Computer Architectures
Exam of 07.2.2023 - part I

First name, Last name, ID.....

Version A - Question #2

Let's consider a MIPS64 pipelined architecture including the following functional units (for each unit the number of clock periods to complete one instruction is reported):

- Integer ALU and Data memory: 1 clock period;
- FP arithmetic unit: 2 clock periods (pipelined);
- FP multiplier unit: 3 clock periods (pipelined);
- FP divider unit: 6 clock periods (unpipelined);

You should also assume that:

- The branch delay slot corresponds to 1 clock cycle, and the branch delay slot is not enabled;
- Data forwarding is enabled;
- The EXE phase can be completed out-of-order.

You should consider the following code fragment and, filling the following tables, determine the pipeline behavior in each clock period, as well as the total number of clock periods required to run it.

```
; ********************** C ***************************
;  for (i = 0; i < 30; i++) {
     v5[i] = (v1[i]/v2[i])*(v3[i]/v4[i])-v2[i]+v4[i];
;  }
; ********************** MIPS64 ***********************
```

```
            .data
v1:     .double “30 values”
v2:     .double “30 values”
v3:     .double “30 values”
v4:     .double “30 values”
v5:     .double “30 values”

            .text
main:  daddui r1,r0,0
       daddui r2,r0,30
loop:  l.d f1,v1(r1)
       l.d f2,v2(r1)
       div.d f5,f1,f2
       l.d f3,v3(r1)
       l.d f4,v4(r1)
       div.d f6,f3,f4
       sub.d f7,f7,f2
       add.d f7,f7,f4
       mul.d f5,f5,f6
       add.d f5,f5,f7
       s.d f5,v5(r2)
       daddi r2,r2,-1
       daddui r1,r1,8
       bnez r2,loop
halt
```

Comments	Sol #1	Sol #2	Sol #3	Sol #4
r1 ← pointer	5	5	5	5
r2 ← 30	1	1	1	1
f1 ← v1[i]	1	1	1	1
f2 ← v2[i]	1	1	1	1
f5 ← v1[i] / v2[i]	7	7	7	7
f3 ← v3[i]	0	0	0	0
f4 ← v4[i]	0	0	0	0
f6 ← v3[i] / v4[i]	6	6	6	6
f7 ← –v2[i]	0	0	0	0
f7 ← –v2[i] + v4[i]	0	0	0	0
f5 ← f5 * f6	3	3	3	3
f5 ← f5 + f7	2	2	2	2
v5[i] ← f5	1	1	1	1
r2 ← r2 – 1	1	1	1	1
r1 ← r1 + 8	1	1	1	1
	1	1	1	1
	1	1	1	1
Total:	6 + 25*30 = 756	6 + 25*30 = 756	6 + 25*30 = 756	6 + 25*30 = 756

Computer Architectures
Exam of 07.2.2023 - part I

First name, Last name, ID.....

Solution #1

main:	daddui r1,r0,0	F	D	E	M	W																													5
	daddui r2,r0,30		F	D	E	M	W																													1
loop:	l.d f1,v1(r1)			F	D	E	M	W																												1
	l.d f2,v2(r1)				F	D	E	M	W																											1
	div.d f5,f1,f2					F	D	D	E_D	E_D	E_D	E_D	E_D	E_D	M	W																				7
	l.d f3,v3(r1)						F	F	D	E	M	W																								0
	l.d f4,v4(r1)								F	D	E	M	W																							0
	div.d f6,f3,f4									F	D	D	D	D	E_D	E_D	E_D	E_D	E_D	E_D	M	W														6
	sub.d f7,f7,f2										F	F	F	F	D	E_A	E_A	M	W																	0
	add.d f7,f7,f4														F	D	D	E_A	E_A	M	W															0
	mul.d f5,f5,f6															F	F	D	D	D	E_M	E_M	E_M	M	W										3
	add.d f5,f5,f7																	F	F	F	D	D	D	E_A	E_A	M	W								2
	s.d f5,v5(r2)																				F	F	F	D	D	E	M	W							1
	daddi r2,r2,-1																							F	F	D	E	M	W						1
	daddui r1,r1,8																									F	D	E	M	W					1
	bnez r2,loop																										F	D	E	M	W				1
halt																												F	D	E	M	W				1

In this solution, the MEM stage can be accessed by multiple instructions simultaneously if only one of them accesses memory.

