

# hls+ vivado + vitis sdk -zcu104 为例

--louwenqi

本教程旨在记录从 hls 到硬件快速部署的流程，其中包含了简单的步骤解释，适合有相关基础的同学。由于目的是快速验证，因此最后的硬件部署并不基于 linux 系统，而是基于裸机 +SD card 的形式进行快速硬件验证。若想从更深入了解，可以参考 Xuan Wang 写的[教程](#)。后续会补充 hls+petalinux 或者 Pynq 的教程。

软件版本为 vitis2020.2，Vivado2019.2 也适用。

## 目录

hls+ vivado + vitis sdk -zcu104 为例 .....	1
一、 HLS 部分 .....	3
1. C Simulation .....	3
2. C Synthesis .....	3
3. Co-Simulation .....	4
4. Export RTL .....	4
二、 Vivado 部分 .....	4
1. 创建工程 .....	4
2. 创建 Block Design .....	5
2.1 添加 HLS IP 核心 .....	5
2.2 添加 PS 及配置 .....	6
2.3 添加 AXI 桥 .....	8
2.4 Address Assign & Valide Design .....	9
3. 综合以及实现 .....	10
4. 导出硬件 .....	10
三、 Vitis 部分 .....	10
1. 创建 platform 工程 .....	11
2. 创建 application 工程 .....	11
3. 编写测试程序 .....	12
4. Program FPGA .....	14

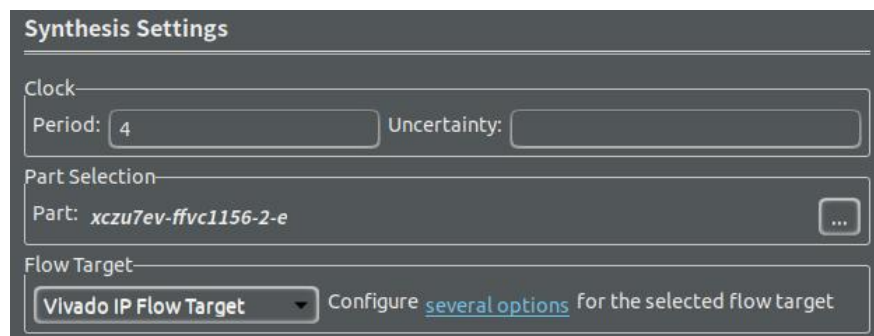
## 一、HLS 部分

编写 hls 源文件以及 tb 测试文件。HLS 的使用可以查看 [Xilinx 官方文档](#)

### 1. C Simulation

该阶段仅仅验证 C++ 功能,不涉及硬件

### 2. C Synthesis



注意这里在 solution 设置时,选择“**Vivado IP Flow Target**”,板子可以直接选官方的 zcu104 板子。

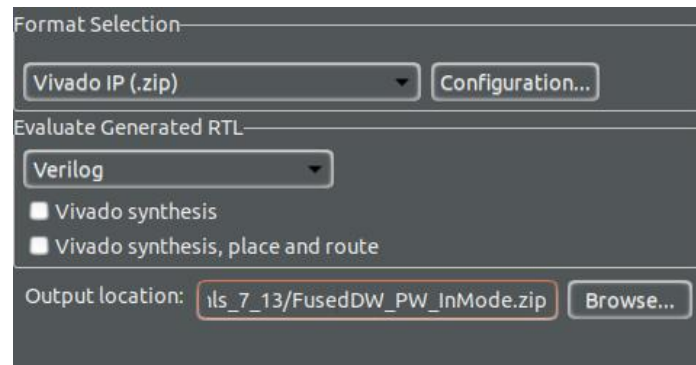
Hls 综合 :可以给出**估计**的 latency 以及 resource 数据 ,该数据一定程度上考虑硬件映射 ,但本质上为 Xilinx 自己对 C 语言翻译成的 RTL 的性能以及资源消耗估计。(资源估计往往过多 ,延迟估计则和代码编写有关 ,若编程者较少考虑硬件 ,则该数值和实际运行差距巨大)

该过程会考虑外部接口的影响。例如 ,若顶层为 AXI4 接口 ,burst、outstanding、latency 会影响资源和性能。Function 内部以及之间的调度可以通过 Analysis ->shedule Viewer 查看

### 3. Co-Simulation

类似于 Vivado 的 RTL 行为级别仿真，该过程相对更接近真实情况，但是不考虑外部 DDR 访存效率。Cosim 中选择 all 则可以在 vivado 中查看波形。

