Computer Architectures
Exam of 24.2.2023 - 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;
- Memory Access (MEM stage) for Load/Store instructions: 2 clock periods;
- FP Arithmetic Unit: 2 clock periods (pipelined);
- FP Multiplier Unit: 4 clock periods (pipelined);

You should also assume that:

- The branch delay slot corresponds to 1 clock cycles, 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++) {
     v6[i] = - v4[i] + v5[i] + (v1[i] * v2[i]) * v3[i+1];
;  }
; ********************** MIPS64 ***********************
```

```
            .data
v1:    .double “30 values”
v2:    .double “30 values”
v3:    .double “31 values”
v4:    .double “30 values”
v5:    .double “30 values”
v6:    .double “30 values”

            .text
main: daddui r1,r0,0
      daddui r2,r0,30
loop: l.d f1,v1(r1)
      l.d f2,v2(r1)
      mul.d f6,f1,f2
      daddui r3,r1,8
      l.d f3,v3(r3)
      mul.d f6,f6,f3
      l.d f4,v4(r1)
      sub.d f6,f6,f4
      l.d  f5,v5(r1)
      add.d f6,f6,f5
      s.d f6,v6(r2)
      daddui r1,r1,8
      daddi r2,r2,-1
      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]	2	2	2	2
f2 ← v2[i]	2	2	2	2
f6 ← v1[i] * v2[i]	5	6	6	6
r3 ← r1 + 8	0	0	0	0
f3 ← v3[i+1]	1	0	0	0
f6 ← f6 * v3[i+1]	4	4	4	4
f4 ← v4[i]	0	0	0	0
f6 ← f6 – v4[i]	2	2	1	1
f5 ← v5[i]	1	2	0	0
f6 ← f6 + v5[i]	3	3	2	2
v6[i] ← f6	2	2	2	2
r1 ← r1 + 8	1	1	1	1
r2 ← r2 – 1	1	1	1	1
	1	2	2	2
	1	1	1	1
Total:	6 + 26*30 = 786	6 + 28*30 = 846	6 + 24*30 = 726	6 + 24*30 = 726

Computer Architectures
Exam of 24.2.2023 - 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	M	W																										2
l.d f2,v2(r1)				F	D	E	E	M	M	W																								2
mul.d f6,f1,f2					F	D	D	D	D	E_M	E_M	E_M	E_M	M	W																			5
daddui r3,r1,8						F	F	F	F	D	E	M	W																					0
l.d f3,v3(r3)									F	D	E	M	M	M	W																			1
mul.d f6,f6,f3										F	D	D	D	E_M	E_M	E_M	E_M	M	W															4
l.d f4,v4(r1)											F	F	F	D	E	M	M	W																0
sub.d f6,f6,f4														F	D	D	D	E_A	E_A	M	W													2
l.d f5,v5(r1)															F	F	F	D	E	M	M	W												1
add.d f6,f6,f5																		F	D	D	D	E_A	E_A	M	W									3
s.d f6,v6(r2)																			F	F	F	D	D	E	M	M	W							2
daddui r1,r1,8																						F	F	D	E	M	M	W						1
daddi r2,r2,-1																								F	D	E	M	M	W					1
bnez r2,loop																									F	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	M	W																										2
l.d f2,v2(r1)				F	D	E	E	M	M	W																								2
mul.d f6,f1,f2					F	D	D	D	D	E_M	E_M	E_M	E_M	E_M	M	W																		6
daddui r3,r1,8						F	F	F	F	D	E	M	W																					0
l.d f3,v3(r3)									F	D	E	M	M	W																				0
mul.d f6,f6,f3										F	D	D	D	E_M	E_M	E_M	E_M	M	W															4
