Computer Architectures **02LSEOV** # Laboratory 3 **S339232** **Bonenti Silvia**	Delivery date: October 22nd 2024, 11.59 PM 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.

Data Hazards:

1. RAW/WAW for: *dmulu r4,r4,r3 - daub r4,r2,r4 - sub.d f4,f4,f2 - sub.d f5,f5,f9 - mul.d f6,f6,f5*

2. RAW for: *cvt.l.d f10, f8 - cvt.l.d f10, f8 - s.d f4, v4(r1) - s.d f5, v5(r1) - s.d f6, v6(r1)*
3. WAW for: *mtc1 r6, f7*

Control Hazards: at every branch condition:

beqz r4, multiple3 - j no_multiple3 - j end - bne r2, r5, cycle

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 **Clock cycle computation**	`program_1.s`	`program_1_a.s`	`program_1_b.s`	`program_1_c.s`
`By hand`	4112	3770	3729	1893
`By simulation`	4048	3846	3793	1893

Initial part (.text) :

daddui r3,r0,3	F	D	E	M	W																									
daddui r1,r0,248		F	D	E	M	W																								
daddui r2,r0,31			F	D	E	M	W																							
daddi r5,r0,-1				F	D	E	M	W																						
ld r6, m(r0)					F	D	E	M	W																					
l.d f9, b(r0)						F	D	E	M	W																				

CC: 10

Cycle part **program_1.s** :

l.d f1, v1(r1)	F	D	E	M	W						1	2																		
l.d f2, v2(r1)		F	D	E	M	W																								
l.d f3, v3(r1)			F	D	E	M	W																							
ddivu r4, r2, r3				F	D	E_D	...	x30	...	E_D	M	W																		
dmulu r4, r4, r3					F	D	...	x30	...	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W												
dsubu r4, r2, r4						F	...	x30	...	F	D	D	D	D	D	D	E	M	W											
beqz r4, multiple3											F	F	F	F	F	F	D	E	M	W										
j no_multiple3 (i mod 3 = 0)																	F	X												
j no_multiple3 (i mod 3 != 0)																	F	D	E	M	W									

CC: 42 if multiple3, 43 if no_multiple3, 42 if no_multiple3 and i=31: (42 * 12) + (43 * 20) = 1364

Cycle part **program_1_a.s** :

ddivu r4, r2, r3	F	D	E_D	...	...	x30	...	...	...	E_D	M	W		30	31															
l.d f1, v1(r1)		F	D	E	M	W																								
l.d f2, v2(r1)			F	D	E	M	W																							
l.d f3, v3(r1)				F	D	E	M	W																						
dmulu r4, r4, r3					F	D	...	x27	...	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W												
dsubu r4, r2, r4						F	...	x27	...	F	D	D	D	D	D	D	E	M	W											
beqz r4, multiple3											F	F	F	F	F	F	D	E	M	W										
j no_multiple3 (i mod 3 = 0)																	F	X												
j no_multiple3 (i mod 3 != 0)																	F	D	E	M	W									

CC: 39 if multiple3, 40 if no_multiple3, 39 if no_multiple3 and i=31: (38 * 12) + (40 * 20) = 1256

Cycle part **program_1_b.s** :

ddivu r4, r2, r3	F	D	E_D	...	...	x30	...	...	...	E_D	M	W		30																
l.d f1, v1(r1)		F	D	E	M	W																								
l.d f2, v2(r1)			F	D	E	M	W																							

dmulu r4, r4, r3				F	D	...	x28	...	...	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W												
dsubu r4, r2, r4					F	...	x28	...	...	F	D	D	D	D	D	D	E	M	W											
beqz r4, multiple3											F	F	F	F	F	F	D	E	M	W										
l.d f3, v3(r1)																	F	D	E	M	W									

CC: 39 * 32 = 1248

multiple3 **program_1.s**, **program_1_a.s**:

dsllv r6, r6, r2	F	D	E	M	W							2																		
mtc1 r6, f7		F	D	E	M	W																								
cvt.d.l f7, f7			F	D	E	M	W																							
div.d f8, f1, f7				F	D	E_D	...	x30	...	E_D	M	W																		
cvt.l.d f10, f8					F	D	...	x30	...	D	E	M	W																	
mfc1 r6, f10						F	...	x30	...	F	D	E	M	W																
j end											F	D	E	M	W															