Solution #2

main:	daddui r1,r0,0	F	D	E	M	W																														5
	daddui r2,r0,30		F	D	E	M	W																													1
loop:	l.d f1,v1(r1)			F	D	E	M	W																												1
	l.d f2,v2(r1)				F	D	E	M	W																											1
	div.d f5,f1,f2					F	D	D	E_D	E_D	E_D	E_D	E_D	E_D	M	W																				7
	l.d f3,v3(r1)						F	F	D	E	M	W																								0
	l.d f4,v4(r1)								F	D	E	M	W																							0
	div.d f6,f3,f4									F	D	D	D	D	E_D	E_D	E_D	E_D	E_D	E_D	M	W														6
	sub.d f7,f7,f2										F	F	F	F	D	E_A	E_A	M	W																	0
	add.d f7,f7,f4														F	D	D	E_A	E_A	M	W															0
	mul.d f5,f5,f6															F	F	D	D	D	E_M	E_M	E_M	M	W										3
	add.d f5,f5,f7																	F	F	F	D	D	D	E_A	E_A	M	W								2
	s.d f5,v5(r2)																				F	F	F	D	D	E	M	W							1
	daddi r2,r2,-1																							F	F	D	E	M	W						1
	daddui r1,r1,8																									F	D	E	M	W					1
	bnez r2,loop																										F	D	E	M	W				1
halt																												F	D	E	M	W				1

In this solution, the MEM stage is accessed by a single instruction at a time. In this case, the solution is equivalent to the one above since this situation never happens.

Computer Architectures
Exam of 07.2.2023 - part I

First name, Last name, ID.....

Solution #3 (WinMIPS64-style)

main:	daddui r1,r0,0	F	D	E	M	W																											5
	daddui r2,r0,30		F	D	E	M	W																											1
loop:	l.d f1,v1(r1)			F	D	E	M	W																										1
	l.d f2,v2(r1)				F	D	E	M	W																									1
	div.d f5,f1,f2					F	D	E_D	E_D	E_D	E_D	E_D	E_D	E_D	M	W																		7
	l.d f3,v3(r1)						F	D	E	M	W																						0
	l.d f4,v4(r1)							F	D	E	M	W																					0
	div.d f6,f3,f4								F	D	D	D	D	D	E_D	E_D	E_D	E_D	E_D	E_D	M	W											6
	sub.d f7,f7,f2									F	F	F	F	F	D	E_A	E_A	M	W														0
	add.d f7,f7,f4														F	D	E_A	E_A	E_A	M	W												0
	mul.d f5,f5,f6															F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W								3
	add.d f5,f5,f7																F	D	E_A	E_A	E_A	E_A	E_A	E_A	E_A	M	W						2
	s.d f5,v5(r2)																	F	D	E	E	E	E	E	E	E	M	W						1
	daddi r2,r2,-1																		F	D	D	D	D	D	D	D	E	M	W					1
	daddui r1,r1,8																			F	F	F	F	F	F	F	D	E	M	W				1
	bnez r2,loop																										F	D	E	M	W				1
halt																												F	D	E	M	W				1

The operations are anticipated as soon as possible, before the operands are actually available. The MEM stage is accessed by a single instruction at a time.

Solution #4 (WinMIPS64-adapted)

main:	daddui r1,r0,0	F	D	E	M	W																											5
	daddui r2,r0,30		F	D	E	M	W																											1
loop:	l.d f1,v1(r1)			F	D	E	M	W																										1
	l.d f2,v2(r1)				F	D	E	M	W																									1
	div.d f5,f1,f2					F	D	E_D	E_D	E_D	E_D	E_D	E_D	E_D	M	W																		7
	l.d f3,v3(r1)						F	D	E	M	W																						0
	l.d f4,v4(r1)							F	D	E	M	W																					0
	div.d f6,f3,f4								F	D	D	D	D	D	E_D	E_D	E_D	E_D	E_D	E_D	M	W											6
	sub.d f7,f7,f2									F	F	F	F	F	D	E_A	E_A	M	W														0
	add.d f7,f7,f4														F	D	E_A	E_A	E_A	M	W												0
	mul.d f5,f5,f6															F	D	E_M	E_M	E_M	E_M	E_M	E_M	M	W								3
	add.d f5,f5,f7																F	D	D	D	D	E_A	E_A	E_A	E_A	M	W						2
	s.d f5,v5(r2)																	F	F	F	F	D	D	D	E	E	M	W						1
	daddi r2,r2,-1																					F	F	F	D	D	E	M	W					1
	daddui r1,r1,8																								F	F	D	E	M	W				1
	bnez r2,loop																										F	D	E	M	W				1
halt																												F	D	E	M	W				1

WinMIPS results have been taken and adapted to the case at hand. Something may differ from what was seen during the lessons and labs due to the divergences of WinMIPS. The given code was not fully simulable due to the shorter FP divider unit stage.

Any additional solutions will be evaluated on a case-by-case basis.

