Exercise Session 5

Tomasulo, Scoreboard, Scoreboard vs Tomasulo
(Multicycle+integer pipelining)

Advanced Computer Architectures

Politecnico di Milano
April 9th, 2025

Alessandro Verosimile <alessandro.verosimile@polimi.it>

POLITECNICO MILANO 1863

POLITECNICO MILANO 1863
NECST laboratory

Recall: Key Idea: dynamic scheduling

Problem:

data dependences that cannot be hidden with bypassing or forwarding cause hardware stalls of the pipeline

Solution: allow instructions behind a stall to proceed

- HW rearranges the instruction execution to reduce stalls

Enables out-of-order execution and completion (commit)

- Out-of order execution introduces possibility of WAR, WAW data hazards.

First implemented in CDC6600 (1963)

Recall: When is it Safe to Issue an Instruction?

Suppose a data structure keeps track of all the instructions in all the functional units

The following checks need to be made before the Issue stage can dispatch an instruction

- Is the required function unit available?
- Is the input data available? → RAW?
- Is it safe to write the destination? → WAR? WAW?
- Is there a structural conflict at the WB stage?

Recall: Scoreboard structure: three parts

1. Instruction status
2. Functional Unit status

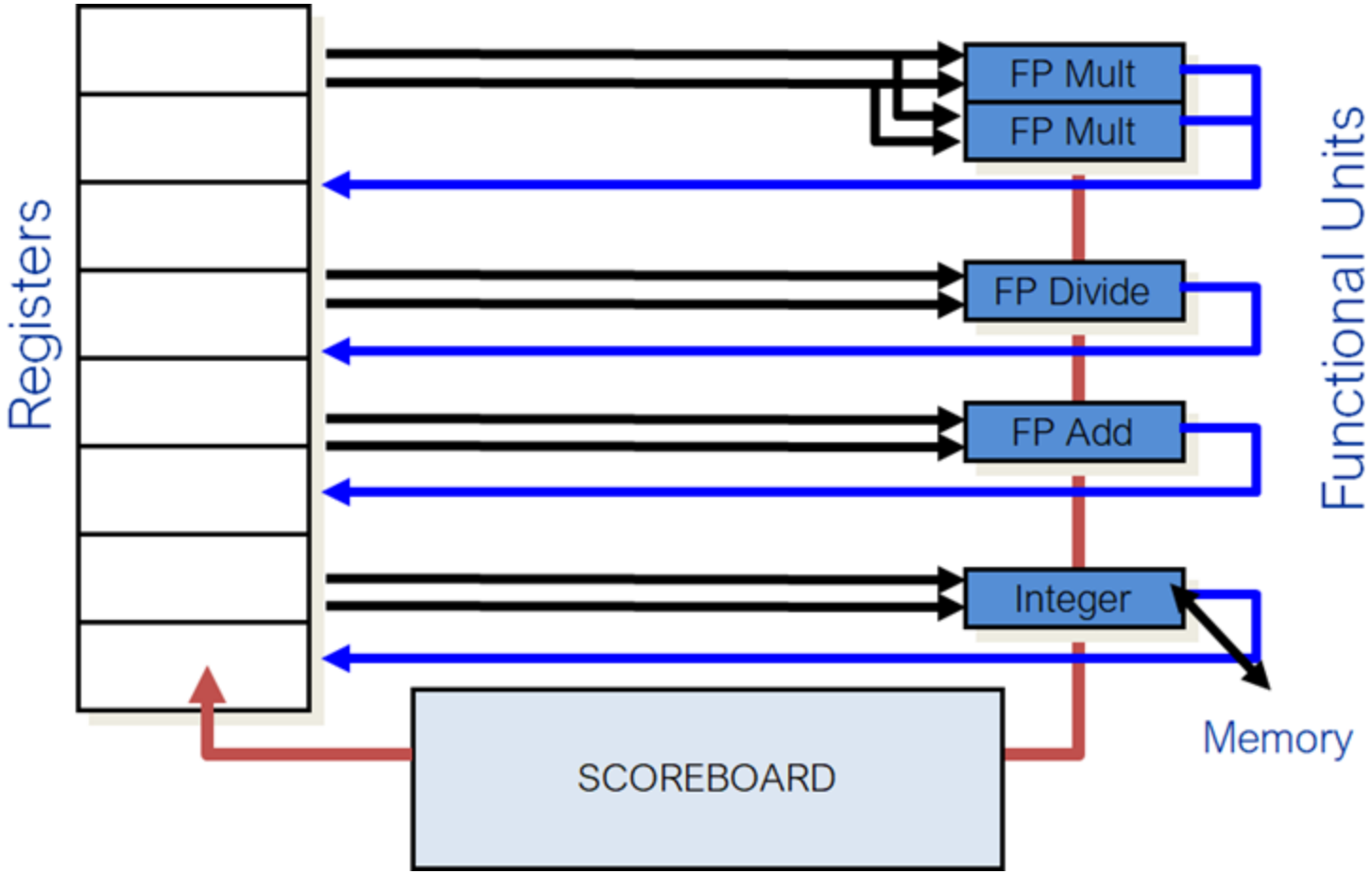
   Indicates the state of the functional unit (FU):

   Busy – Indicates whether the unit is busy or not
   Op - The operation to perform in the unit (+,-, etc.)
   Fi - Destination register
   Fj, Fk – Source register numbers
   Qj, Qk – Functional units producing source registers
   Rj, Rk – Flags indicating when Fj, Fk are ready

3. Register result status

   Indicates which **functional unit will write** each register.
   Blank if **no pending** instructions will write that register.

Recall: Scoreboard



Parallel operation in the control data 6600

Recall: Detailed Scoreboard Pipeline Control
(a.k.a. Instruction Status)

Instruction status	Wait until	Bookkeeping
Issue	Not busy (FU) and not result(D)	Busy(FU)← yes; Op(FU) ← op; Fi(FU) ← `D'; Fj(FU) ← `S1'; Fk(FU) ← `S2'; Qj ← Result('S1'); Qk ← Result(`S2'); Rj ← not Qj; Rk ← not Qk; Result('D') ← FU;
Read operands	Rj and Rk	Rj ← No; Rk ← No
Execution complete	Functional unit done	
Write result	∀f((Fj(f)≠Fi(FU) or Rj(f)=No) & (Fk(f) ≠Fi(FU) or Rk(f)=No))	∀ f(if Qj(f)=FU then Rj(f) ← Yes); ∀ f(if Qk(f)=FU then Rk(f) ← Yes); Result(Fi(FU)) ← 0; Busy(FU) ← No

Recall: the Scoreboard pipeline

ISSUE	READ OPERAND	EXE COMPLETE	WB
Decode instruction;	Read operands;	Operate on operands;	Finish exec;
Structural FUs check; WAW checks	RAW check; WAR if need to read	Notify Scoreboard on completion;	WAR &Struct check (FUs will hold results); Can overlap issue/read&write 4 Structural Hazard;

Recall: Tomasulo Algorithm

Another dynamic algorithm: allows execution to proceed in the presence of dependences

Invented at IBM 3 years after CDC 6600 for the IBM 360/91

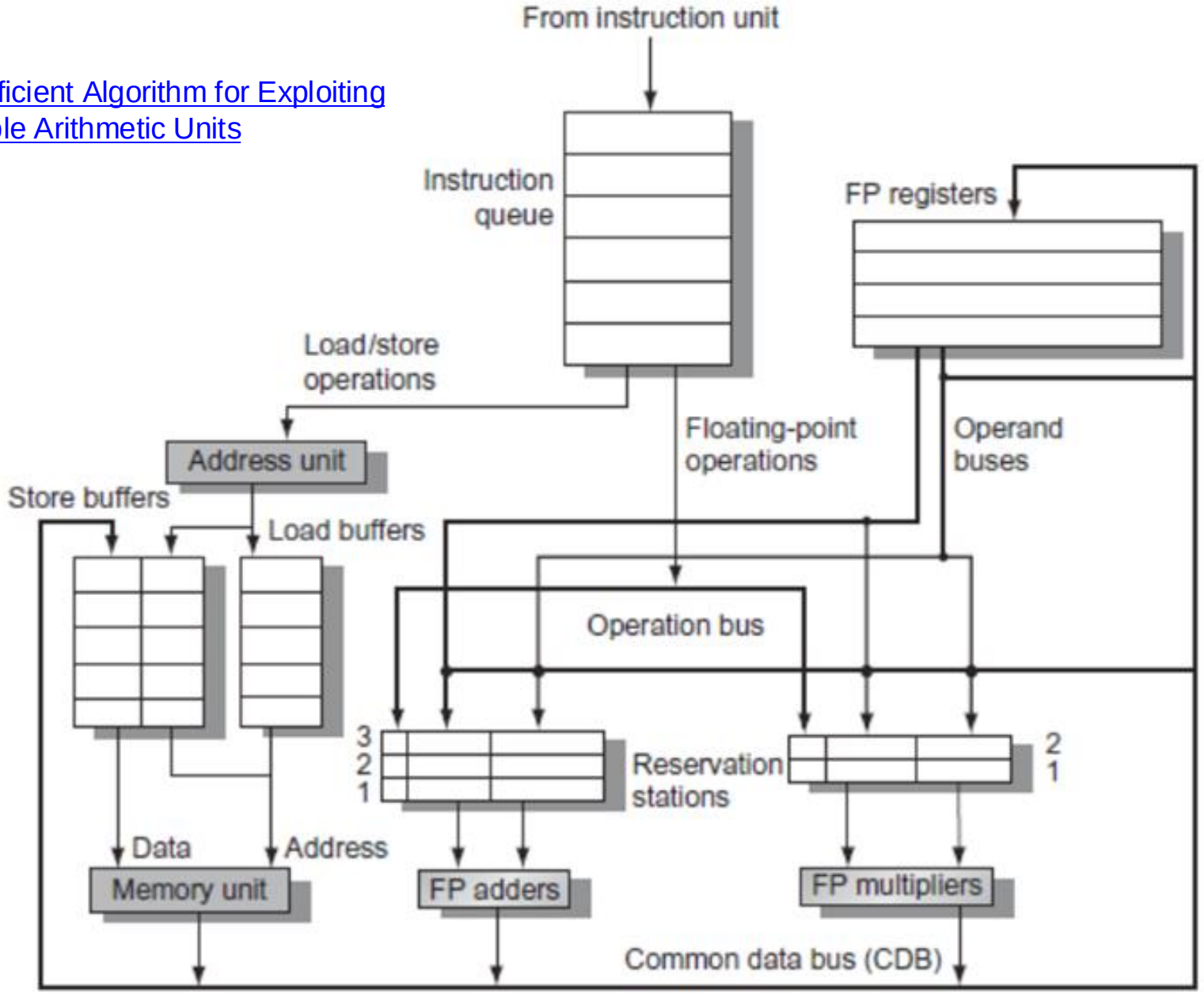
Same Goal: high performance w/o special compilers

Lead to:

Alpha 21264, HP 8000, MIPS 10000, Pentium II, PowerPC 604

Exe 3 Tomasulo

[An Efficient Algorithm for Exploiting Multiple Arithmetic Units]



Recall: the Tomasulo pipeline

ISSUE	EXECUTION	WRITE
Get Instruction from Queue and Rename Registers	Execute and Watch CDB;	Write on CDB;
Structural RSs check; WAW and WAR solved by Renaming (!!!in-order-issue!!!);	Check for Struct on FUs; RAW delaying; Struct check on CDB;	(FUs will hold results unless CDB free) RSs/FUs marked free

Exe 1 Tomasulo: the Code

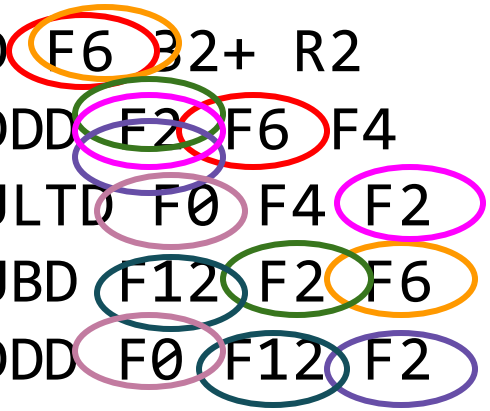
```
I1:  LD F6 32+ R2
I2:  ADDD F2 F6 F4
I3:  MULTD F0 F4 F2
I4:  SUBD F12 F2 F6
I5:  ADDD F0 F12 F2
```

Exe 1 Tomasulo: Conflicts

```
I1:  LD F6 32+ R2
I2:  ADDD F2 F6 F4
I3:  MULTD F0 F4 F2
I4:  SUBD F12 F2 F6
I5:  ADDD F0 F12 F2
```

Exe 1 Tomasulo: Conflicts

```
I1:   LD   F6  32+ R2
I2:   ADDD F2  F6  F4
I3:   MULTD F0  F4  F2
I4:   SUBD F12 F2  F6
I5:   ADDD F0  F12 F2
```



RAW F6 I1-I2

RAW F6 I1-I4

RAW F2 I2-I3

RAW F2 I2-I4

RAW F2 I2-I5

RAW F12 I4-I5

WAW F0 I3-I5

Exe 1 Tomasulo

- 2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE unit (LDU1) with latency 2
- 3 RESERVATION STATIONS (RS3, RS4, RS5) + 3 ALU/BR FUs (ALU1, ALU2, ALU3) with latency 3

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1:LD F6 32+ R2						
I2:ADDD F2 F6 F4						
I3:MULTD F0 F4 F2						
I4:SUBD F12 F2 F6						
I5:ADDD F0 F12 F2						

RAW F6 I1-I2

RAW F6 I1-I4

RAW F2 I2-I3

RAW F2 I2-I4

RAW F2 I2-I5

RAW F12 I4-I5

WAW F0 I3-I5

Exe 1 Tomasulo CC 0

- 2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE unit (LDU1) with latency 2
- 3 RESERVATION STATIONS (RS3, RS4, RS5) + 3 ALU/BR FUs (ALU1, ALU2, ALU3) with latency 3

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1:LD F6 32+ R2						
I2:ADDD F2 F6 F4						
I3:MULTD F0 F4 F2						
I4:SUBD F12 F2 F6						
I5:ADDD F0 F12 F2						

RAW F6 I1-I2

RAW F6 I1-I4

RAW F2 I2-I3

RAW F2 I2-I4

RAW F2 I2-I5

RAW F12 I4-I5

~~**WAW F0 I3-I5**~~

Exe 1 Tomasulo CC 1

- 2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE unit (LDU1) with latency 2
- 3 RESERVATION STATIONS (RS3, RS4, RS5) + 3 ALU/BR FUs (ALU1, ALU2, ALU3) with latency 3

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1:LD F6 32+ R2	1				RS1	
I2:ADDD F2 F6 F4						
I3:MULTD F0 F4 F2						
I4:SUBD F12 F2 F6						
I5:ADDD F0 F12 F2						

RAW F6 I1-I2

RAW F6 I1-I4

RAW F2 I2-I3

RAW F2 I2-I4

RAW F2 I2-I5

RAW F12 I4-I5

~~**WAW F0 I3-I5**~~

Exe 1 Tomasulo CC 2

- 2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE unit (LDU1) with latency 2
- 3 RESERVATION STATIONS (RS3, RS4, RS5) + 3 ALU/BR FUs (ALU1, ALU2, ALU3) with latency 3

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1:LD F6 32+ R2	1	2			RS1	LDU1
I2:ADDD F2 F6 F4	2				RS3	
I3:MULTD F0 F4 F2						
I4:SUBD F12 F2 F6						
I5:ADDD F0 F12 F2						

RAW F6 I1-I2

RAW F6 I1-I4

RAW F2 I2-I3

RAW F2 I2-I4

RAW F2 I2-I5

RAW F12 I4-I5

~~**WAW F0 I3-I5**~~

Exe 1 Tomasulo CC 3

- 2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE unit (LDU1) with latency 2
- 3 RESERVATION STATIONS (RS3, RS4, RS5) + 3 ALU/BR FUs (ALU1, ALU2, ALU3) with latency 3

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1:LD F6 32+ R2	1	2			RS1	LDU1
I2:ADDD F2 F6 F4	2			RAW $F6	RS3	
I3:MULTD F0 F4 F2	3				RS4	
I4:SUBD F12 F2 F6						
I5:ADDD F0 F12 F2						

RAW F6 I1-I2

RAW F6 I1-I4

RAW F2 I2-I3

RAW F2 I2-I4

RAW F2 I2-I5

RAW F12 I4-I5

~~**WAW F0 I3-I5**~~

Exe 1 Tomasulo CC 4

- 2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE unit (LDU1) with latency 2
- 3 RESERVATION STATIONS (RS3, RS4, RS5) + 3 ALU/BR FUs (ALU1, ALU2, ALU3) with latency 3

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1:LD F6 32+ R2	1	2	4		RS1	LDU1
I2:ADDD F2 F6 F4	2			RAW $F6	RS3	
I3:MULTD F0 F4 F2	3			RAW $F2	RS4	
I4:SUBD F12 F2 F6	4				RS5	
I5:ADDD F0 F12 F2						

RAW F6 I1-I2

RAW F6 I1-I4

RAW F2 I2-I3

RAW F2 I2-I4

RAW F2 I2-I5

RAW F12 I4-I5

~~**WAW F0 I3-I5**~~

Exe 1 Tomasulo CC 5

- 2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE unit (LDU1) with latency 2
- 3 RESERVATION STATIONS (RS3, RS4, RS5) + 3 ALU/BR FUs (ALU1, ALU2, ALU3) with latency 3

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1:LD F6 32+ R2	1	2	4		RS1	LDU1
I2:ADDD F2 F6 F4	2	5		RAW $F6	RS3	ALU1
I3:MULTD F0 F4 F2	3			RAW $F2	RS4	
I4:SUBD F12 F2 F6	4			RAW $F2	RS5	
I5:ADDD F0 F12 F2				Struct RS3		

~~RAW F6 I1-I2~~

~~RAW F6 I1-I4~~

RAW F2 I2-I3

RAW F2 I2-I4

RAW F2 I2-I5

RAW F12 I4-I5

~~WAW F0 I3-I5~~

Exe 1 Tomasulo CC 8

- 2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE unit (LDU1) with latency 2
- 3 RESERVATION STATIONS (RS3, RS4, RS5) + 3 ALU/BR FUs (ALU1, ALU2, ALU3) with latency 3

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1:LD F6 32+ R2	1	2	4		RS1	LDU1
I2:ADDD F2 F6 F4	2	5	8	RAW $F6	RS3	ALU1
I3:MULTD F0 F4 F2	3			RAW $F2	RS4	
I4:SUBD F12 F2 F6	4			RAW $F2	RS5	
I5:ADDD F0 F12 F2				Struct RS3		

~~RAW F6 I1-I2~~

~~RAW F6 I1-I4~~

~~RAW F2 I2-I3~~

~~RAW F2 I2-I4~~

~~RAW F2 I2-I5~~

RAW F12 I4-I5

~~WAW F0 I3-I5~~

Exe 1 Tomasulo CC 9

- 2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE unit (LDU1) with latency 2
- 3 RESERVATION STATIONS (RS3, RS4, RS5) + 3 ALU/BR FUs (ALU1, ALU2, ALU3) with latency 3

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1:LD F6 32+ R2	1	2	4		RS1	LDU1
I2:ADDD F2 F6 F4	2	5	8	RAW $F6	RS3	ALU1
I3:MULTD F0 F4 F2	3	9		RAW $F2	RS4	ALU2
I4:SUBD F12 F2 F6	4	9		RAW $F2	RS5	ALU3
I5:ADDD F0 F12 F2	9			Struct RS3	RS3	

~~RAW F6 I1-I2~~

~~RAW F6 I1-I4~~

~~RAW F2 I2-I3~~

~~RAW F2 I2-I4~~

~~RAW F2 I2-I5~~

RAW F12 I4-I5

~~WAW F0 I3-I5~~

Exe 1 Tomasulo CC 10

- 2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE unit (LDU1) with latency 2
- 3 RESERVATION STATIONS (RS3, RS4, RS5) + 3 ALU/BR FUs (ALU1, ALU2, ALU3) with latency 3

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1:LD F6 32+ R2	1	2	4		RS1	LDU1
I2:ADDD F2 F6 F4	2	5	8	RAW $F6	RS3	ALU1
I3:MULTD F0 F4 F2	3	9		RAW $F2	RS4	ALU2
I4:SUBD F12 F2 F6	4	9		RAW $F2	RS5	ALU3
I5:ADDD F0 F12 F2	9			Struct RS3 + RAW $F12	RS3	

~~RAW F6 I1-I2~~

~~RAW F6 I1-I4~~

~~RAW F2 I2-I3~~

~~RAW F2 I2-I4~~

~~RAW F2 I2-I5~~

RAW F12 I4-I5

~~WAW F0 I3-I5~~

Exe 1 Tomasulo CC 12

- 2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE unit (LDU1) with latency 2
- 3 RESERVATION STATIONS (RS3, RS4, RS5) + 3 ALU/BR FUs (ALU1, ALU2, ALU3) with latency 3

