实验三 缓存

首先，更新代码仓库：

```
cd Arch-2022Spring

# 首先，commit 之前的代码。如果你已经 commit，就跳过下面两行
git add .
git commit -m "lab2"

git remote -v
# 如果地址是git://github.com/FDUCSLG/Arch-2022Spring-FDU.git，则执行这一步：
git remote set-url origin https://github.com/FDUCSLG/Arch-2022Spring-FDU.git

# 这一步如果卡很久，就 Ctrl-C 后重试
git fetch --all

git merge origin/master
# Merge 之后可以需要处理 Merge Conflict，在VS Code里可以很方便地处理
git submodule update
```

更新完成后，运行 `make test-refcache`，如果报错 `verilated.h` 之类的头文件 `not found`，请把 `verilator` 的目录软链接到 `/usr/share/verilator/`，比如：

```
ls /usr/local/share/verilator

# if exists
sudo ln -s /usr/local/share/verilator /usr/share/verilator
```

再运行 `make test-refcache`。

1 支持更多的指令

添加以下指令：

MUL DIV DIVU REM REMU MULW DIVW DIVUW REMW REMUW

要求实现多周期乘除法器。不允许使用 `/` 与 `%` 运算符，允许使用 `*` 运算符。可参考 多周期乘除法器，有竖式乘除法的实现思路。

需要判断除以零的情况，设置余数为被除数，商为 `'1`。

高阶内容：

- 除法器 shift over zero, high radix （见《计算机体系结构-量化分析方法》）
- Tree-based 乘法器 （见《计算机体系结构-量化分析方法》）
- DSP-based 乘法器（在评论区/slack群里一起讨论）

2 实现缓存

详见缓存的文档。

3 测试方法

3.1 Verilator 仿真

检测是否通过：

- **使用 `make test-lab3 TEST=all`，查看是否有 `HIT GOOD TRAP`。**
- **使用 `make test-cache`。**

3.2 Vivado 仿真与上板

在 Vivado 中执行 `vsrc/add_sources.tcl` 添加源文件。

将实验板上右侧四个开关往下拨，连接串口软件，然后 Program Device。

在 Vivado 中执行 `run simulation` 即可开始仿真。在 `vivado/src/with_delay/simtop.sv` 中修改 `.sw(0)`，可执行单个测试。

4 作业提交

DDL：**5月15日23:59**

提交内容：以学号命名的`tar`压缩包，如`18307130024.tar`，包括 `vsrc` `verilate` 和 `report.pdf`。

生成方法：

```
mkdir 18307130024
cp -r vsrc verilate 18307130024
cp report.pdf 18307130024
tar cf 18307130024.tar 18307130024
```

你需要通过 Verilator 仿真和 Vivado 上板。

实验报告需要包括：

- 为了支持随机延迟，流水线的改动
- 测试通过的截图

实验报告里不应有大段代码的复制。

通过测试+实验报告有上述内容，本次实验就可以满分。

实验报告里可以有：

- 对本门课程实验（文档、代码风格、视频录制等）的建议
- 对后续内容的期待