第二次实验报告

20307130112 马成

一、 解决冲突的策略

在实验 1 中我解决冲突的策略是使用的气泡法，在这次实验中我先将解决冲突的策略调整成了转发法，结构法老师已经搭建框架这里不做考虑，控制冒险方面我在实验 1 中好像就已经使用了转发，只是等待了一个时钟。下面主要讨论数据冒险的情况

1. 在这次的实验中我将已经向后传输的数据的 dst、result、bubble 和 regwrite 回传到 decode 模块中，依次检验 excute、memory、write 的各个数据的 bubble&regwrite 是不是 1，dst 和 decode 中译码出来的 ra1，ra2 是否相同。一般情况下如果相同的话将对应的数值直接替换成传输回来的 result
2. 特殊情况就是当 excute 的数据 dst 和某一个 ra 相同但是译码结果显示这一条 Load 的指令的话，回传的 result 其实是需要访问内存的地址，这是不能直接替换，我是在 decode 中给出了气泡，等到这一条数据在 memory 模块中完成了取值在接受对应的回传。

二、 为了支持随机延迟，流水线的改动

1. 在 fetch 阶段，由于我们一直需要取出指令，ireq.valid=1 一直保持即可，我在外部添加了一条 1 位的 stop_forfetch 用于指示当前是不是因为 fetch 指令的延长而需要阻塞，如果 stop_forfetch=1 的时候保持 pc 不变，dataF.bubble=1 保证后续的数据全都无效，知道 data_OK=1 的时候将 stop_forfetch=0 将指令传给后续的模块
2. 在 memory 阶段，由于需要更多的粒度，使用了老师给的 readdata 和 writedata 模块处理数据和 strobe 信号。当检测到访存指令的时候将 dreq.valid 写为 1，在 data_OK=0 的时候将 1 位 stop_formem 总线信号置为 1。其他模块全部获取 stop_formem 信号，并让他们只有在这个信号为 0 的时候才将 data 赋值为 data_next 否则保持数据传输不变用于阻塞。当 data_OK=1 的时候 stop_formem=0 继续流水线的流动即可

```
always_ff @( posedge clk ) begin //对于 stop_formem 总线的相应
        if (reset) begin
           dataE<='0;
        end
        else if (~stop_formem) begin
            dataE<=dataE_next;
        end
    end
```

三、 通过截图

1. Verilator 通过截图

```
-------------------------------------------------------------
Function    Best Rate MB/s  Avg time     Min time     Max time
Copy:           5.5         0.006556     0.006000     0.007000
Scale:          0.1         0.267111     0.266000     0.268000
Add:            0.7         0.072889     0.070000     0.078000
Triad:          0.2         0.322667     0.320000     0.327000
-------------------------------------------------------------
Solution Validates: avg error less than 1.000000e-13 on all three arrays
-------------------------------------------------------------
Run conwaygame
Play Conway's life game for 200 rounds.
seed=10142
**
**
                    **
                    **              **     *
                                    **    * *
                                           *


                                 **
                                * *
                                 *
[src/cpu/cpu-exec.c,320,cpu_exec] nemu: HIT GOOD TRAP at pc = 0x000000008000152c0
[src/cpu/cpu-exec.c,321,cpu_exec] trap code:0
[src/cpu/cpu-exec.c,62,monitor_statistic] host time spent = 144129710 us
[src/cpu/cpu-exec.c,64,monitor_statistic] total guest instructions = 59171279
[src/cpu/cpu-exec.c,65,monitor_statistic] simulation frequency = 410541 instr/s
Program execution has ended. To restart the program, exit NEMU and run again.
sh: 1: spike-dasm: not found

============== Commit Group Trace (Core 0) ==============
commit group [0]: pc 0080015078 cmtcnt 1
commit group [1]: pc 008001507c cmtcnt 1
```

2. 上板通过截图