31 January 2012 -- Computer Architectures –- part 2/2

Name, Matricola .....

Question 1

Considering the MIPS64 architecture presented in the following:

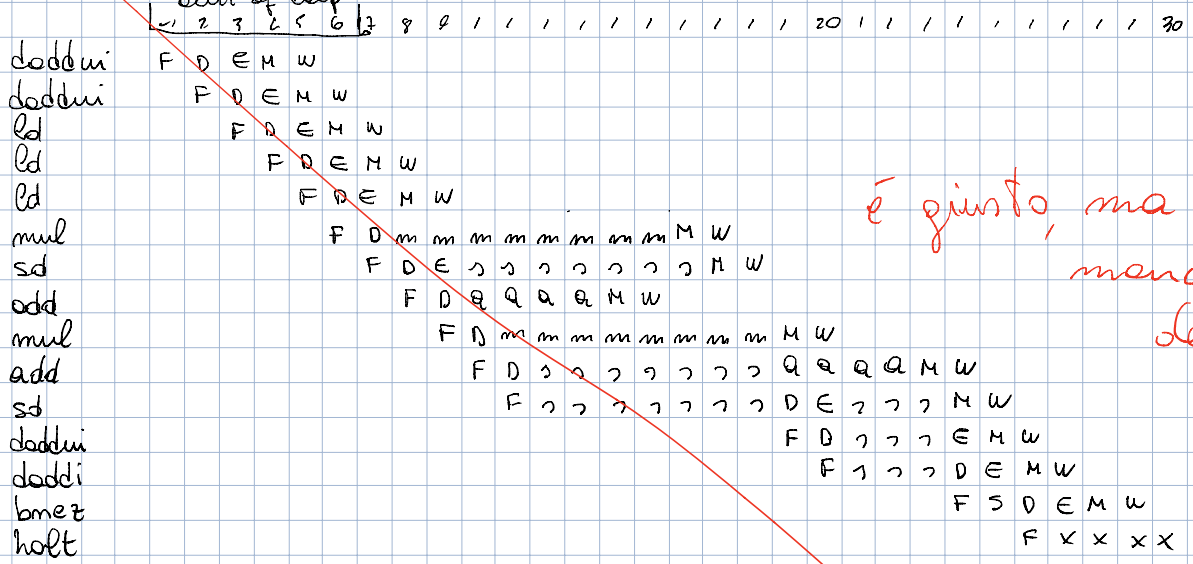
- Integer ALU: 1 clock cycle
- Data memory: 1 clock cycle
- FP multiplier unit: pipelined 8 stages
- FP arithmetic unit: pipelined 4 stages
- FP divider unit: not pipelined unit that requires 10 clock cycles
- branch delay slot: 1 clock cycle, and the branch delay slot is not enable
- forwarding is enabled
- it is possible to complete instruction EXE stage in an out-of-order fashion.

o and using the following code fragment, show the timing of the presented loop-based program and compute how many cycles does this program take to execute?

```
; ********************* MIPS64 ***********************
;     for (i = 0; i < 100; i++) {
;          v4[i] = v1[i]*v2[i];
;          v5[i] = v1[i]+v2[i]+(v1[i]*v3[i]);
;}
```