### 4. Export RTL



注意这里同样保持“Vivado IP”，如果不需要提前知道该 IP 核心消耗的资源，这里不用勾选“Vivado synthesis”以及“Vivado synthesis, place and route”以节省时间。

Vitis\_hls 2020.2 在该过程中存在 bug，需将系统时钟调整至 2022 年之前，否则导出失败。成功导出则会有类似“xxx.zip”，该文件后续会在 vivado 中使用到。

另外，注意。导出成功后，solution -> impl-> ip->drivers 文件夹中包含好了裸机驱动函数，可以省去在编写测试文件时对地址的显式访问

## 二、Vivado 部分

### 1. 创建工程

这里需要与 hls 中的选择保持一致，在下方板子中选择 zcu104.

### Default Part

Choose a default Xilinx part or board for your project.



Parts | **Boards**

[Reset All Filters](#)

Install/Update Boards

Vendor: 

All

Name: 


All

Board Rev: 

Latest


Search: 

Q

Display Name	Preview	Vendor	File Version	Part
Alpha-Data ADM-PCI-E-7V3		alpha-data.com	1.1	xc7vx690tffg1157-2

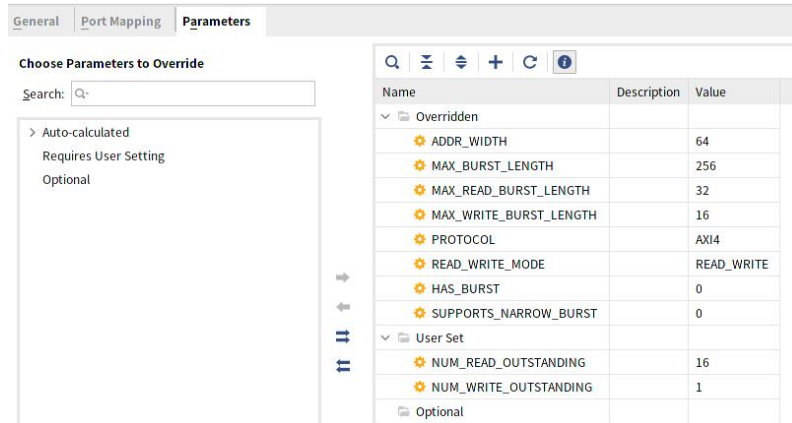
## 2. 创建 Block Design

在 vivado 左侧导航栏，选择” create Block Design “，命名之后就出现了空白的设计，点击” 加号 “即可添加 IP。在这之前首先在 IP 库里面添加 HLS 中自定义的 IP 核心。

This design is empty. Press the  button to add IP.

### 2.1 添加 HLS IP 核心

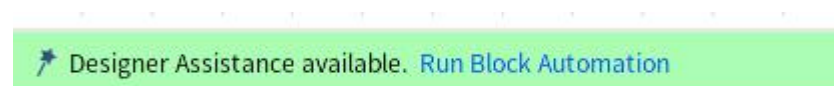
左侧 settings-> IP Catalog，右键” add Repository “将到处的 IP 包所在的路径添加进 User Repository。 详细信息可以左键该 IP，选择”edit the IP packager”查看。其中 Edit Interface 可以看到 hls 里面顶层 interface 的配置。默认的 outstanding 为 32,不使用的可以设置为 1,以减少资源消耗。



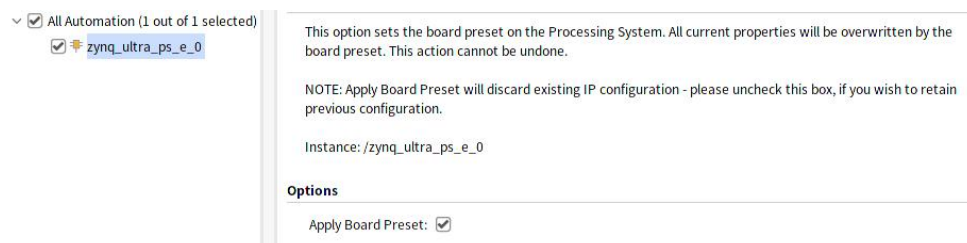
## 2.2 添加 PS 及配置

首先添加 zynqMP ( Zynq UltraScale+ MPSoC ), 即配置 ARM 中的一些参数.

添加该 IP 后, 会出现“**Run Block Automation**”的提示。



点击进去, 可以发现, 该操作对板子进行了“Preset”操作

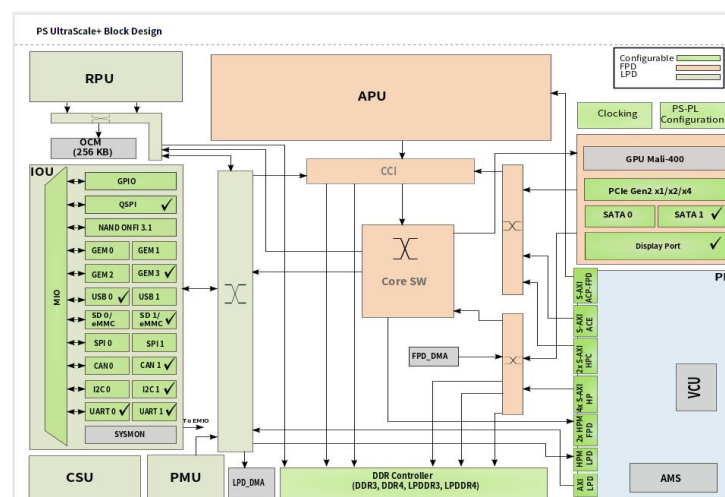


确认过后, ZynqMP 的配置就符合了 ZCU104 的板子了。

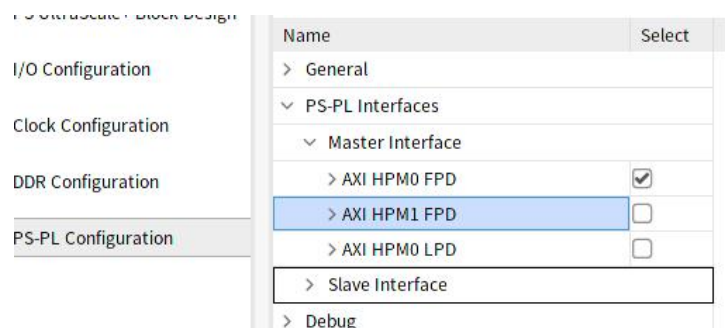


这里双击该 IP 核心，进行具体配置。(这里如果是官方板子，不建议对 **QSPI**，**SATA**、**I2C** 等进行操作，很容易导致后续测试 **failed**)。如果仅仅进行 SD 卡读写测试，则这里进行玩 Automation 即可。

