Cache 实验常见疑问：

问：无论进行快速排序，还是矩阵乘法，最终的主存（mem.sv 模块里的 ram_cell 变量）里的数据都与正确结果整体上是相同的，但会略有差异，是为什么？

答：因为是写回策略的 cache，所以最终会有一些数据还在 cache 中未写入主存。属于正常现象。但如果你的 cache 写错了，那么快速排序和矩阵乘法的结果就会很离谱。在检查实验时，助教主要通过第一阶段 cache_tb.sv （即脱离 CPU 的 cache 检验）去判断你的 cache 的正确性。

问：Cache 实验都检查什么内容

答：阶段 1 的 cache_tb 必须正确通过，然后需要向助教展示你所写的 cache 代码，讲解你的思路，包括但不限于如何在原有基础上加并行 TAG 比较、如何实现 LRU 和 FIFO 换出策略。阶段 2 的分值不高，只需要展示你所写的 LRU 和 FIFO 策略的 cache 能够跑通快速排序和矩阵乘法就行，对于快速排序来说，跑通的结果应该是主存中的数据变成有序的。对于矩阵乘法来说，跑通的结果应该是主存中的数据与注释中的数据相同。当然，因为有些数据在 cache 中还未写入主存，所以允许有一些不同。

问：如何写 cache 实验的报告

Cache 实验报告在 cache 实验中占很大的分值比重。因为 Cache 实验是相对于 CPU 流水线设计而言是开放的，重点在权衡性能和面积的体会，而 Cache 本身的实现只是前提。阶段二所提供的快速排序和矩阵乘法的 benchmark 就是用来在实验报告中进行实验分析的，也鼓励自己编写更多的汇编 benchmark 进行测试。在做实验时，修改 Cache 的参数，体会 cache size、组相连度、替换策略针对不同程序的优化效果，以及策略改变带来的电路面积的变化。针对不同程序，权衡性能和电路面积给出一个较优的 cache 参数和策略。注意，需要权衡性能和电路面积。

其中“性能”参数使用运行仿真时的时钟周期数量进行评估。“资源占用”参数使用 vivado 给出的综合报告进行评估。在写报告时，不能仅仅进行理论分析，实验报告中需要给出实验结果（例如仿真波形的截图、vivado 综合报告等）。

问：如何修改 cache 的参数

答：cache.sv 的开头中有如下的 parameter 定义：

```
module cache #(
    parameter  LINE_ADDR_LEN = 3, // line内地址长度，决定了每个line具有2^3个word
    parameter  SET_ADDR_LEN  = 3, // 组地址长度，决定了一共有2^3=8组
    parameter  TAG_ADDR_LEN  = 7, // tag长度
    parameter  WAY_CNT       = 3  // 组相连度，决定了每组中有多少路line，这里是直
```

这些参数应该在编写实验报告时进行修改，用于权衡性能和电路面积。修改方法是在调用它的模块（即 WBSegReg.v）中，修改这几个参数，如下：

```
cache #(
    .LINE_ADDR_LEN  ( 3             ),
    .SET_ADDR_LEN   ( 3             ),
    .TAG_ADDR_LEN   ( 5             ),
    .WAY_CNT        ( 4             )
) cache_test_instance (
    .clk            ( clk           ),
    .rst            ( rst           ),
    .miss           ( CacheMiss     ),
    .addr           ( A             ),
    .rd_req         ( MemToRegM     ),
    .rd_data        ( RD_raw        ),
    .wr_req         ( |WE           ),
    .wr_data        ( WD            )
);
```

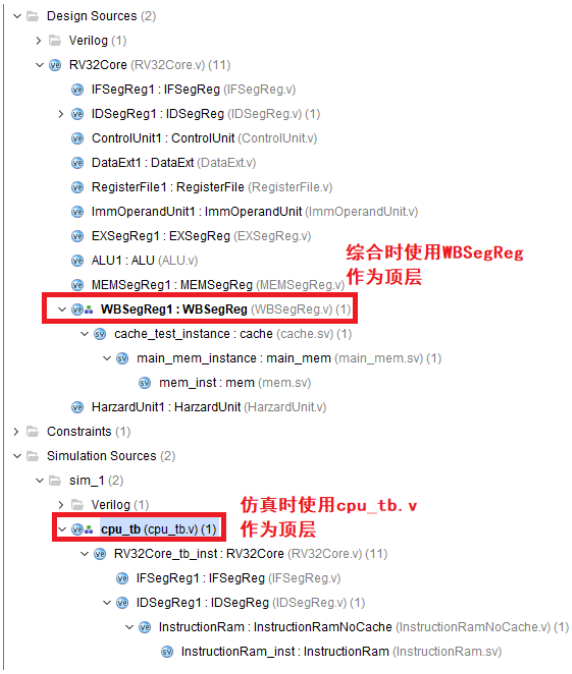
Verilog/SystemVerilog 中的 parameter 类似 C++中的默认参数，在被调模块中只是指定默认值。在主调模块中，如果指定，则使用该指定值；如果不指定，则使用默认值。

问：vivado 的综合和仿真是否互相依赖？

答：没有，综合前不需要仿真，仿真前也不需要综合。

问：为了进行电路面积评估，如何进行综合？

答：因为我们只关心 cache 的电路面积，而 CPU 我们是不会修改的，所以建议在综合前，将 WBSegReg.v 设置为顶层，可以提高综合速度。在 vivado 中应该如下设置仿真和综合的顶层：



因为仿真与综合互相独立，所以可以使用不同的顶层。设置 cpu_tb.v 作为仿真的顶层，用于进行 cache 性能的评估。设置 WBSegReg.v 作为综合的顶层，综合后查看综合报告，用于进行电路面积的评估。

问：综合特别慢怎么办？

在 5 月 16 日之前提供的 mem.sv 里，ram_cell 的定义为：

```
reg  [31:0] [0:MEM_SIZE-1] ram_cell;
```

要改成

```
reg  [31:0] ram_cell [MEM_SIZE];
```

后者 vivado 会综合成 BRAM，所以很快，前者 vivado 会综合到 LUT 里，所以慢。5 月 16 日后的 github 库修改了这个问题。修改后综合一般只需要不到 1 分钟。