CC: 37 * 11 = 407

multiple3 **program_1_b.s**:

dsllv r6, r6, r2	F	D	E	M	W							1																		
mtc1 r6, f7		F	D	E	M	W																								
cvt.d.l f7, f7			F	D	E	M	W																							
div.d f8, f1, f7				F	D	E_D	...	x30	...	E_D	M	W																		
cvt.l.d f10, f8					F	D	...	x30	...	D	E	M	W																	
mfc1 r6, f10						F	...	x30	...	F	D	E	M	W																

CC: 35 * 11 = 385

no_multiple3 **program_1.s**:

dmulu r6, r6, r2	F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W						7														
mtc1 r6, f7		F	D	D	D	D	D	D	E	M	W																			
cvt.d.l f7, f7			F	F	F	F	F	F	D	E	M	W																		
mul.d f8, f1, f7								F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W													
cvt.l.d f10, f8									F	D	D	D	D	D	D	E	M	W												

mfc1 r6, f10											F	F	F	F	F	F	D	E	M	W										
j end																	F	D	E	M	W									

CC: 18 * 21 = 378

no_multiple3 **program_1_a.s**:

dmulu r6, r6, r2	F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W						7														
mtc1 r6, f7		F	D	D	D	D	D	D	E	M	W																			
cvt.d.l f7, f7			F	F	F	F	F	F	D	E	M	W																		
mul.d f8, f1, f7									F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W												
cvt.l.d f10, f8										F	D	D	D	D	D	D	E	M	W											
mfc1 r6, f10											F	F	F	F	F	F	D	E	M	W										

CC: 17 * 21 = 357

no_multiple3 **program_1_b.s**:

dmulu r6, r6, r2	F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W						6														
mtc1 r6, f7		F	D	D	D	D	D	D	E	M	W																			
cvt.d.l f7, f7			F	F	F	F	F	F	D	E	M	W																		
mul.d f8, f1, f7									F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W												
cvt.l.d f10, f8										F	D	D	D	D	D	D	E	M	W											
j end											F	F	F	F	F	F	D	E	M	W										
mfc1 r6, f10																	F	D	E	M	W									

CC: 17 * 21 = 357

end **program_1.s**:

mul.d f4, f8, f1	F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W						7														
sub.d f4, f4, f2		F	D	D	D	D	D	D	E_A	E_A	E_A	E_A	M	W																
s.d f4, v4(r1)			F	F	F	F	F	F	D	D	D	D	E	M	W															
div.d f5, f4, f3									F	F	F	F	D	E_D	…	x30	…	E_D	M	W										
sub.d f5, f5, f9													F	D	…	x30	…	D	E_A	E_A	E_A	E_A	M	W						
s.d f5, v5(r1)														F	…	x30	…	F	D	D	D	D	E	M	W					
sub.d f6, f4, f1																			F	F	F	F	D	E_A	E_A	E_A	E_A	M	W	

mul.d f6, f6, f5	F	D	D	D	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W			6														
s.d f6, v6(r1)		F	F	F	F	D	D	D	D	D	D	E	M	W																
daddi r1, r1, -8						F	F	F	F	F	F	D	E	M	W															
daddi r2, r2, -1												F	D	E	M	W														
bne r2, r5, cycle													F	D	E	M	W													
halt (i >= 0)														F	X															
halt (i < 0)														F	D	E	M	W												

CC: 61 if i>= 0, 62 if i<0: (61 * 31) + (62 * 1) = 1953

end **program_1_a.s**:

mul.d f4, f8, f1	F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W					6	7														
sub.d f4, f4, f2		F	D	D	D	D	D	D	E_A	E_A	E_A	E_A	M	W																
div.d f5, f4, f3			F	F	F	F	F	F	D	D	D	D	E_D	…	…	…	x30	…	…	…	E_D	M	W							
sub.d f6, f4, f1									F	F	F	F	D	E_A	E_A	E_A	E_A	M	W											
sub.d f5, f5, f9													F	D	…	…	…	x29	…	…	D	E_A	E_A	E_A	E_A	M	W			

mul.d f6, f6, f5	F	…	x29	…	F	D	D	D	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W				6									
s.d f4, v4(r1)						F	F	F	F	D	E	M	W																	
s.d f5, v5(r1)										F	D	E	M	W																
s.d f6, v6(r1)											F	D	D	D	D	E	M	W												
daddi r1, r1, -8												F	F	F	F	D	E	M	W											
daddi r2, r2, -1																F	D	E	M	W										
bne r2, r5, cycle																	F	D	E	M	W									
halt (i >= 0)																		F	X											
halt (i < 0)																		F	D	E	M	W								