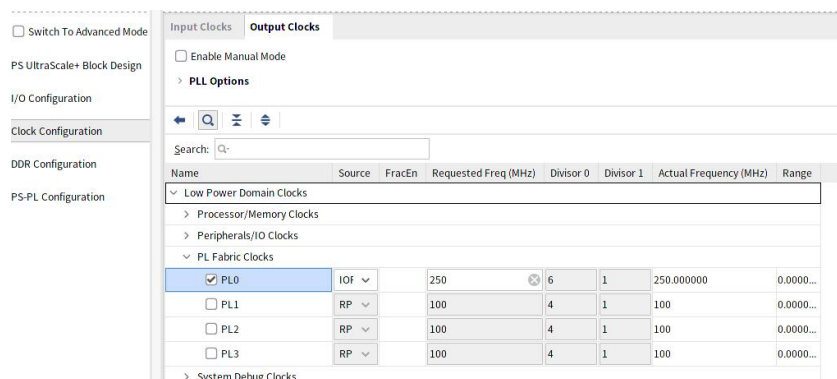
我们这里展示怎么配置，AXI 接口与时钟。



例如，我这里只需要一个 PS Master 接口，则将 HPM1 接口取消

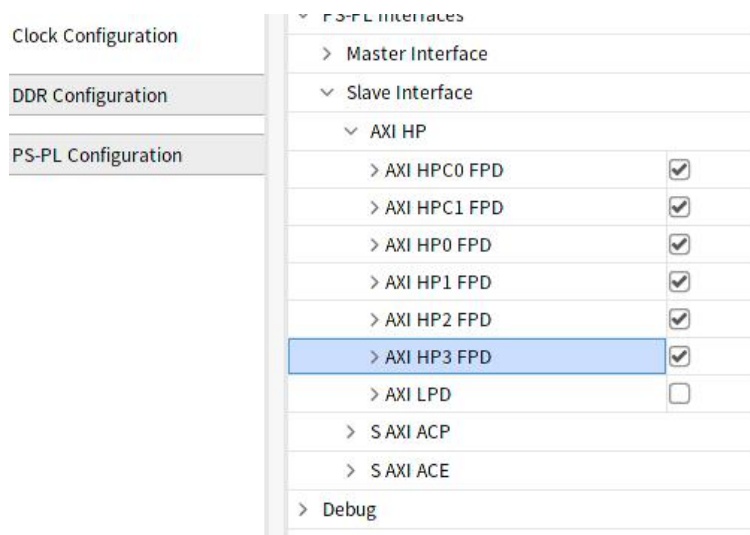


**PL 的时钟配置**



## AXI Interface 设置

这里为了更好的利用带宽，使用了 6 个 HP 口，每个 HP 口默认的位宽为 128bit，频率由 PL 决定。这里确保带宽瓶颈不在 HP 接口（96Bytes/cycle）。已经满足 ZCU104/102 的理论带宽。



## 2.3 添加 AXI 桥

这里选择 **AXI SmartConnection IP** 核，相较于上一代的，该 IP 核心更优化。

这里经过测试，为每一个 HP 口添加一个单独的桥性能和资源上都优于 6 个 HP 口共用相同的桥。

共享的桥：





**unsigned all** 然后这里直接进行 **Auto Assign all** 操作。

之后，点击下方的对号，进行 valide 该设计



若有 **critical warning**，则很大程度上下一阶段会报错。

然后点击 Sources 导航栏下的.bd 文件，左键选择 “create HDL Wrapper”

选择自动生成。之后再左键选择 “generate output Products”

### 3. 综合以及实现

上一步完成后可以直接点击” Generate Bitstream “直接到生成 bit 文件，这里并不需要引脚约束。资源消耗以及静态功耗可以查看实现后的报告。

### 4. 导出硬件

File-> export->export hardware,选择 include bitstream，则将生成 xsa 文件

#### Output

Set the platform properties to inform downstream tools of the intended use of the target platform's hardware design.

- ☐ Pre-synthesis  
This platform includes a hardware specification for downstream software tools.
- ☒ Include bitstream  
This platform includes the complete hardware implementation and bitstream, in addition to the hardware specification for software tools.

## 三、Vitis 部分

这里需要安装 **putty**，linux 系统中，vitis 自带的终端很难用。

## 1. 创建 platform 工程

### Create Platform Project

Create new platform project

Enter a name for your platform project



This wizard will guide you through creation of a platform project from the output of Vivado [Xilinx Shell Archive (XSA)] or from an existing platform. A platform will enable you to specify options for the kernels, BSPs, as well as settings required for creating new applications. Platforms are currently supported for embedded software developers.

Platform project name:

- A platform provides hardware information and software environment settings.
- A system project contains one or more applications that run at the same time.
- A domain provides runtime for applications, such as operating system or BSP.
- A workspace can contain unlimited platforms and unlimited system projects.

选择上一步生成的 xsa 文件，后 finish

Create a new platform from hardware (XSA) | Select a platform from repository

Hardware Specification

XSA File:

Software Specification

Specify the details for the initial domain to be added to the platform. More domains can be added after the platform is created by double clicking the platform.spr file

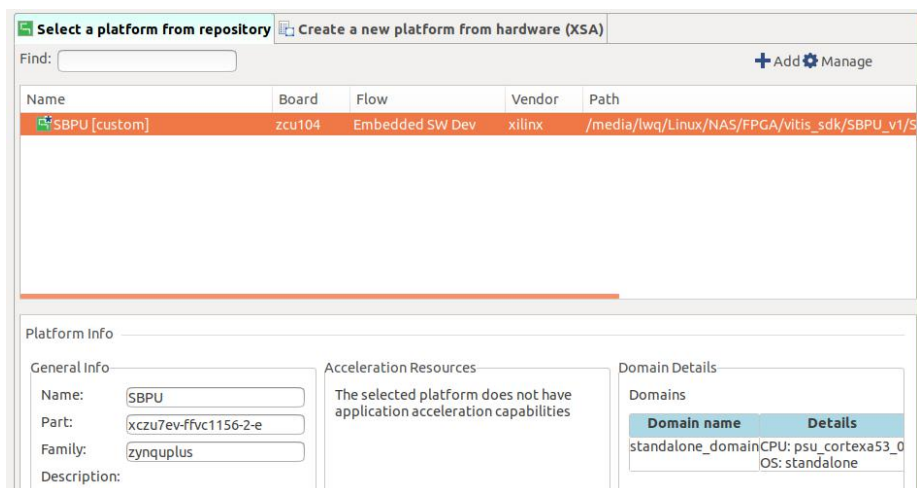
Operating system:

