

Laboratory 3

Expected delivery of lab_03.zip must include:

- program_1_a.s, program_1_b.s, and program_1_c.s
- This file, filled with information and possibly compiled in a pdf format.

This lab will explore some of the concepts seen during the lessons, such as hazards, rescheduling, and loop unrolling. The first thing to do is to configure the WinMIPS64 simulator with the *Initial Configuration* provided below:

- Integer ALU: 1 clock cycle
- Data memory: 1 clock cycle
- Code address bus: 12
- Data address bus: 12
- FP arithmetic unit: pipelined, 4 clock cycles
- FP multiplier unit: pipelined, 6 clock cycles
- FP divider unit: not pipelined, 30 clock cycles
- Forwarding is enabled
- Branch prediction is disabled
- Branch delay slot is disabled

- 1) Enhance the assembly program you created in the previous lab called **program_1.s**:

```
int m=1 /* 64 bit */
double a, b
for (i = 31; i >= 0; i--){
    if (i is a multiple of 3) {
        a = v1[i] / ((double)m<<i) /*logic shift */
        m = (int) a
    } else {
        a = v1[i] * ((double) m* i)
        m = (int) a
    }
    v4[i] = a*v1[i] - v2[i];
    v5[i] = v4[i]/v3[i] - b;
    v6[i] = (v4[i]-v1[i])*v5[i];
}
```

- a. Manually detect the different data, structural, and control hazards that cause a pipeline stall.

Hazards:

1. Structural hazard between mtc1 r8, f8 and cvt.d.l f8, f8
2. RAW/WAW with ddiv r7,r2,r6 and dmul r7,r7,r6
3. RAW with both ddiv r7,r6,r6,dmul r7,r7,r6 and beq r7,r2,multiple_3
4. RAW between dmul r4, r3, r2 and mtc1 r4, f4
5. Structural hazard between dmul r4, r3, r2 and mtc1 r4, f4
6. RAW between mul.d f5, f2, f4 and cvt.l.d f6, f5

7. Structural hazard between mul.d f5, f2, f4 and cvt.l.d f6, f5
8. RAW/WAW between sub.d f6, f6, f2 and mul.d f6, f5, f1
9. RAW between s.d f6, v4(r1) and sub.d f6, f6, f2
10. RAW between div.d f7, f6, f3 and sub.d f6, f6, f2
11. RAW between sub.d f7, f7, f8 and s.d f7, v5(r1)
12. RAW/WAW between sub.d f7, f7, f8 and div.d f7, f6, f3
13. RAW between s.d f7, v5(r1) and sub.d f7, f7, f8
14. Structural hazard between sub.d f7, f7, f8 and s.d f7, v5(r1)
15. RAW between mul.d f9, f9, f1 and s.d f9, v6(r1)
16. Structural hazard between mul.d f9, f9, f1 and s.d f9, v6(r1)
17. RAW between slt r10, r2, 0 and beq r10, 0, cycle
18. RAW between div.d f5, f1, f4 and cvt.l.d f6, f5
19. Structural hazard between div.d f5, f1, f4 and cvt.l.d f6, f5
20. RAW between dmul r4, r3, r2 and r4, f4
21. Structural hazard between dmul r4, r3, r2 and r4, f4
22. RAW between mul.d f5, f2, f4 and cvt.l.d f6, f5
23. Structural hazard between mul.d f5, f2, f4 and cvt.l.d f6, f5

- b. Optimize the program by re-scheduling the program instructions to eliminate as many hazards as possible. Manually calculate the number of clock cycles for the new program (**program_1_a.s**) to execute and compare the results with those obtained by the simulator.
- c. Starting from **program_1_a.s**, enable the *branch delay slot* and re-schedule some instructions to improve the previous program execution time. Manually calculate the number of clock cycles needed by the new program (**program_1_b.s**) to execute and compare the results obtained with those obtained by the simulator.
- d. Unroll the program (**program_1_b.s**) 3 times; if necessary, re-schedule some instructions and increase the number of registers used. Manually calculate the number of clock cycles to execute the new program (**program_1_c.s**) and compare the results obtained with those obtained by the simulator.

Complete the following table with the obtained results:

Program	program_1.s	program_1_a.s	program_1_b.s	program_1_c.s
Clock cycle computation				
By hand				
By simulation	<u>2922</u>	<u>2730</u>	<u>2677</u>	

2) Collect the Cycles Per Instruction (CPI) from the simulator for different programs

	program_1.s	program_1_a.s	program_1_b.s	program_1_c.s
CPI	<u>3.423</u>	<u>3.200</u>	<u>3.179</u>	

Compare the results obtained in 1) and provide some explanation if the results are different.

Eventual explanation: