说明

一、修改过的sniper-sim依赖以下软件包：

（1）libzmq

（2）cppzmq

（3）boost

此外，还需要使用cmake生成编译脚本。

各模块功能：

（1）interchiplet：转发不同模拟器实例间的消息，提供通讯功能。

（2）sniper（经修改）：模拟器，用于执行任务程序（benchmark）。每一个模拟器实例就是一个芯粒。

（3）interchiplet\_app：一个静态库，包装了一些通讯接口，用于任务程序芯粒间的通讯。

（4）record\_combine：整合各个模拟器的跨芯粒通讯记录，并将其转换成popnet可使用的轨迹文件。

（5）popnet（经修改）：读取轨迹文件并模拟，得出延时等指标。

二、编译：

（1）在sniper目录下运行“make”。

（2）在sniper目录下的changes/build依次执行“cmake ..”和“make”。

（3）在interchiplet目录下的build执行“cmake ..”和“make”。

（4）在interchiplet\_app目录下的lib执行“cmake ..”和“make”。

（5）将record\_combine目录下的两个cpp文件分别编译成两个可执行文件。

（6）在popnet目录下运行“make”。

三、使用：

（1）编写任务程序：运用interchiplet\_app提供的通讯接口，编写任务程序的代码。要包含interchiplet\_app.h头文件和链接libinterchiplet\_app.a静态库。编译任务程序代码，得到可执行文件。

（2）执行模拟：配置interchiplet的build/interchiplet.ini文件。在各模拟器实例的changes/zmq\_pro.ini文件配置芯粒号和连接到interchiplet的地址 。在interchiplet目录下的build中运行./interchiplet，在sniper目录下运行“./run-sniper （任务程序命令行）”。注意，有多少个芯粒就要复制多少份sniper，单独配置，同时运行。

（3）合并转换通讯记录：将各sniper目录下的record\_\*.txt文件复制到任意同一目录，然后在该目录下先后运行record\_combine的record\_combine和popnet\_trace\_transfer。record\_combine需要一个命令行参数，即芯粒数量。popnet\_trace\_transfer需要参数有芯粒每行数量、芯粒内核心每行数量和以GHz为单位的频率（将芯粒和核心从0开始，排满第0行，然后在排满第1行、第2行……）。

（4）延时模拟：使用popnet模拟上一步得到的轨迹文件。注意c=4，R=2。