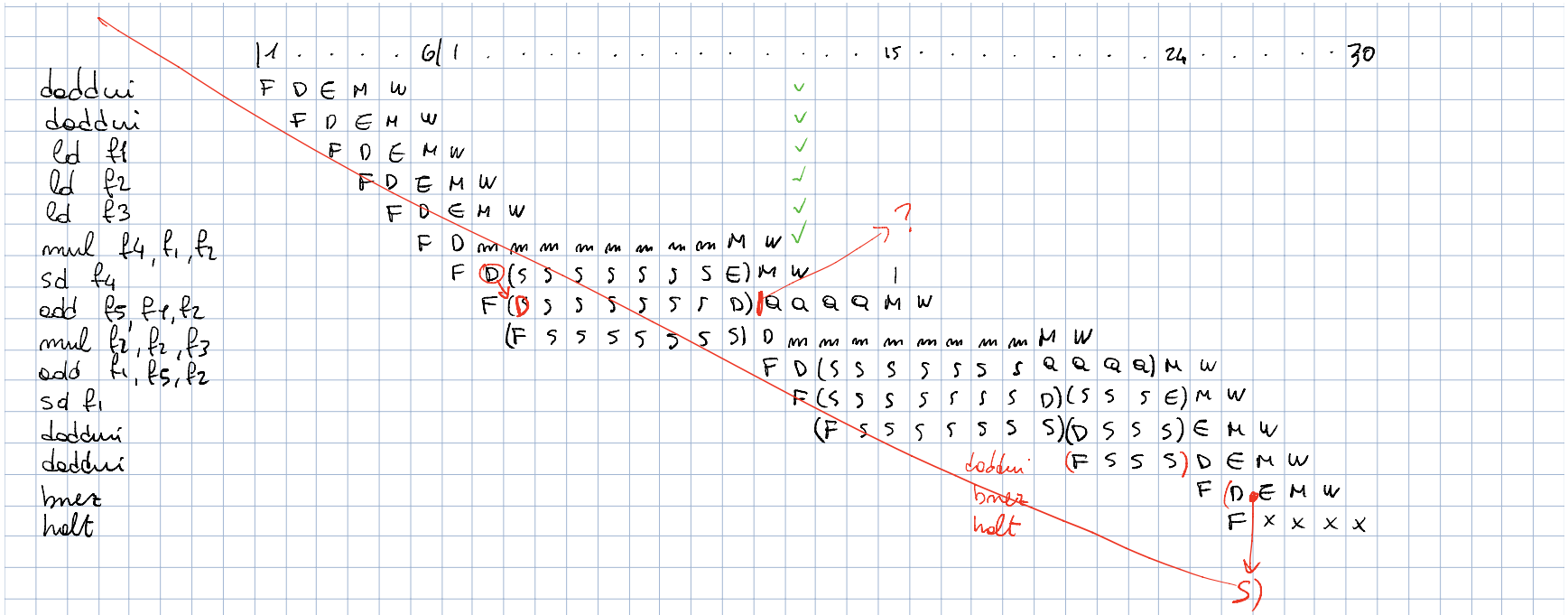
		comments	Clock cycles
	.data		
V1:	.double “100 values”		
V2:	.double “100 values”		
V3:	.double “100 values”		
V4:	.double “100 zeros”		
V5:	.double “100 zeros”		
	.text		
main:	daddui r1,r0,0	r1← pointer	5
	daddui r2,r0,100	r2 <= 100	1
loop:	l.d f1,v1(r1)	f1 <= v1[i]	1
	l.d f2,v2(r1)	f2 <= v2[i]	1
	l.d f3,v3(r1)	f3 <= v3[i]	1
	mul.d f4,f1,f2	f4 ← v1[i]*v2[i]	8
	s.d f4,v4(r1)	v4[i] ← f4	1
	add.d f5,f1,f2	f5 ← v1[i]+v2[i]	0
	mul.d f2,f1,f3	f2 ← v1[i]*v3[i]	2
	add.d f1,f5,f2	f1 ← v1[i]+v2[i]+v1[i]*v3[i]	4
	s.d f1,v5(r1)	v5[i] ← f1	1
	daddui r1,r1,8	r1 ← r1 + 8	1
	daddi r2,r2,-1	r2 ← r2 - 1	1
	bnez r2,loop		2
	Halt		1
Total			2406

