# before-boot-testing

## 简介

This is a before-boot testing for memory access and multi-core communication on Chipyard-generated SoC.

这是一个启动前进行访存与多核通讯验证的测试用例，用于 Chipyard 所生成的 SoC。

## 快速开始

假设 Chipyard 与 Vivado 已经配置好，相应的环境变量已经设置（如 Chipyard 的 env.sh 与 vivado 的 settings64.sh）。

进入 Chipyard 的根目录，克隆本项目：

git clone git@github.com:ljchen98/before-boot-testing.git

进入 before-boot-testing 文件夹，并运行 beforeBootTestingGenerator.py：

cd before-boot-testing  
python beforeBootTestingGenerator.py

beforeBootTestingGenerator.py 实质上是运行其中的 beforeBootTest 函数，主要完成下面三件事：

* 根据函数参数生成测试用例的配置文件 testConfig.h；
* 调用 util/setup\_boot\_test.sh，备份 BootROM 原始代码中的 head.S、 sd.c、 smp.h、 kprintf.c 文件，并从本项目目录下的 util 文件夹中拷贝已加入测试用例支持的这些文件到相应位置；
* 编译 BootROM 新的代码。

回到 Chipyard 的根目录，并进入 fpga 文件夹，走 FPGA prototyping 的流程生成 bitstream，如：

cd ../fpga # 进入 Chipyard/fpga 目录  
make SUB\_PROJECT=vcu108 CONFIG=My4CoreRocketVCU108Config bitstream

备注：

1. 运行 beforeBootTestingGenerator.py实质上是运行其中的 beforeBootTest 函数，故在其他 python 程序中调用该函数亦能工作，但需要保证：1）util 文件夹与 beforeBootTestingGenerator.py 在同一目录下，因为该函数会利用相对路径使用到 util 文件夹内的文件；2）给予该函数的 outputPath 参数需为到 sdboot 文件夹的相对路径。
2. 若需要调用beforeBootTest 函数，需提供以下参数：1）核心数量 coreNum；2）L2\_cache\_size，用于计算访存范围的大小；3）文件替换路径 outputPath，即 sdboot 的相对路径。
3. 尽管beforeBootTest 函数会调用 util/setup\_boot\_test.sh 来备份 BootROM 的原始代码，但是如果需要恢复备份，需在项目文件夹before-boot-testing 下运行（最后一个参数为到 sdboot 文件夹的相对路径）：

* chmod u+x ./util/setup\_boot\_test.sh # 如果脚本还没有执行权限  
  ./util/setup\_boot\_test.sh clean ../fpga/src/main/resources/vcu118/sdboot

1. 尽管beforeBootTest 函数会自动编译 BootROM 新的代码，但是如果需要手动编译，进入了sdboot 目录后，需要先清除以往的编译，再编译：

* # 假设当前目录在 Chipyard 的根目录下  
  cd fpga/src/main/resources/vcu118/sdboot  
  make clean # 最好先清除之前的编译  
  make

## 工作原理介绍

### 基本原理

通过在 BootROM 中添加额外的测试用例，以完成在 Linux 系统启动前对 SoC 的访存与多核通讯进行验证。在系统启动前，由于内存里无大部分有意义的内容，方便对内存进行写读操作而不必考虑对现有数据的影响。此外，由于 SoC 的功能都被映射在内存地址空间上，对 SoC 的访存测试亦能完成对 SoC 的其它功能的测试。

### 测试用例验证范围

目前，测试用例会根据 SoC 所使用的 CPU 核心数目，完成如下测试任务：

* 各个核心单独背靠背地写读内存；
* 各个核心单独非背靠背地先写一段内存，再读相应的内存；
* 各个核心单独背靠背地使用各个数据位写读内存；
* 各个核心单独非背靠背地使用各个数据位先写一段内存，再读相应的内存；
* 对于每两个核心，一个核心先写一段内存，另一个核心再读相应的内存。

### 对 BootROM 的修改

BootROM 内的启动程序于 Chipyard 的 /fpga/src/main/resources/vcu118/sdboot路径内。

./setup\_boot\_test.sh会先备份添加测试用例后被更改的文件，再下载并添加测试用例的相关文件。其中：

* head.S 为上电时最先开始运行的代码。当有多个核心时，里面的代码同时运行，需要通过死循环与中断的添加来软件上控制多核的任务分配。添加了测试用例的调用；
* sd.c 内的函数被 head.S 调用，除了配置 SD 卡启动系统相关外，测试用例也在此添加；
* include/smp.h 也被 head.S 使用，主要用于添加死循环与中断来软件上将任务分配给单独的某一个核。添加了对任一核使用而其它核不使用的支持；
* kprintf.c 用于 SoC 与上位机通讯的信息打印。添加了对整数打印的支持；
* testConfig.h 为新增的文件，用于配置测试用例，测试用例需要此文件才能运行，由 generator.py 中的 beforeBootTest 函数自动生成。