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1:LD F6 32+ R2	1	2	4		RS1	LDU1
I2:ADDD F2 F6 F4	2	5	8	RAW $F6	RS3	ALU1
I3:MULTD F0 F4 F2	3	9	12	RAW $F2	RS4	ALU2
I4:SUBD F12 F2 F6	4	9		RAW $F2 + Struct CDB	RS5	ALU3
I5:ADDD F0 F12 F2	9			Struct RS3 + RAW $F12	RS3	

~~RAW F6 I1-I2~~

~~RAW F6 I1-I4~~

~~RAW F2 I2-I3~~

~~RAW F2 I2-I4~~

~~RAW F2 I2-I5~~

RAW F12 I4-I5

~~WAW F0 I3-I5~~

Exe 1 Tomasulo CC 13

- 2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE unit (LDU1) with latency 2
- 3 RESERVATION STATIONS (RS3, RS4, RS5) + 3 ALU/BR FUs (ALU1, ALU2, ALU3) with latency 3

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1:LD F6 32+ R2	1	2	4		RS1	LDU1
I2:ADDD F2 F6 F4	2	5	8	RAW $F6	RS3	ALU1
I3:MULTD F0 F4 F2	3	9	12	RAW $F2	RS4	ALU2
I4:SUBD F12 F2 F6	4	9	13	RAW $F2 + Struct CDB	RS5	ALU3
I5:ADDD F0 F12 F2	9			Struct RS3 + RAW $F12	RS3	

~~RAW F6 I1-I2~~

~~RAW F6 I1-I4~~

~~RAW F2 I2-I3~~

~~RAW F2 I2-I4~~

~~RAW F2 I2-I5~~

~~RAW F12 I4-I5~~

~~WAW F0 I3-I5~~

Exe 1 Tomasulo CC 14

- 2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE unit (LDU1) with latency 2
- 3 RESERVATION STATIONS (RS3, RS4, RS5) + 3 ALU/BR FUs (ALU1, ALU2, ALU3) with latency 3

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1:LD F6 32+ R2	1	2	4		RS1	LDU1
I2:ADDD F2 F6 F4	2	5	8	RAW $F6	RS3	ALU1
I3:MULTD F0 F4 F2	3	9	12	RAW $F2	RS4	ALU2
I4:SUBD F12 F2 F6	4	9	13	RAW $F2 + Struct CDB	RS5	ALU3
I5:ADDD F0 F12 F2	9	14		Struct RS3 + RAW $F12	RS3	ALU1

~~RAW F6 I1-I2~~

~~RAW F6 I1-I4~~

~~RAW F2 I2-I3~~

~~RAW F2 I2-I4~~

~~RAW F2 I2-I5~~

~~RAW F12 I4-I5~~

~~WAW F0 I3-I5~~

Exe 1 Tomasulo CC 17

- 2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE unit (LDU1) with latency 2
- 3 RESERVATION STATIONS (RS3, RS4, RS5) + 3 ALU/BR FUs (ALU1, ALU2, ALU3) with latency 3

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1:LD F6 32+ R2	1	2	4		RS1	LDU1
I2:ADDD F2 F6 F4	2	5	8	RAW $F6	RS3	ALU1
I3:MULTD F0 F4 F2	3	9	12	RAW $F2	RS4	ALU2
I4:SUBD F12 F2 F6	4	9	13	RAW $F2 + Struct CDB	RS5	ALU3
I5:ADDD F0 F12 F2	9	14	17	Struct RS3 + RAW $F12	RS3	ALU1