CC: (54 if i mod 3 != 0, 55 if i mode 3 = 0) if i>= 0, 56 if i<0: (54 * 21) + (55 * 10) + (56 * 1) = 1740

end **program_1_b.s**:

mul.d f4, f8, f1	F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W								6												
sub.d f4, f4, f2		F	D	D	D	D	D	D	E_A	E_A	E_A	E_A	M	W																

div.d f5, f4, f3			F	F	F	F	F	F	D	D	D	D	E_D	…	…	…	x30	…	…	…	E_D	M	W							
sub.d f6, f4, f1									F	F	F	F	D	E_A	E_A	E_A	E_A	M	W											
sub.d f5, f5, f9													F	D	…	…	…	x29	…	…	D	E_A	E_A	E_A	E_A	M	W			

mul.d f6, f6, f5	F	…	x29	…	F	D	D	D	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W				6									
s.d f4, v4(r1)						F	F	F	F	D	E	M	W																	
s.d f5, v5(r1)										F	D	E	M	W																
s.d f6, v6(r1)											F	D	D	D	D	E	M	W												
daddi r2, r2, -1												F	F	F	F	D	E	M	W											
bne r2, r5, cycle																F	D	E	M	W										
daddi r1, r1, -8																	F	D	E	M	W									
halt (i < 0)																		F	D	E	M	W								

CC: 54 if i>= 0, 55 if i<0: (54 * 31) + (55 * 1) = 1729

program_1_c.s :

daddui r1,r0,248	F	D	E	M	W																									
daddui r2,r0,248		F	D	E	M	W																								
daddui r3,r0,248			F	D	E	M	W																							
daddui r4,r0,31				F	D	E	M	W																						
ld r6, m(r0)					F	D	E	M	W																					
daddi r5,r0,-1						F	D	E	M	W																				
l.d f9, b(r0)							F	D	E	M	W																			
dmulu r6, r6, r2								F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W													
l.d f1, v1(r1)									F	D	E	M	W																	
l.d f2, v2(r1)										F	D	E	M	W																
l.d f3, v3(r1)											F	D	E	M	W															
l.d f11, v1(r2)												F	D	E	M	W														
mtc1 r6, f7													F	D	D	E	M	W												
cvt.d.l f7, f7														F	F	D	E	M	W											
mul.d f8, f1, f7																F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W					

l.d f12, v2(r2)																	F	D	E	M	W									
l.d f13, v3(r2)																		F	D	E	M	W								

mul.d f4, f8, f1	F	D	D	D	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W																	
cvt.l.d f10, f8		F	F	F	F	D	E	M	W																					
mfc1 r6, f10						F	D	E	M	W																				
sub.d f4, f4, f2							F	D	D	D	D	E_A	E_A	E_A	E_A	M	W													
div.d f5, f4, f3								F	F	F	F	D	D	D	D	E_D	...	...	x30	...	...	...	E_D	M	W					
dsllv r6, r6, r2												F	F	F	F	D	E	M	W											
mtc1 r6, f7																F	D	E	M	W										
cvt.d.l f7, f7																	F	D	E	M	W									

div.d f8, f11, f7	F	D	...	x27	...	D	E_D	...	...	...	...	x30	...	...	...	...	E_D	M	W											
sub.d f6, f4, f1		F	...	x27	...	F	D	E_A	E_A	E_A	E_A	M	W																	
sub.d f5, f5, f9							F	D	E_A	E_A	E_A	E_A	M	W																
mul.d f6, f6, f5								F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W													
s.d f4, v4(r1)									F	D	E	M	W																	
s.d f5, v5(r1)										F	D	E	M	W																
cvt.l.d f10, f8											F	D	D	D	D	D	D	E	M	W										
mfc1 r6, f10																	F	D	E	M	W									
mul.d f4, f8, f11																		F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W			
dmulu r6, r6, r2																			F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W		
s.d f6, v6(r1)																				F	D	E	M	W						
daddi r1, r1, -24																					F	D	E	M	W					