Computer Architectures
Exam of 07.2.2023 - part I

First name, Last name, ID.....

Version B - Question #2

Let's consider a MIPS64 pipelined architecture including the following functional units (for each unit the number of clock periods to complete one instruction is reported):

- Integer ALU and Data memory: 1 clock period;
- FP arithmetic unit: 2 clock periods (pipelined);
- FP multiplier unit: 3 clock periods (pipelined);
- FP divider unit: 6 clock periods (unpipelined);

You should also assume that:

- The branch delay slot corresponds to 1 clock cycle, and the branch delay slot is not enabled;
- Data forwarding is enabled;
- The EXE phase can be completed out-of-order.

You should consider the following code fragment and, filling the following tables, determine the pipeline behavior in each clock period, as well as the total number of clock periods required to run it.

```
; ********************** C ************************
;  for (i = 0; i < 20; i++) {
     v5[i] = v2[i]-v4[i]+(v1[i]*v2[i])/(v3[i]*v4[i]);
;  }
; ********************** MIPS64 ***********************
```

		Comments	Sol #1	Sol #2	Sol #3	Sol #4
	.data					
v1:	.double “20 values”					
v2:	.double “20 values”					
v3:	.double “20 values”					
v4:	.double “20 values”					
v5:	.double “20 values”					
	.text					
main:	daddui r1,r0,0	r1 ← pointer	5	5	5	5
	daddui r2,r0,20	r2 ← 20	1	1	1	1
loop:	l.d f1,v1(r1)	f1 ← v1[i]	1	1	1	1
	l.d f2,v2(r1)	f2 ← v2[i]	1	1	1	1
	mul.d f5,f1,f2	f5 ← v1[i] * v2[i]	4	4	4	4
	add.d f7,f7,f2	f7 ← v2[i]	1	1	0	0
	l.d f3,v3(r1)	f3 ← v3[i]	1	1	1	1
	l.d f4,v4(r1)	f4 ← v4[i]	1	1	1	1
	mul.d f6,f3,f4	f6 ← v3[i] * v4[i]	4	4	4	4
	sub.d f7,f7,f4	f7 ← v2[i] – v4[i]	1	1	0	0
	div.d f5,f5,f6	f5 ← f5 / f6	5	5	6	6
	add.d f5,f5,f7	f5 ← f5 + f7	2	2	2	2
	s.d f5,v5(r2)	v5[i] ← f5	1	1	1	1
	daddi r2,r2,-1	r2 ← r2 – 1	1	1	1	1
	daddui r1,r1,8	r1 ← r1 + 8	1	1	1	1
	bnez r2,loop		1	1	1	1
halt			1	1	1	1
	Total:		6 + 26*20 = 526	6 + 26*20 = 526	6 + 25*20 = 506	6 + 25*20 = 506

Computer Architectures
Exam of 07.2.2023 - part I

First name, Last name, ID.....

Solution #1

main:	daddui r1,r0,0	F	D	E	M	W																													5
	daddui r2,r0,20		F	D	E	M	W																													1
loop:	l.d f1,v1(r1)			F	D	E	M	W																												1
	l.d f2,v2(r1)				F	D	E	M	W																											1
	mul.d f5,f1,f2					F	D	D	E_M	E_M	E_M	M	W																							4
	add.d f7,f7,f2						F	F	D	E_A	E_A	M	M	W																							1
	l.d f3,v3(r1)								F	D	E	M	M	M	W																					1
	l.d f4,v4(r1)									F	D	E	E	E	M	W																				1
	mul.d f6,f3,f4										F	D	D	D	D	E_M	E_M	E_M	M	W																4
	sub.d f7,f7,f4											F	F	F	F	D	E_A	E_A	M	M	W															1
	div.d f5,f5,f6															F	D	D	E_D	E_D	E_D	E_D	E_D	E_D	M	W										5
	add.d f5,f5,f7																F	F	D	D	D	D	D	D	E_A	E_A	M	W								2
	s.d f5,v5(r2)																		F	F	F	F	F	F	D	D	E	M	W							1
	daddi r2,r2,-1																								F	F	D	E	M	W						1
	daddui r1,r1,8																										F	D	E	M	W					1
	bnez r2,loop																											F	D	E	M	W				1
halt																													F	D	E	M	W				1

In this solution, the MEM stage can be accessed by multiple instructions simultaneously if only one of them accesses memory.