l.d f4,v4(r1)											F	F	F	D	E	M	M	W																0
sub.d f6,f6,f4														F	D	D	D	E_A	E_A	M	W													2
l.d f5,v5(r1)															F	F	F	D	E	E	M	M	W											2
add.d f6,f6,f5																		F	D	D	D	D	E_A	E_A	M	W								3
s.d f6,v6(r2)																			F	F	F	F	D	D	E	M	M	W						2
daddui r1,r1,8																							F	F	D	E	E	M	W					1
daddi r2,r2,-1																									F	D	D	E	M	W				1
bnez r2,loop																										F	F	F	D	E	M	W		2
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 24.2.2023 - 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	M	W																											2
l.d f2,v2(r1)				F	D	E	E	M	M	W																									2
mul.d f6,f1,f2					F	D	E_M	E_M	E_M	E_M	E_M	E_M	E_M	E_M	M	W																				6
daddui r3,r1,8						F	D	E	M	W																									0
l.d f3,v3(r3)							F	D	E	M	M	W																							0
mul.d f6,f6,f3								F	D	D	E_M	E_M	E_M	E_M	E_M	E_M	E_M	M	W																4
l.d f4,v4(r1)									F	F	D	E	M	M	W																				0
sub.d f6,f6,f4											F	D	E_A	E_A	E_A	E_A	E_A	E_A	E_A	M	W															1
l.d f5,v5(r1)												F	D	E	E	M	M	W																	0
add.d f6,f6,f5													F	D	D	D	D	D	E_A	E_A	E_A	M	W												2
s.d f6,v6(r2)														F	F	F	F	F	D	E	E	E	M	M	W										2
daddui r1,r1,8																			F	D	D	D	E	E	M	W									1
daddi r2,r2,-1																				F	F	F	D	D	E	M	W								1
bnez r2,loop																							F	F	D	D	E	M	W						2
halt																									F	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	M	W																											2
l.d f2,v2(r1)				F	D	E	E	M	M	W																									2
mul.d f6,f1,f2					F	D	D	E_M	E_M	E_M	E_M	E_M	E_M	E_M	M	W																				6
daddui r3,r1,8						F	F	D	E	M	W																								0
l.d f3,v3(r3)								F	D	E	M	M	W																						0
mul.d f6,f6,f3									F	D	E_M	E_M	E_M	E_M	E_M	E_M	E_M	E_M	M	W															4
l.d f4,v4(r1)										F	D	E	M	M	W																				0
sub.d f6,f6,f4											F	D	D	D	E_A	E_A	E_A	E_A	E_A	E_A	M	W													1
l.d f5,v5(r1)												F	F	F	D	E	M	M	W																0
add.d f6,f6,f5															F	D	D	D	E_A	E_A	E_A	M	W												2
s.d f6,v6(r2)																F	F	F	D	D	E	E	M	M	W										2
daddui r1,r1,8																			F	F	D	D	E	E	M	W									1
daddi r2,r2,-1																				F	F	D	D	E	M	W									1
bnez r2,loop																							F	F	D	D	E	M	W						2
halt																									F	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 24.2.2023 - 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;
- Memory Access (MEM stage) for Load/Store instructions: 2 clock periods;
- FP Arithmetic Unit: 2 clock periods (pipelined);
- FP Multiplier Unit: 4 clock periods (pipelined);