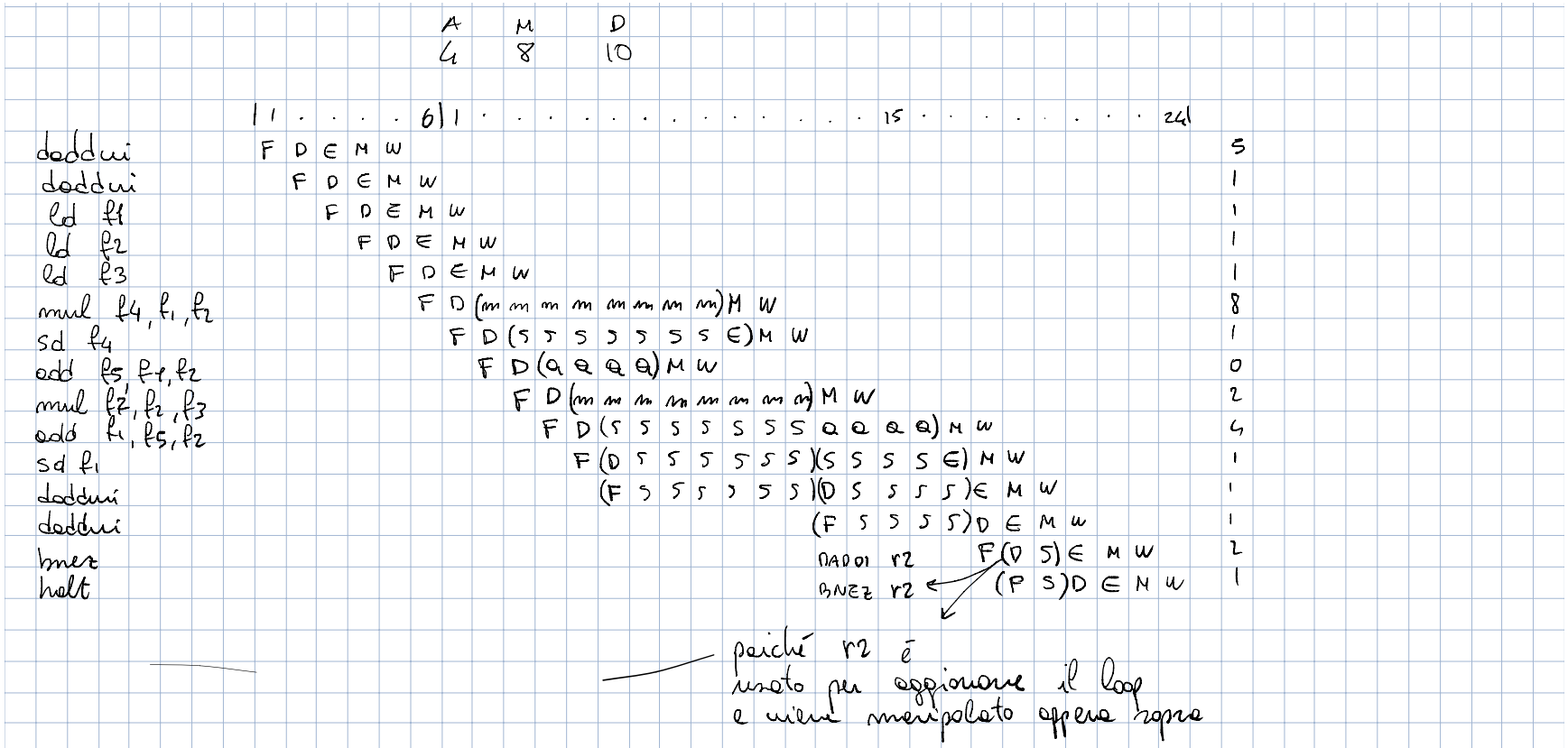
Exam 31 January 2012



$[(30 - 6) \cdot 100] + 6 = (24 \cdot 100) + 6 = 2406$

è giusto, ma concettualmente manca delle precisione degli stalli


1 6 15 24 30
daddui
daddui
ld f1
ld f2
ld f3
mul f4, f1, f2
sd f4
add f5, f1, f2
mul f2, f2, f3
add f1, f5, f2
sd f1
daddui
daddui
bnez
halt
?
daddui
bnez
halt
S)


A M D
4 8 10
daddui
daddui
ld f1
ld f2
ld f3
mul f4, f1, f2
sd f4
add f5, f1, f2
mul f2, f2, f3
add f1, f5, f2
sd f1
daddui
daddui
bnez
halt
NADDI r2
BNEZ r2
poiché r2 è
usato per aggiornare il loop
e viene manipolato appena sopra

31 January 2012 -- Computer Architectures –- part 2/2

Name, Matricola .....

Question 2

Considering the same loop-based program, and assuming the following processor architecture for a superscalar MIPS64 processor implemented with multiple-issue and speculation:

- issue 2 instructions per clock cycle
- jump instructions require 1 issue
- handle 2 instructions commit per clock cycle
- timing facts for the following separate functional units:
  i. 1 Memory address 1 clock cycle
  ii. 1 Integer ALU 1 clock cycle
  iii. 1 Jump unit 1 clock cycle
  iv. 1 FP multiplier unit, which is pipelined: 8 stages
  v. 1 FP divider unit, which is not pipelined: 10 clock cycles
  vi. 1 FP Arithmetic unit, which is pipelined: 4 stages
- Branch prediction is always correct
- There are no cache misses
- There are 2 CDB (Common Data Bus).

o Complete the table reported below showing the processor behavior for the 2 initial iterations.

o

# iteration		Issue	EXE	MEM	CDB x2	COMMIT x2
1	l.d f1,v1(r1)	1	2 m	3	4	5
1	l.d f2,v2(r1)	1	3 m	4	5	6
1	l.d f3,v3(r1)	2	4 m	5	6	7
1	mul.d f4,f1,f2	2	6 x		14	15
1	s.d f4,v4(r1)	3	5 m			15
1	add.d f5,f1,f2	3	6 a		10	16
1	mul.d f2,f1,f3	4	7 x		15	16
1	add.d f1,f5,f2	4	16 a		20	21
1	s.d f1,v5(r1)	5	6 m			21
1	daddui r1,r1,8	5	6 i		7	22
1	daddi r2,r2,-1	6	7 i		8	22
1	bnez r2,loop	7	9 b			23
2	l.d f1,v1(r1)	8	9 m	10	11	23
2	l.d f2,v2(r1)	8	10 m	11	12	24
2	l.d f3,v3(r1)	9	11 m	12	13	24
2	mul.d f4,f1,f2	9	13 x		21	25
2	s.d f4,v4(r1)	10	12 m			25
2	add.d f5,f1,f2	10	13 a		17	26
2	mul.d f2,f1,f3	11	14 x		22	26
2	add.d f1,f5,f2	11	23 a		27	28
2	s.d f1,v5(r1)	12	13 m			28
2	daddui r1,r1,8	12	13 i		14	29
2	daddi r2,r2,-1	13	14 i		15	29
2	bnez r2,loop	14	16 b			30