~~RAW F6 I1-I2~~

~~RAW F6 I1-I4~~

~~RAW F2 I2-I3~~

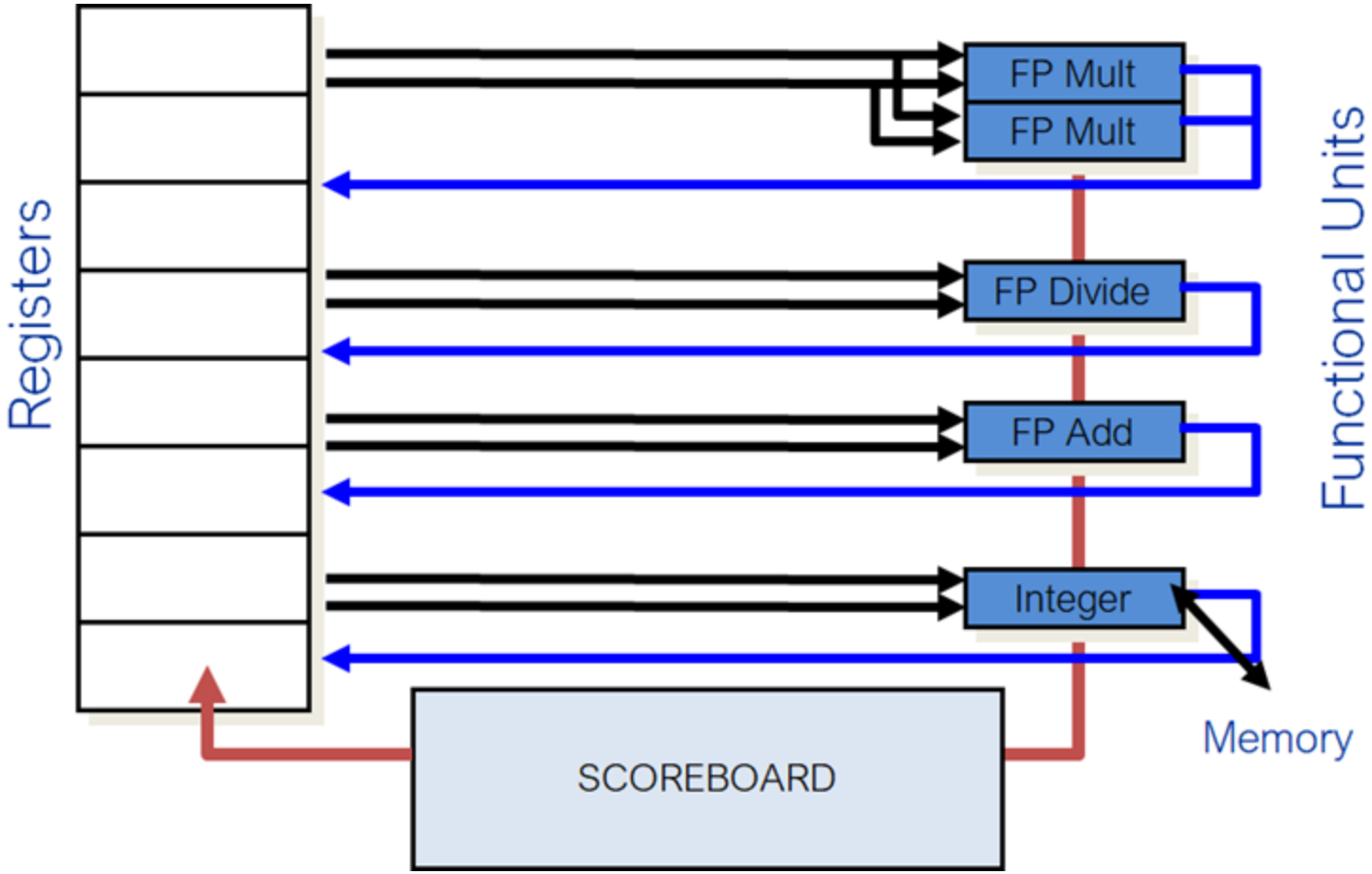
~~RAW F2 I2-I4~~

~~RAW F2 I2-I5~~

~~RAW F12 I4-I5~~

~~WAW F0 I3-I5~~

Exe Scoreboard



Parallel operation in the control data 6600

Recall: the Scoreboard pipeline

ISSUE	READ OPERAND	EXE COMPLETE	WB
Decode instruction;	Read operands;	Operate on operands;	Finish exec;
Structural FUs check; WAW checks	RAW check; WAR if need to read	Notify Scoreboard on completion;	WAR & Struct check (FUs will hold results); Can overlap issue/read&write;

Exe.2 The code

I1: LD $F1, 0($R1)
I2: FADD $F2, $F2, $F3
I3: ADDI $R3, $R3, 8
I4: LD $F4, 0(R2)
I5: FADD $F5, $F4, $F2
I6: FMULT $F6, $F1, $F4
I7: ADDI $R5, $R5, 1
I8: LD $R6, 0($R4)
I9: SD $F6, 0($R5)
I10: SD $F5, 0($R1)
I11: ADD $R1, $R6, $R1

Exe.2 The conflicts

I1: LD $F1, 0($R1)
I2: FADD $F2, $F2, $F3
I3: ADDI $R3, $R3, 8
I4: LD $F4, 0(R2)
I5: FADD $F5, $F4, $F2
I6: FMULT $F6, $F1, $F4
I7: ADDI $R5, $R5, 1
I8: LD $R6, 0($R4)
I9: SD $F6, 0($R5)
I10: SD $F5, 0($R1)
I11: ADD $R1, $R6, $R1

Exe.2 The conflicts

I1: LD $F1, 0($R1)
I2: FADD $F2, $F2, $F3
I3: ADDI $R3, $R3, 8
I4: LD $F4, 0(R2)
I5: FADD $F5, $F4, $F2
I6: FMULT $F6, $F1, $F4
I7: ADDI $R5, $R5, 1
I8: LD $R6, 0($R4)
I9: SD $F6, 0($R5)
I10: SD $F5, 0($R1)
I11: ADD $R1, $R6, $R1

RAW F0 I1-I6

Exe.2 The conflicts

```
I1:  LD $F1, 0($R1)
I2:  FADD $F2, $F2, $F3
I3:  ADDI $R3, $R3, 8
I4:  LD $F4, 0(R2)
I5:  FADD $F5, $F4, $F2
I6:  FMULT $F6, $F1, $F4
I7:  ADDI $R5, $R5, 1
I8:  LD $R6, 0($R4)
I9:  SD $F6, 0($R5)
I10: SD $F5, 0($R1)
I11: ADD $R1, $R6, $R1
```

RAW F0 I1-I6

RAW F2 I2-I5

Exe.2 The conflicts

```
I1:  LD $F1, 0($R1)
I2:  FADD $F2, $F2, $F3
I3:  ADDI $R3, $R3, 8
I4:  LD $F4, 0(R2)
I5:  FADD $F5, $F4, $F2
I6:  FMULT $F6, $F1, $F4
I7:  ADDI $R5, $R5, 1
I8:  LD $R6, 0($R4)
I9:  SD $F6, 0($R5)
I10: SD $F5, 0($R1)
I11: ADD $R1, $R6, $R1
```

RAW F0 I1-I6

RAW F2 I2-I5

RAW F4 I4-I5

RAW F4 I4-I6

Exe.2 The conflicts

```
I1:  LD $F1, 0($R1)
I2:  FADD $F2, $F2, $F3
I3:  ADDI $R3, $R3, 8
I4:  LD $F4, 0(R2)
I5:  FADD $F5, $F4, $F2
I6:  FMULT $F6, $F1, $F4
I7:  ADDI $R5, $R5, 1
I8:  LD $R6, 0($R4)
I9:  SD $F6, 0($R5)
I10: SD $F5, 0($R1)
I11: ADD $R1, $R6, $R1
```

RAW F0 I1-I6

RAW F2 I2-I5

RAW F4 I4-I5

RAW F4 I4-I6

RAW F5 I5-I10

Exe.2 The conflicts

```
I1:  LD $F1, 0($R1)
I2:  FADD $F2, $F2, $F3
I3:  ADDI $R3, $R3, 8
I4:  LD $F4, 0(R2)
I5:  FADD $F5, $F4, $F2
I6:  FMULT $F6, $F1, $F4
I7:  ADDI $R5, $R5, 1
I8:  LD $R6, 0($R4)
I9:  SD $F6, 0($R5)
I10: SD $F5, 0($R1)
I11: ADD $R1, $R6, $R1
```

RAW F0 I1-I6

RAW F2 I2-I5

RAW F4 I4-I5

RAW F4 I4-I6

RAW F5 I5-I10

RAW F6 I6-I9

Exe.2 The conflicts

```
I1:  LD $F1, 0($R1)
I2:  FADD $F2, $F2, $F3
I3:  ADDI $R3, $R3, 8
I4:  LD $F4, 0(R2)
I5:  FADD $F5, $F4, $F2
I6:  FMULT $F6, $F1, $F4
I7:  ADDI $R5, $R5, 1
I8:  LD $R6, 0($R4)
I9:  SD $F6, 0($R5)
I10: SD $F5, 0($R1)
I11: ADD $R1, $R6, $R1
```

RAW F0 I1-I6

RAW F2 I2-I5

RAW F4 I4-I5

RAW F4 I4-I6

RAW F5 I5-I10

RAW F6 I6-I9

RAW R5 I7-I9

Exe.2 The conflicts

```
I1:  LD $F1, 0($R1)
I2:  FADD $F2, $F2, $F3
I3:  ADDI $R3, $R3, 8
I4:  LD $F4, 0(R2)
I5:  FADD $F5, $F4, $F2
I6:  FMULT $F6, $F1, $F4
I7:  ADDI $R5, $R5, 1
I8:  LD $R6, 0($R4)
I9:  SD $F6, 0($R5)
I10: SD $F5, 0($R1)
I11: ADD $R1, $R6, $R1
```

RAW F0 I1-I6

RAW F2 I2-I5

RAW F4 I4-I5

RAW F4 I4-I6

RAW F5 I5-I10

RAW F6 I6-I9

RAW R5 I7-I9

RAW R6 I8-I11

Exe.2 The conflicts

```
I1:   LD $F1, 0($R1)
I2:   FADD $F2, $F2, $F3
I3:   ADDI $R3, $R3, 8
I4:   LD $F4, 0(R2)
I5:   FADD $F5, $F4, $F2
I6:   FMULT $F6, $F1, $F4
I7:   ADDI $R5, $R5, 1
I8:   LD $R6, 0($R4)
I9:   SD $F6, 0($R5)
I10: SD $F5, 0($R1)
I11: ADD $R1, $R6, $R1
```