Processor:

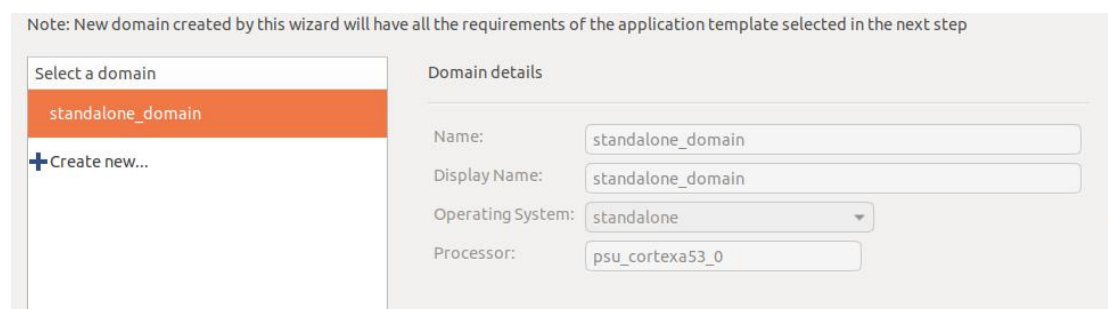
## 2. 创建 application 工程

File-> New-> create a New Application Project

选择上一步生成的 platform



选择默认的 standalone 即可

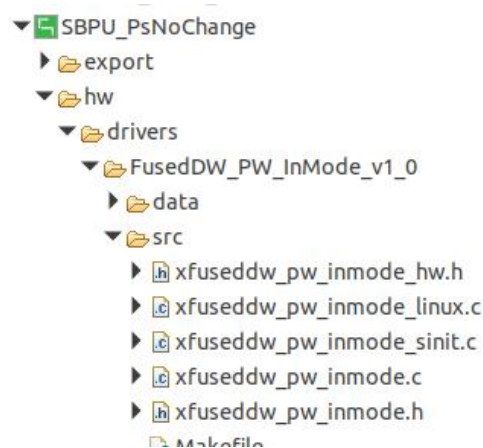


之后选择 hello world 即可

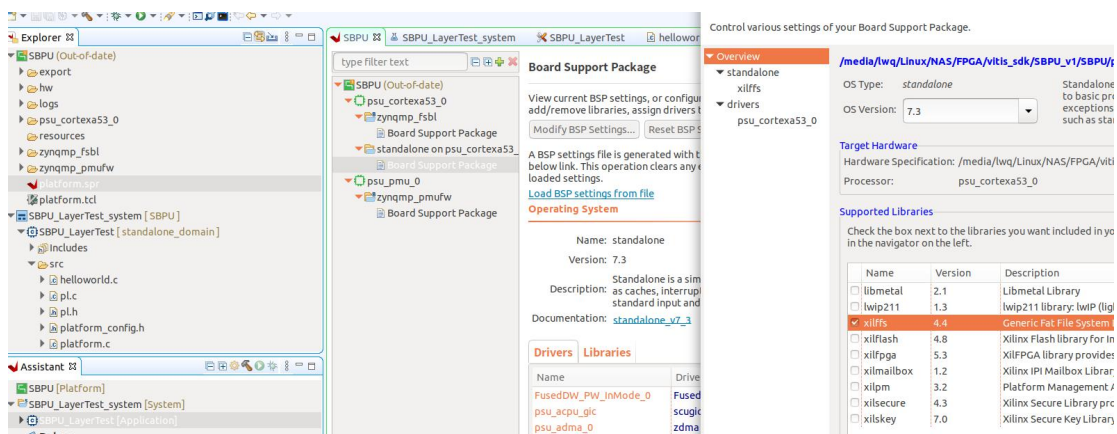
### 3. 编写测试程序

这里可以使用 SD Test 文件中的测试文件，进行 SD 卡读写测试。注意，SD 卡格式化为 FAT 格式。

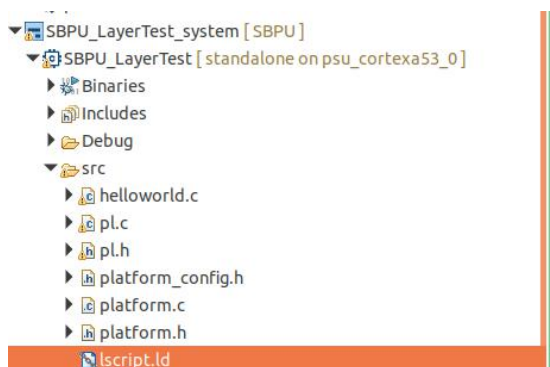
我们这里直接 则将和具体 HLS IP 核心相关的测试文件粘贴至工程目录中的 src 文件夹中，进行替换。其中所需的涉及硬件 IP 的驱动。在该目录下，这和 Vitis HLS 中导出的是一致的。



