Verilog Image Rotation

项目使用Verilog输入图像 旋转90度输出。

项目依赖

1. `Verilator`
2. `python>=3.6`
    1. `pillow`
3. `c++ tool chain`
    1. `STBI`库 使用`git submodule`进行管理
4. `GNU Make`

在ArchLinux中 你可以使用以下命令配置环境 这里使用yay作为包管理器

```
# 0. clone此项目
git clone --recursive https://github.com/steven12138/image_rotate_verilog.git
# 或者如果已经clone了项目（解压了提交的zip包）
git submodule update --init --recursive #更新submodule

# 1. python
yay -S python
yay -S python-pillow # 使用yay作为包管理
pip install pillow # 使用pip作为包管理器

# 2. C++
yay -S gcc

# 3. verilator
yay -S verilator

# 4. make （通常来说都不用单独安装）
yay -S make
```

对于非Archlinux用户 Verilator也提供了配置方法， 可以参考Verilator WIKI

```
# Ubuntu 可以使用
sudo apt-get install verilator   # On Ubuntu
```

对于Windows用户 推荐使用WSL来构建此项目

运行

```
make            # 运行全部流程
make clean      # 清除构建文件 重新构建
make png2bmp    # 转换原文件为标准bmp和标准大小
```

文件内容解释

Verilog 文件

```
src/our.v       # Hello World 文件, 用于测试是否可以被调用
src/sram.v      # SRAM, 用于存储输入信息
src/adapter.v   # Adapter, 流程控制和存储控制
```

Makefile

`Makefile`用于控制项目的构建流程和每一步的依赖关系

```
TOP_MODULE=adapter
BUILD_DIR=obj_dir

VERILOG_SRC=$(wildcard ./src/*.v)
CPP_SRC=$(wildcard *.cpp *.hpp)

VERILOG_FLAGS= --cc --exe -Wall --top-module $(TOP_MODULE)

all: run

# 验证 Verilog Module
verify: $(VERILOG_SRC)
    verilator $(VERILOG_FLAGS) $(VERILOG_SRC) $(CPP_SRC)
# 编译构建
compile: verify
    make -j -C $(BUILD_DIR) -f V$(TOP_MODULE).mk V$(TOP_MODULE)
# 运行编译结果
run: compile png2bmp
    ./$(BUILD_DIR)/V$(TOP_MODULE)
# 标准化转换
png2bmp: res/lenna.png
    python converter.py ./res/lenna.png
# 清除缓存
clean:
    rm -rf $(BUILD_DIR)
    rm ./output_lenna.bmp
    rm ./res/lenna.bmp
    rm ./output.bmp
```

C++

`test.cpp`C++文件用于实施TestBench， 参考了Verilator的wiki的c++部分

python

`converter.py` 用于从网络下载的lenna.png 转换，转换为lenna.bmp `256x256`， `24bit`位宽