RAW F0 I1-I6

RAW F2 I2-I5

RAW F4 I4-I5

RAW F4 I4-I6

RAW F5 I5-I10

RAW F6 I6-I9

RAW R5 I7-I9

RAW R6 I8-I11

WAR R1 I10-I11

Exe.2 Scoreboard CC0

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)						
I2	FADD $F2, $F2, $F3						
I3	ADDI $R3, $R3, 8						
I4	LD $F4, 0(R2)						
I5	FADD $F5, $F4, $F2						
I6	FMULT $F6, $F1, $F4						
I7	ADDI $R5, $R5, 1						
I8	LD $R6, 0($R4)						
I9	SD $F6, 0($R5)						
I10	SD $F5, 0($R1)						
I11	ADD $R1, $R6, $R1						

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
RAW R5 I7-I9

Exe.2 Scoreboard CC1

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1					MU1
I2	FADD $F2, $F2, $F3						
I3	ADDI $R3, $R3, 8						
I4	LD $F4, 0(R2)						
I5	FADD $F5, $F4, $F2						
I6	FMULT $F6, $F1, $F4						
I7	ADDI $R5, $R5, 1						
I8	LD $R6, 0($R4)						
I9	SD $F6, 0($R5)						
I10	SD $F5, 0($R1)						
I11	ADD $R1, $R6, $R1						

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC2

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2				MU1
I2	FADD $F2, $F2, $F3	2					FPU1
I3	ADDI $R3, $R3, 8						
I4	LD $F4, 0(R2)						
I5	FADD $F5, $F4, $F2						
I6	FMULT $F6, $F1, $F4						
I7	ADDI $R5, $R5, 1						
I8	LD $R6, 0($R4)						
I9	SD $F6, 0($R5)						
I10	SD $F5, 0($R1)						
I11	ADD $R1, $R6, $R1						

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11

RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11

RAW R5 I7-I9

Exe.2 Scoreboard CC3

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2				MU1
I2	FADD $F2, $F2, $F3	2	3				FPU1
I3	ADDI $R3, $R3, 8	3					ALU1
I4	LD $F4, 0(R2)						
I5	FADD $F5, $F4, $F2						
I6	FMULT $F6, $F1, $F4						
I7	ADDI $R5, $R5, 1						
I8	LD $R6, 0($R4)						
I9	SD $F6, 0($R5)						
I10	SD $F5, 0($R1)						
I11	ADD $R1, $R6, $R1						

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
RAW R5 I7-I9

Exe.2 Scoreboard CC4

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2				MU1
I2	FADD $F2, $F2, $F3	2	3				FPU1
I3	ADDI $R3, $R3, 8	3	4				ALU1
I4	LD $F4, 0(R2)	4					MU2
I5	FADD $F5, $F4, $F2						
I6	FMULT $F6, $F1, $F4						
I7	ADDI $R5, $R5, 1						
I8	LD $R6, 0($R4)						
I9	SD $F6, 0($R5)						
I10	SD $F5, 0($R1)						
I11	ADD $R1, $R6, $R1						

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC5

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5			MU1
I2	FADD $F2, $F2, $F3	2	3				FPU1
I3	ADDI $R3, $R3, 8	3	4	5			ALU1
I4	LD $F4, 0(R2)	4	5				MU2
I5	FADD $F5, $F4, $F2	5					FPU2
I6	FMULT $F6, $F1, $F4						
I7	ADDI $R5, $R5, 1						
I8	LD $R6, 0($R4)						
I9	SD $F6, 0($R5)						
I10	SD $F5, 0($R1)						
I11	ADD $R1, $R6, $R1						

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC6

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5	6		MU1
I2	FADD $F2, $F2, $F3	2	3				FPU1
I3	ADDI $R3, $R3, 8	3	4	5		Struct RF	ALU1
I4	LD $F4, 0(R2)	4	5				MU2
I5	FADD $F5, $F4, $F2	5				RAW F2 + RAW F4	FPU2
I6	FMULT $F6, $F1, $F4	6					FPU3
I7	ADDI $R5, $R5, 1						
I8	LD $R6, 0($R4)						
I9	SD $F6, 0($R5)						
I10	SD $F5, 0($R1)						
I11	ADD $R1, $R6, $R1						

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC7

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5	6		MU1
I2	FADD $F2, $F2, $F3	2	3	7			FPU1
I3	ADDI $R3, $R3, 8	3	4	5	7	Struct RF	ALU1
I4	LD $F4, 0(R2)	4	5				MU2
I5	FADD $F5, $F4, $F2	5				RAW F2 + RAW F4	FPU2
I6	FMULT $F6, $F1, $F4	6				RAW F4	FPU3
I7	ADDI $R5, $R5, 1	7					ALU2
I8	LD $R6, 0($R4)						
I9	SD $F6, 0($R5)						
I10	SD $F5, 0($R1)						
I11	ADD $R1, $R6, $R1						

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC8

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5	6		MU1
I2	FADD $F2, $F2, $F3	2	3	7	8		FPU1
I3	ADDI $R3, $R3, 8	3	4	5	7	Struct RF	ALU1
I4	LD $F4, 0(R2)	4	5	8			MU2
I5	FADD $F5, $F4, $F2	5				RAW F2 + RAW F4	FPU2
I6	FMULT $F6, $F1, $F4	6				RAW F4	FPU3
I7	ADDI $R5, $R5, 1	7	8				ALU2
I8	LD $R6, 0($R4)	8					MU3
I9	SD $F6, 0($R5)						
I10	SD $F5, 0($R1)						
I11	ADD $R1, $R6, $R1						

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
___ RAW R5 I7-I9 _____

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC9

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5	6		MU1
I2	FADD $F2, $F2, $F3	2	3	7	8		FPU1
I3	ADDI $R3, $R3, 8	3	4	5	7	Struct RF	ALU1
I4	LD $F4, 0(R2)	4	5	8	9		MU2
I5	FADD $F5, $F4, $F2	5				RAW F2 + RAW F4	FPU2
I6	FMULT $F6, $F1, $F4	6				RAW F4	FPU3
I7	ADDI $R5, $R5, 1	7	8	9			ALU2
I8	LD $R6, 0($R4)	8	9				MU3
I9	SD $F6, 0($R5)	9					MU1
I10	SD $F5, 0($R1)						
I11	ADD $R1, $R6, $R1						

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC10

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5	6		MU1
I2	FADD $F2, $F2, $F3	2	3	7	8		FPU1
I3	ADDI $R3, $R3, 8	3	4	5	7	Struct RF	ALU1
I4	LD $F4, 0(R2)	4	5	8	9		MU2
I5	FADD $F5, $F4, $F2	5	10			RAW F2 + RAW F4	FPU2
I6	FMULT $F6, $F1, $F4	6	10			RAW F4	FPU3
I7	ADDI $R5, $R5, 1	7	8	9	10		ALU2
I8	LD $R6, 0($R4)	8	9				MU3
I9	SD $F6, 0($R5)	9				RAW R5 + RAW F6	MU1
I10	SD $F5, 0($R1)	10					MU2
I11	ADD $R1, $R6, $R1						