Solution #2

main:	daddui r1,r0,0	F	D	E	M	W																														5
	daddui r2,r0,20		F	D	E	M	W																													1
loop:	l.d f1,v1(r1)			F	D	E	M	W																												1
	l.d f2,v2(r1)				F	D	E	M	W																											1
	mul.d f5,f1,f2					F	D	D	E_M	E_M	E_M	M	W																							4
	add.d f7,f7,f2						F	F	D	E_A	E_A	E_A	M	W																							1
	l.d f3,v3(r1)								F	D	E	E	E	M	W																					1
	l.d f4,v4(r1)									F	D	D	D	E	M	W																				1
	mul.d f6,f3,f4										F	F	F	D	D	E_M	E_M	E_M	M	W																4
	sub.d f7,f7,f4													F	F	D	E_A	E_A	E_A	M	W															1
	div.d f5,f5,f6															F	D	D	E_D	E_D	E_D	E_D	E_D	E_D	M	W										5
	add.d f5,f5,f7																F	F	D	D	D	D	D	D	E_A	E_A	M	W								2
	s.d f5,v5(r2)																		F	F	F	F	F	F	D	D	E	M	W							1
	daddi r2,r2,-1																								F	F	D	E	M	W						1
	daddui r1,r1,8																										F	D	E	M	W					1
	bnez r2,loop																											F	D	E	M	W				1
halt																													F	D	E	M	W				1

In this solution, the MEM stage is accessed by a single instruction at a time.

Computer Architectures
Exam of 07.2.2023 - part I

First name, Last name, ID.....

Solution #3 (WinMIPS64-style)

main:	daddui r1,r0,0	F	D	E	M	W																													5
	daddui r2,r0,20		F	D	E	M	W																													1
loop:	l.d f1,v1(r1)			F	D	E	M	W																												1
	l.d f2,v2(r1)				F	D	E	M	W																											1
	mul.d f5,f1,f2					F	D	E_M	E_M	E_M	E_M	M	W																							4
	add.d f7,f7,f2						F	D	E_A	E_A	M	W																								0
	l.d f3,v3(r1)							F	D	E	E	E	M	W																					1
	l.d f4,v4(r1)								F	D	D	D	E	M	W																				1
	mul.d f6,f3,f4									F	F	F	D	E_M	E_M	E_M	E_M	M	W																4
	sub.d f7,f7,f4												F	D	E_A	E_A	M	W																	0
	div.d f5,f5,f6													F	D	E_D	E_D	E_D	E_D	E_D	E_D	E_D	E_D	M	W									6
	add.d f5,f5,f7														F	D	E_A	E_A	E_A	E_A	E_A	E_A	E_A	E_A	E_A	M	W							2
	s.d f5,v5(r2)															F	D	E	E	E	E	E	E	E	E	M	W								1
	daddi r2,r2,-1																F	D	D	D	D	D	D	D	D	E	M	W							1
	daddui r1,r1,8																	F	F	F	F	F	F	F	F	D	E	M	W						1
	bnez r2,loop																									F	D	E	M	W					1
halt																											F	D	E	M	W				1

The operations are anticipated as soon as possible, before the operands are actually available. The MEM stage is accessed by a single instruction at a time.

Solution #4 (WinMIPS64-adapted)

main:	daddui r1,r0,0	F	D	E	M	W																														5
	daddui r2,r0,20		F	D	E	M	W																													1
loop:	l.d f1,v1(r1)			F	D	E	M	W																												1
	l.d f2,v2(r1)				F	D	E	M	W																											1
	mul.d f5,f1,f2					F	D	E_M	E_M	E_M	E_M	M	W																							4
	add.d f7,f7,f2						F	D	E_A	E_A	M	W																								0
	l.d f3,v3(r1)							F	D	E	E	E	M	W																					1
	l.d f4,v4(r1)								F	D	D	D	E	M	W																				1
	mul.d f6,f3,f4									F	F	F	D	E_M	E_M	E_M	E_M	M	W																4
	sub.d f7,f7,f4												F	D	E_A	E_A	M	W																	0
	div.d f5,f5,f6													F	D	E_D	E_D	E_D	E_D	E_D	E_D	E_D	E_D	M	W									6
	add.d f5,f5,f7														F	D	D	D	E_A	E_A	E_A	E_A	E_A	E_A	E_A	M	W							2
	s.d f5,v5(r2)															F	F	F	D	D	D	D	D	D	E	E	M	W							1
	daddi r2,r2,-1																		F	F	F	F	F	F	D	D	E	M	W						1
	daddui r1,r1,8																								F	F	D	E	M	W					1
	bnez r2,loop																										F	D	E	M	W				1
halt																												F	D	E	M	W			1

WinMIPS results have been taken and adapted to the case at hand. Something may differ from what was seen during the lessons and labs due to the divergences of WinMIPS. The given code was not fully simulable due to the shorter FP divider unit stage.

Any additional solutions will be evaluated on a case-by-case basis.