sub.d f4, f4, f12	F	D	D	D	E_A	E_A	E_A	E_A	M	W																				
div.d f5, f4, f13		F	F	F	D	D	D	D	E_D	...	...	...	...	...	...	x30	...	...	...	...	...	...	E_D	M	W					
mtc1 r6, f7					F	F	F	F	D	E	M	W																		
cvt.d.l f7, f7									F	D	E	M	W																	

mul.d f8, f1, f7										F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W											
l.d f1, v1(r3)											F	D	E	M	W															
l.d f2, v2(r3)												F	D	E	M	W														
l.d f3, v3(r3)													F	D	E	M	W													
cvt.l.d f10, f8														F	D	D	D	E	M	W										
mfc1 r6, f10															F	F	F	D	E	M	W									
mul.d f14, f8, f1																		F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W			

sub.d f14, f14, f2	F	D	D	D	D	D	D	E_A	E_A	E_A	E_A	M	W																	
div.d f15, f14, f3		F	F	F	F	F	F	D	...	x13	...	D	E_D	...	...	...	...	...	...	...	...	...	x30	...	...	...	...	...	...	...
sub.d f6, f4, f11								F	...	x13	...	F	D	E_A	E_A	E_A	E_A	M	W											
sub.d f5, f5, f9													F	D	E_A	E_A	E_A	E_A	M	W										
mul.d f6, f6, f5														F	D	D	D	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W				
s.d f4, v4(r2)															F	F	F	F	D	E	M	W								
s.d f5, v5(r2)																			F	D	E	M	W							
s.d f6, v6(r2)																				F	D	E	M	W						
daddi r2, r2, -24																					F	D	E	M	W					
sub.d f6, f14, f1																						F	D	E_A	E_A	E_A	E_A	M	W	

div.d f15, f14, f3	...	...	...	...	...	E_D	M	W																						
sub.d f15, f15, f9	F	D	...	x19	...	D	E_A	E_A	E_A	E_A	M	W																		
mul.d f6, f6, f15		F	...	x19	...	F	D	D	D	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W												
s.d f14, v4(r3)							F	F	F	F	D	E	M	W																
s.d f15, v5(r3)											F	D	E	M	W															
s.d f6, v6(r3)												F	D	D	D	D	E	M	W											
daddi r4, r4, -3													F	F	F	F	D	E	M	W										
bne r4, r5, cycle																	F	D	D	E	M	W								
daddi r3, r3, -24																		F	F	D	E	M	W							
halt (i< 0)																				F	D	E	M	W						

CC = 11 + (171 * 10) + (172 * 1) = 1893

CC **program_1.s** : initial part + cycle + no_multiple3 + multiple3 + end =
10 + 1364 + 378 + 407 + 1953 = 4112

CC **program_1_a.s** : initial part + cycle + no_multiple3 + multiple3 + end =
10 + 1256 + 357 + 407 + 1740 = 3770

CC **program_1_b.s** : initial part + cycle + no_multiple3 + multiple3 + end =
10 + 1248 + 357 + 385 + 1729 = 3729

CC **program_1_c.s** : 1893

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	4.707	4.584	4.581	2.881

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

Eventual explanation:

The results are quite similar comparing the ones obtained by hand and those obtained by simulation. The small difference are due to the different way we handle hazards with respect to how the simulator does: we stall every instruction always in D stage, while the simulator when it's possible stalls instructions once they reach EX stage.

For what concerns the difference in CC between the various programs, we see that every time we create a new program enhancing the one before, the CC are reduced, so the enhancements are effective!

Another thing I think is relevant is that, while the first three programs took around the same number of CCs (4000), the last program drastically reduced them to around 1800. This is because in the last case, since 3 loops were unrolled, the computation of *i mod 3* was not needed anymore and being it one of the most impacting computations, this allowed this very important enhancement.