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC11

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5	6		MU1
I2	FADD $F2, $F2, $F3	2	3	7	8		FPU1
I3	ADDI $R3, $R3, 8	3	4	5	7	Struct RF	ALU1
I4	LD $F4, 0(R2)	4	5	8	9		MU2
I5	FADD $F5, $F4, $F2	5	10			RAW F2 + RAW F4	FPU2
I6	FMULT $F6, $F1, $F4	6	10			RAW F4	FPU3
I7	ADDI $R5, $R5, 1	7	8	9	10		ALU2
I8	LD $R6, 0($R4)	8	9				MU3
I9	SD $F6, 0($R5)	9				RAW R5 + RAW F6	MU1
I10	SD $F5, 0($R1)	10				RAW F5	MU2
I11	ADD $R1, $R6, $R1	11					ALU1

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC12

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5	6		MU1
I2	FADD $F2, $F2, $F3	2	3	7	8		FPU1
I3	ADDI $R3, $R3, 8	3	4	5	7	Struct RF	ALU1
I4	LD $F4, 0(R2)	4	5	8	9		MU2
I5	FADD $F5, $F4, $F2	5	10			RAW F2 + RAW F4	FPU2
I6	FMULT $F6, $F1, $F4	6	10			RAW F4	FPU3
I7	ADDI $R5, $R5, 1	7	8	9	10		ALU2
I8	LD $R6, 0($R4)	8	9	12			MU3
I9	SD $F6, 0($R5)	9				RAW R5 + RAW F6	MU1
I10	SD $F5, 0($R1)	10				RAW F5	MU2
I11	ADD $R1, $R6, $R1	11				RAW F5	ALU1

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC13

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5	6		MU1
I2	FADD $F2, $F2, $F3	2	3	7	8		FPU1
I3	ADDI $R3, $R3, 8	3	4	5	7	Struct RF	ALU1
I4	LD $F4, 0(R2)	4	5	8	9		MU2
I5	FADD $F5, $F4, $F2	5	10			RAW F2 + RAW F4	FPU2
I6	FMULT $F6, $F1, $F4	6	10			RAW F4	FPU3
I7	ADDI $R5, $R5, 1	7	8	9	10		ALU2
I8	LD $R6, 0($R4)	8	9	12	13		MU3
I9	SD $F6, 0($R5)	9				RAW R5 + RAW F6	MU1
I10	SD $F5, 0($R1)	10				RAW F5	MU2
I11	ADD $R1, $R6, $R1	11				RAW F5	ALU1

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC14

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5	6		MU1
I2	FADD $F2, $F2, $F3	2	3	7	8		FPU1
I3	ADDI $R3, $R3, 8	3	4	5	7	Struct RF	ALU1
I4	LD $F4, 0(R2)	4	5	8	9		MU2
I5	FADD $F5, $F4, $F2	5	10	14		RAW F2 + RAW F4	FPU2
I6	FMULT $F6, $F1, $F4	6	10	14		RAW F4	FPU3
I7	ADDI $R5, $R5, 1	7	8	9	10		ALU2
I8	LD $R6, 0($R4)	8	9	12	13		MU3
I9	SD $F6, 0($R5)	9				RAW R5 + RAW F6	MU1
I10	SD $F5, 0($R1)	10				RAW F5	MU2
I11	ADD $R1, $R6, $R1	11	14			RAW F5	ALU1

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC15

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5	6		MU1
I2	FADD $F2, $F2, $F3	2	3	7	8		FPU1
I3	ADDI $R3, $R3, 8	3	4	5	7	Struct RF	ALU1
I4	LD $F4, 0(R2)	4	5	8	9		MU2
I5	FADD $F5, $F4, $F2	5	10	14	15	RAW F2 + RAW F4	FPU2
I6	FMULT $F6, $F1, $F4	6	10	14		RAW F4 + Struct RF	FPU3
I7	ADDI $R5, $R5, 1	7	8	9	10		ALU2
I8	LD $R6, 0($R4)	8	9	12	13		MU3
I9	SD $F6, 0($R5)	9				RAW R5 + RAW F6	MU1
I10	SD $F5, 0($R1)	10				RAW F5	MU2
I11	ADD $R1, $R6, $R1	11	14	15		RAW F5	ALU1

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC16

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5	6		MU1
I2	FADD $F2, $F2, $F3	2	3	7	8		FPU1
I3	ADDI $R3, $R3, 8	3	4	5	7	Struct RF	ALU1
I4	LD $F4, 0(R2)	4	5	8	9		MU2
I5	FADD $F5, $F4, $F2	5	10	14	15	RAW F2 + RAW F4	FPU2
I6	FMULT $F6, $F1, $F4	6	10	14	16	RAW F4 + Struct RF	FPU3
I7	ADDI $R5, $R5, 1	7	8	9	10		ALU2
I8	LD $R6, 0($R4)	8	9	12	13		MU3
I9	SD $F6, 0($R5)	9				RAW R5 + RAW F6	MU1
I10	SD $F5, 0($R1)	10	16			RAW F5	MU2
I11	ADD $R1, $R6, $R1	11	14	15		RAW F5 + WAR R1	ALU1

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11
RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11
RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC17

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5	6		MU1
I2	FADD $F2, $F2, $F3	2	3	7	8		FPU1
I3	ADDI $R3, $R3, 8	3	4	5	7	Struct RF	ALU1
I4	LD $F4, 0(R2)	4	5	8	9		MU2
I5	FADD $F5, $F4, $F2	5	10	14	15	RAW F2 + RAW F4	FPU2
I6	FMULT $F6, $F1, $F4	6	10	14	16	RAW F4 + Struct RF	FPU3
I7	ADDI $R5, $R5, 1	7	8	9	10		ALU2
I8	LD $R6, 0($R4)	8	9	12	13		MU3
I9	SD $F6, 0($R5)	9	17			RAW R5 + RAW F6	MU1
I10	SD $F5, 0($R1)	10	16			RAW F5	MU2
I11	ADD $R1, $R6, $R1	11	14	15	17	RAW F5 + WAR R1	ALU1

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11

RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11

RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC19

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5	6		MU1
I2	FADD $F2, $F2, $F3	2	3	7	8		FPU1
I3	ADDI $R3, $R3, 8	3	4	5	7	Struct RF	ALU1
I4	LD $F4, 0(R2)	4	5	8	9		MU2
I5	FADD $F5, $F4, $F2	5	10	14	15	RAW F2 + RAW F4	FPU2
I6	FMULT $F6, $F1, $F4	6	10	14	16	RAW F4 + Struct RF	FPU3
I7	ADDI $R5, $R5, 1	7	8	9	10		ALU2
I8	LD $R6, 0($R4)	8	9	12	13		MU3
I9	SD $F6, 0($R5)	9	17			RAW R5 + RAW F6	MU1
I10	SD $F5, 0($R1)	10	16	19		RAW F5	MU2
I11	ADD $R1, $R6, $R1	11	14	15	17	RAW F5 + WAR R1	ALU1

