MVPGPU-Sim Architecture Refactoring Design Principles

基本原则

- 类的实现和声明分开在cpp和hpp文件中，便于复用
- 模块化，将相关功能单元进行模块化设计
- 每个模块以动态链接库的形式生成
- 用CMakeLists.txt构建库或者应用

可编程部分与固定管线的交互设计

- 方案一 **将固定管线部分的实现直接放到整体GPU的框架下，完全按照GPU的设计思路**
  - 编译问题较麻烦
- 方案二 **将固定管线部分单独放置，GPU框架下再调用固定管线的实现**
  - 编译问题好解决
  - 可独立为模块进行release

构建原则

1. 以动态链接库的形式为主，各模块构建为独立的动态链接库so
2. 用CmakeLists.txt构建(主要针对新增或重构模块)

当前的构建结构

libOpenCL.so

- $(LIBS)
  - driver
  - gpu
    - driver
      - driver/cuda_sim
    - gpu_uarch
      - driver

        - gpu/gpu_uarch
        - gpu/gpu_uarch/mvp_core
      - $(INTERSIM)
    - $(INTERSIM)
      - driver
      - gpu_uarch
    - opencllib
      - driver
      - api/libopencl

期望的构建结构

libOpenGL.so

- $(LIBS)
  - libdriver.so
  - libgpu.so
    - libuarch.so
    - libgraphics.so
    - libgpuwattch.so
    - libintersim2.so
    - libhardwaremodel.so

动态库模板

```
$(SIM_LIB_DIR)/libOpenCL.so: makedirs $(LIBS) libopencl g++ -shared -Wl, -soname, libOpenCL.so \
                             $(MCPAT) \
                             $(SIM_OBJ_FILES_DIR)/libopencl/*.o \
                             -o $(SIM_LIB_DIR)/libOpenCL.so
```

CMakeLists.txt模板

```
cmake_minimum_required(VERSION 3.10)
SET(CMAKE_C_COMPILER "/usr/bin/gcc-9")
SET(CMAKE_CXX_COMPILER "/usr/bin/g++-9")

project(Graphic)

find_package(OpenCV REQUIRED)

set(CMAKE_CXX_STANDARD 17)
set(CMAKE_CXX_FLAGS   "-g")
set(CMAKE_CXX_FLAGS_DEBUG   "-O0" )
set(CMAKE_CXX_FLAGS_RELEASE "-O2 -DNDEBUG " )

include_directories(/usr/local/include/opencv4 include texture rast pa rop)
include_directories($ENV{GPGPUSIM_ROOT}/gpu $ENV{GPGPUSIM_ROOT} $ENV{GPGPUSIM_ROOT}/include
                    $ENV{CUDA_INSTALL_PATH}/include $ENV{GPGPUSIM_ROOT}/api)

link_directories($ENV{GPGPUSIM_ROOT}/lib/gcc-5.3.1/cuda-11000/debug)

add_library(Graphic SHARED models/OBJ_Loader.cpp pa/Primitive_assemble.hpp pa/Primitive_assemble
            rast/Rasterizer.cpp include/global.hpp Triangle.hpp Triangle.cpp texture/Texture.cpp
            rop/Render_output.hpp rop/Render_output.cpp include/Shader.hpp include/OBJ_Loader.h

target_link_libraries(Graphic ${OpenCV_LIBRARIES} OpenCL)

#message(${OpenCV_LIBRARIES})
message($ENV{GPGPUSIM_ROOT}/include)

add_custom_command( TARGET Graphic
                    POST_BUILD
                    COMMAND ${CMAKE_COMMAND} -E copy ./libGraphic.so $(GPGPUSIM_ROOT)/lib/libGra
)
```