You should also assume that:

- The branch delay slot corresponds to 1 clock cycles, 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++) {
     v6[i] = (v1[i] * v2[i]) * v3[i] + v4[i] - v5[i+1];
;  }
; ********************** MIPS64 ***********************
```

```
            .data
v1:    .double “20 values”
v2:    .double “20 values”
v3:    .double “20 values”
v4:    .double “20 values”
v5:    .double “21 values”
v6:    .double “20 values”

            .text
main: daddui r1,r0,0
      daddui r2,r0,20
loop: l.d f1,v1(r1)
      l.d f2,v2(r1)
      l.d f3,v3(r1)
      l.d f4,v4(r1)
      daddui r3,r1,8
      l.d f5,v5(r3)
      mul.d f6,f1,f2
      mul.d f6,f6,f3
      add.d f6,f6,f4
      sub.d f6,f6,f5
      s.d f6,v6(r2)
      daddui r1,r1,8
      daddi r2,r2,-1
      bnez r2,loop
halt
```

Comments	Sol #1	Sol #2	Sol #3	Sol #4
r1 ← pointer	5	5	5	5
r2 ← 20	1	1	1	1
f1 ← v1[i]	2	2	2	2
f2 ← v2[i]	2	2	2	2
f3 ← v3[i]	2	2	2	2
f4 ← v4[i]	2	2	2	2
r3 ← r1 + 8	1	1	1	1
f5 ← v5[i+1]	1	2	2	2
f6 ← v1[i] * v2[i]	3	3	3	3
f6 ← f6 * v3[i]	4	4	4	4
f6 ← f6 + v4[i]	2	2	2	2
f6 ← f6 – v5[i]	2	2	2	2
v6[i] ← f6	2	2	2	2
r1 ← r1 + 8	1	1	1	0
r2 ← r2 – 1	1	1	1	1
	1	2	2	2
	1	1	1	1
Total:	6 + 27*20 = 546	6 + 29*20 = 586	6 + 29*20 = 586	6 + 29*20 = 586

Computer Architectures
Exam of 24.2.2023 - 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	M	W																											2
l.d f2,v2(r1)				F	D	E	E	M	M	W																									2
l.d f3,v3(r1)					F	D	D	E	E	M	M	W																							2
l.d f4,v4(r1)						F	F	D	D	E	E	M	M	W																					2
daddui r3,r1,8								F	F	D	D	E	M	M	W																				1
l.d f5,v5(r3)										F	F	D	E_L	M	M	W																			1
mul.d f6,f1,f2												F	D	E_M	E_M	E_M	E_M	M	W																3
mul.d f6,f6,f3													F	D	D	D	D	E_M	E_M	E_M	E_M	M	W												4
add.d f6,f6,f4														F	F	F	F	D	D	D	D	E_A	E_A	M	W										2
sub.d f6,f6,f5																		F	F	F	F	D	D	E_A	E_A	M	W								2
s.d f6,v6(r2)																						F	F	D	D	E	M	M	W						2
daddui r1,r1,8																								F	F	D	E	M	M	W					1
daddi r2,r2,-1																										F	D	E	M	M	W				1
bnez r2,loop																											F	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	M	W																											2
l.d f2,v2(r1)				F	D	E	E	M	M	W																									2
l.d f3,v3(r1)					F	D	D	E	E	M	M	W																							2
l.d f4,v4(r1)						F	F	D	D	E	E	M	M	W																					2
daddui r3,r1,8								F	F	D	D	E	E	M	W																				1
l.d f5,v5(r3)										F	F	D	D	E	M	M	W																		2
mul.d f6,f1,f2												F	F	D	E_M	E_M	E_M	E_M	M	W																3
mul.d f6,f6,f3														F	D	D	D	D	E_M	E_M	E_M	E_M	M	W											4
add.d f6,f6,f4															F	F	F	F	D	D	D	D	E_A	E_A	M	W									2
sub.d f6,f6,f5																			F	F	F	F	D	D	E_A	E_A	M	W							2
s.d f6,v6(r2)																							F	F	D	D	E	M	M	W						2
daddui r1,r1,8																									F	F	D	E	E	M	W					1
daddi r2,r2,-1																											F	D	D	E	M	W				1
bnez r2,loop																												F	F	F	D	E	M	W		2
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 24.2.2023 - 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	M	W																											2
l.d f2,v2(r1)				F	D	E	E	M	M	W																									2
l.d f3,v3(r1)					F	D	D	E	E	M	M	W																							2
l.d f4,v4(r1)						F	F	D	D	E	E	M	M	W																					2
daddui r3,r1,8								F	F	D	D	E	E	M	W																				1
l.d f5,v5(r3)										F	F	D	D	E	M	M	W																		2
mul.d f6,f1,f2												F	F	D	E_M	E_M	E_M	E_M	M	W																3
mul.d f6,f6,f3														F	D	E_M	E_M	E_M	E_M	E_M	E_M	E_M	E_M	M	W										4
add.d f6,f6,f4															F	D	E_F	E_F	E_F	E_F	E_F	E_F	E_F	E_F	M	W									2
sub.d f6,f6,f5																F	D	D	D	D	D	D	D	E_F	E_F	E_F	M	W							2
s.d f6,v6(r2)																	F	F	F	F	F	F	F	D	D	E	E	M	M	W					2
daddui r1,r1,8																								F	F	D	D	E	E	M	W				1
daddi r2,r2,-1																										F	F	D	D	E	M	W			1
bnez r2,loop																												F	F	D	D	E	M	W	2
halt																														F	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	M	W																											2
l.d f2,v2(r1)				F	D	E	E	M	M	W																									2
l.d f3,v3(r1)					F	D	D	E	E	M	M	W																							2
l.d f4,v4(r1)						F	F	D	D	E	E	M	M	W																					2
daddui r3,r1,8								F	F	D	D	E	E	M	W																				1
l.d f5,v5(r3)										F	F	D	D	E	M	M	W																		2
mul.d f6,f1,f2												F	F	D	E_M	E_M	E_M	E_M	M	W																3
mul.d f6,f6,f3														F	D	E_M	E_M	E_M	E_M	E_M	E_M	E_M	E_M	M	W										4
add.d f6,f6,f4															F	D	D	D	D	E_F	E_F	E_F	E_F	E_F	M	W									2
sub.d f6,f6,f5																F	F	F	F	D	D	D	D	E_F	E_F	E_F	M	W							2
s.d f6,v6(r2)																				F	F	F	F	D	D	E	E	M	M	W					2
daddui r1,r1,8																								F	F	D	D	E	E	M	W				0
daddi r2,r2,-1																										F	F	D	D	E	M	W			1
bnez r2,loop																												F	F	D	D	E	M	W	2
halt																														F	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.