**注意这里的关键配置** ,选择 standalone -> Board Support Package ,修改 BSP ,将 xiff.h 勾选上 , 确保包含了访问 SD 卡的驱动。注意不需要改 xiffs.h 的具体配置 , 否则编译将失败。



**配置堆栈大小 , 否则测试文件较大时 , 无反应**



堆栈大小改为 0x2000 -> 0x2000000

最后点击小锤子 , 对程序进行编译。大概 1min

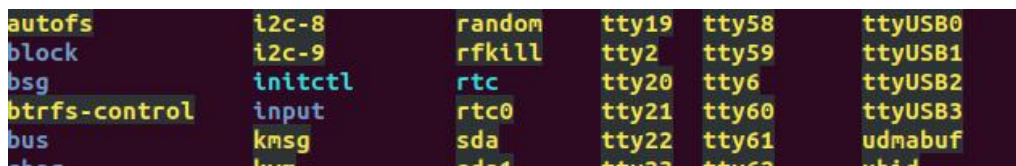


## 4. Program FPGA

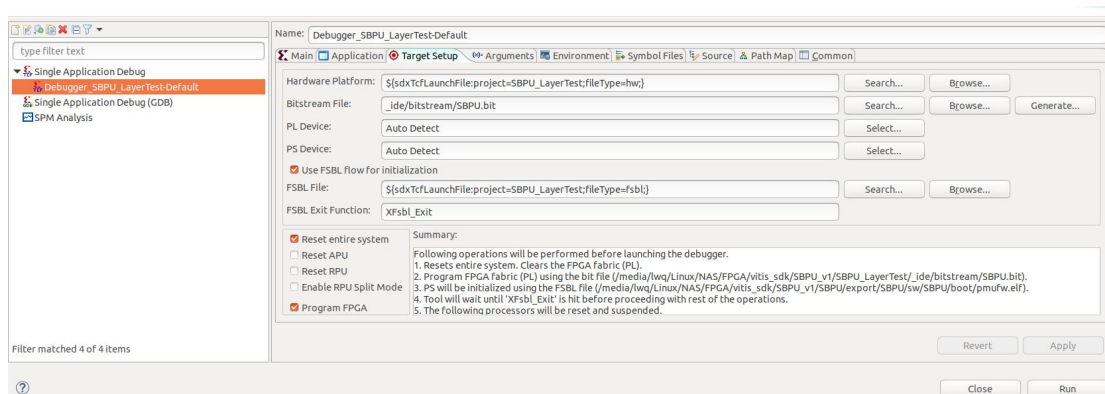
在这之前，先检查启动设置是否正确，以及是否检测到串口。

**将 mode 0,1,2,3 都向箭头方向推上，从 jtag 启动。**

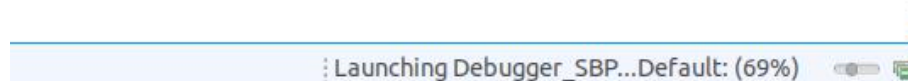
Linux 下 ls /dev，将出现 ttyUSB0~3。这其中的某一个，具体需要试一下，我这里是 ttyUSB1。这里建议不使用 vitis 自带的串口工具，较难使用



Sudo putty，检测串口，FPGA 上电之后，在 vitis 端，选择“run configuration”启动程序，这里第一次选择 Program FPGA。



点击 run 之后。软件有下方显示进度。



在 putty 端口等待

这里返回的数据可以和 Vitis HLS 中的报告进行对比，我这里两者仅有 5%的差距，可以说

非常接近了。

```
Kilinx Zynq MP First Stage Boot Loader
Release 2020.2 Jul 20 2022 - 09:00:08
PMU-FW is not running, certain applications may not be supported.
Begin Test
total time elapsed is 24089 us
time elapsed is 481 us
=====Right Result=====
Test end
```