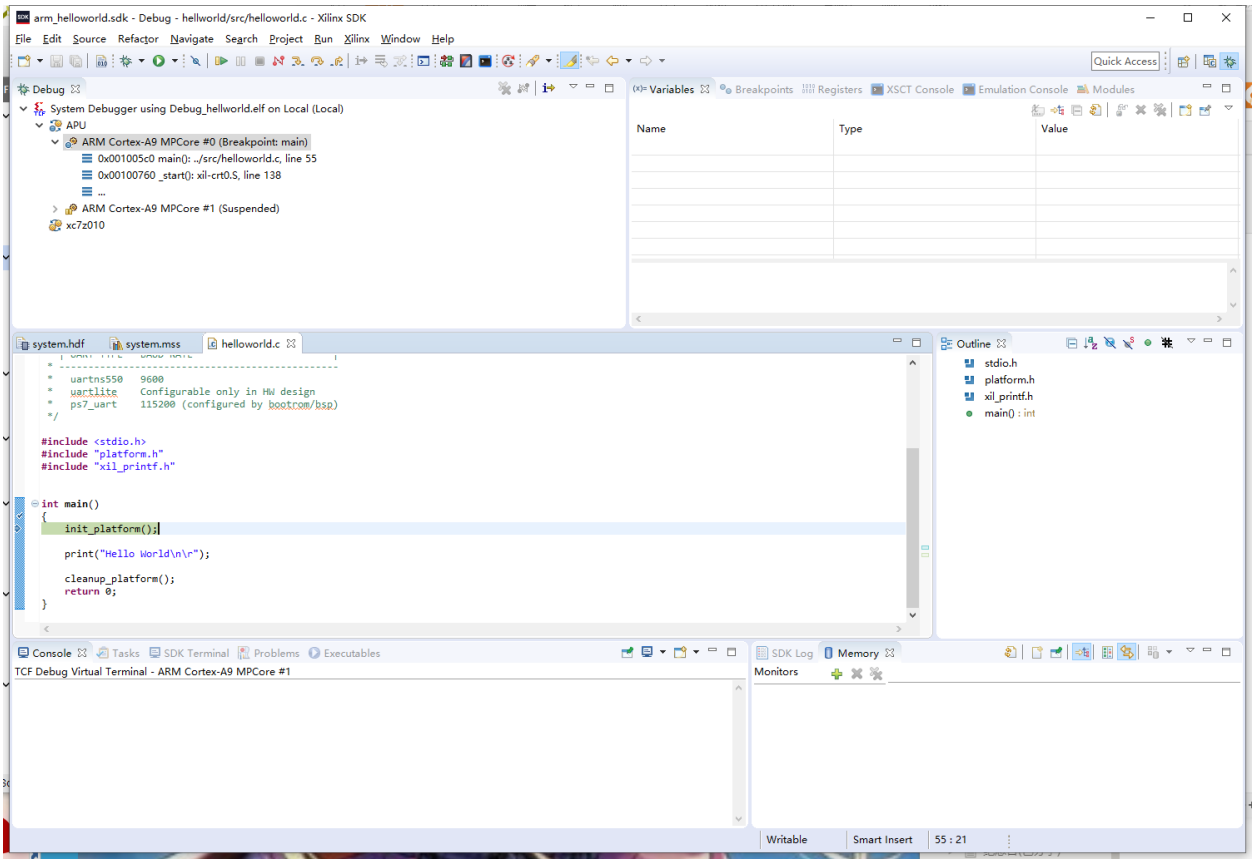
编译然后开始调试.



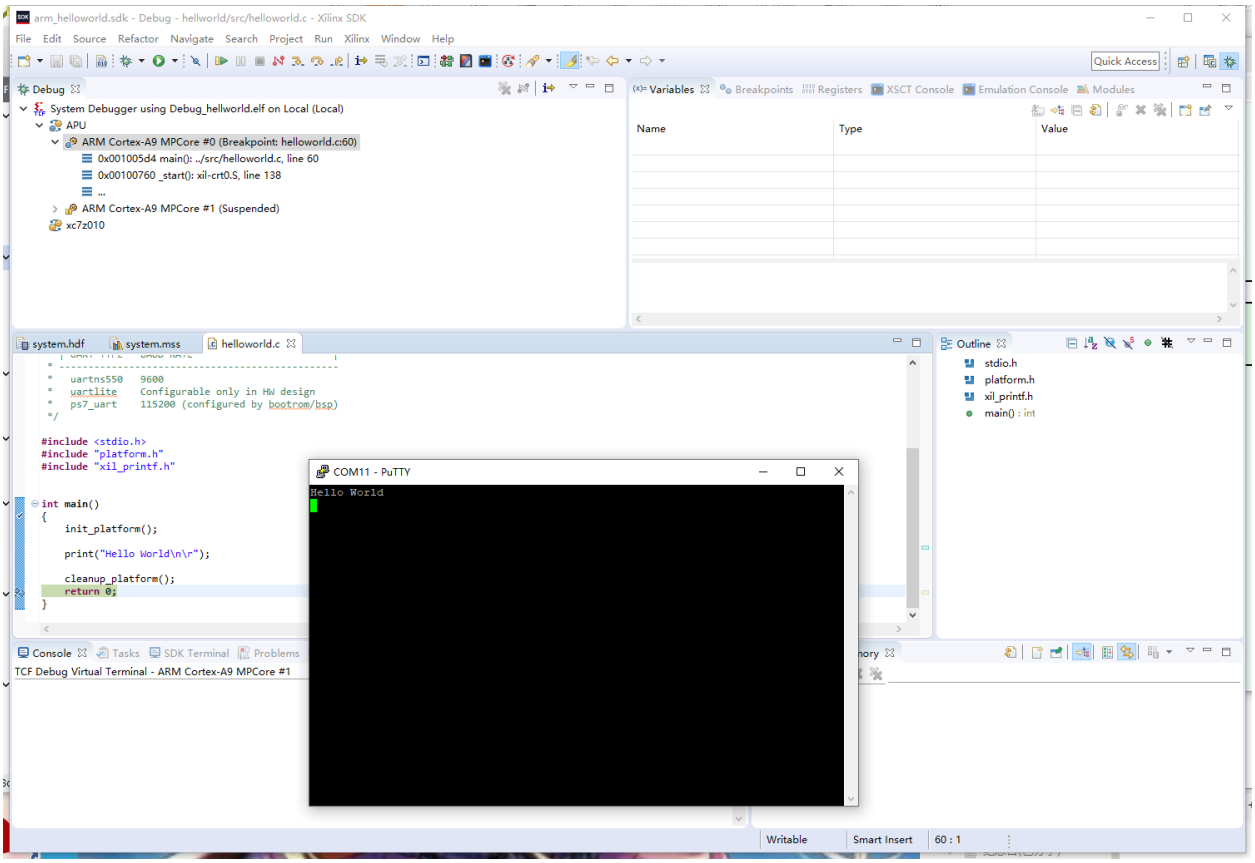
弹出的提示也Yes,然后就进入调试界面了.



可以看到第一个A9正在跑呢.



拿起手里熟悉的串口工具来试试吧.



写在最后,为什么没有用PL实现串口,因为PL实现串口不但麻烦,不环保,而且为什么不发挥ARM C编程优势呢?到这里了,应该很熟悉主要操作了,后续的教程里必然会省略很多这些步骤,如果不熟悉就不断训练吧.