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11

RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11

RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC20

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5	6		MU1
I2	FADD $F2, $F2, $F3	2	3	7	8		FPU1
I3	ADDI $R3, $R3, 8	3	4	5	7	Struct RF	ALU1
I4	LD $F4, 0(R2)	4	5	8	9		MU2
I5	FADD $F5, $F4, $F2	5	10	14	15	RAW F2 + RAW F4	FPU2
I6	FMULT $F6, $F1, $F4	6	10	14	16	RAW F4 + Struct RF	FPU3
I7	ADDI $R5, $R5, 1	7	8	9	10		ALU2
I8	LD $R6, 0($R4)	8	9	12	13		MU3
I9	SD $F6, 0($R5)	9	17	20		RAW R5 + RAW F6	MU1
I10	SD $F5, 0($R1)	10	16	19	20	RAW F5	MU2
I11	ADD $R1, $R6, $R1	11	14	15	17	RAW F5 + WAR R1	ALU1

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11

RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11

RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.2 Scoreboard CC20

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB	Hazards	Unit
I1	LD $F1, 0($R1)	1	2	5	6		MU1
I2	FADD $F2, $F2, $F3	2	3	7	8		FPU1
I3	ADDI $R3, $R3, 8	3	4	5	7	Struct RF	ALU1
I4	LD $F4, 0(R2)	4	5	8	9		MU2
I5	FADD $F5, $F4, $F2	5	10	14	15	RAW F2 + RAW F4	FPU2
I6	FMULT $F6, $F1, $F4	6	10	14	16	RAW F4 + Struct RF	FPU3
I7	ADDI $R5, $R5, 1	7	8	9	10		ALU2
I8	LD $R6, 0($R4)	8	9	12	13		MU3
I9	SD $F6, 0($R5)	9	17	20	21	RAW R5 + RAW F6	MU1
I10	SD $F5, 0($R1)	10	16	19	20	RAW F5	MU2
I11	ADD $R1, $R6, $R1	11	14	15	17	RAW F5 + WAR R1	ALU1

3 MU ,3cc
3 FPUs, 4cc
2 Integer ALU, 1cc
Single W port overall

RAW F0 I1-I6 RAW F4 I4-I5 RAW F5 I5-I10 RAW R6 I8-I11

RAW F2 I2-I5 RAW F4 I4-I6 RAW F6 I6-I9 WAR R1 I10-I11

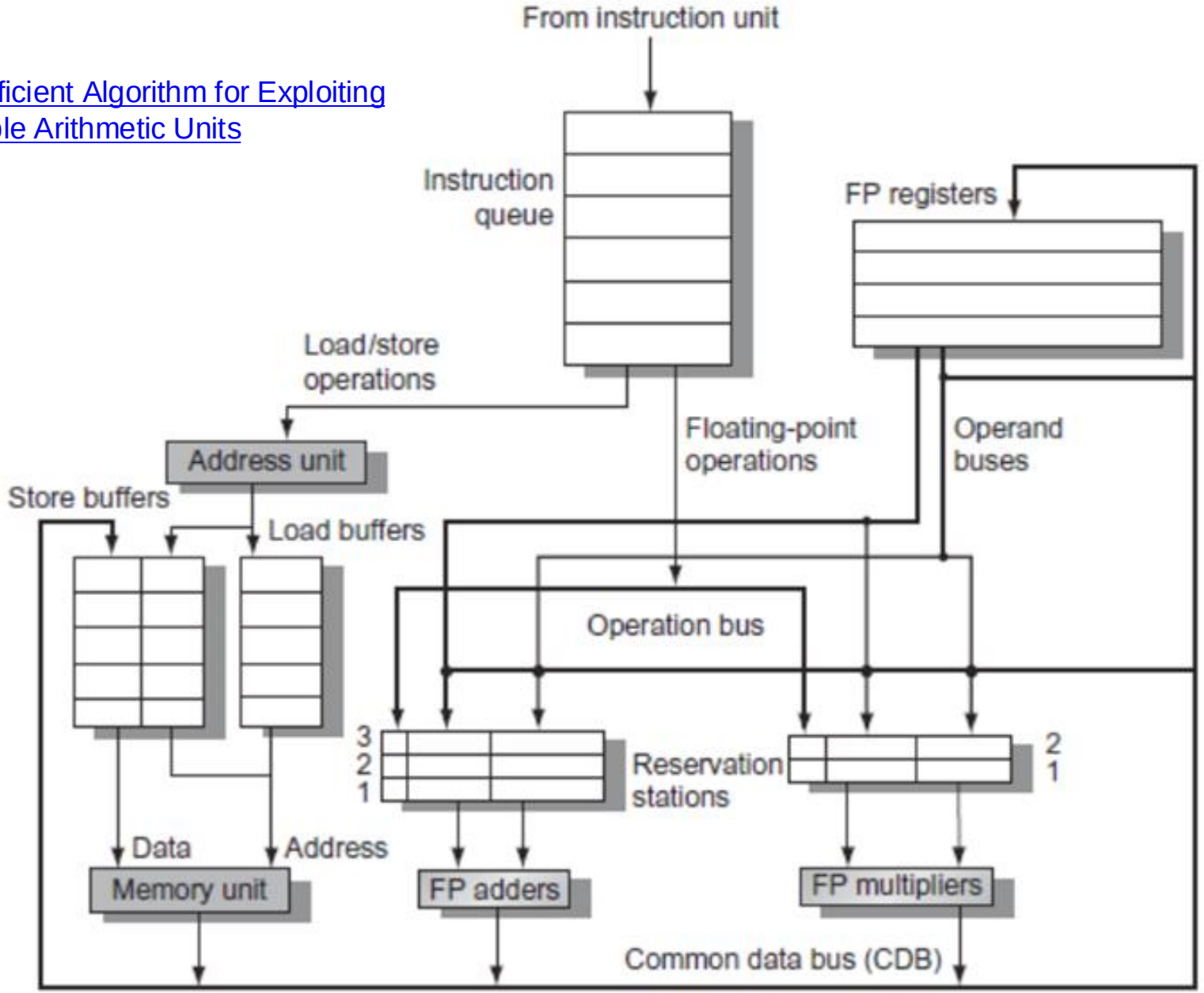
RAW R5 I7-I9

POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

Exe.3 Tomasulo

[An Efficient Algorithm for Exploiting Multiple Arithmetic Units]



Exe.3 Tomasulo

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Exe.3 The conflicts

I1: LD $F1, 0($R1)
I2: FADD $F2, $F2, $F3
I3: ADDI $R3, $R3, 8
I4: LD $F4, 0(R2)
I5: FADD $F5, $F4, $F2
I6: FMULT $F6, $F1, $F4
I7: ADDI $R5, $R5, 1
I8: LD $R6, 0($R4)
I9: SD $F6, 0($R5)
I10: SD $F5, 0($R1)
I11: ADD $R1, $R6, $R1

Exe.3 The conflicts

```
I1:   LD $F1, 0($R1)
I2:   FADD $F2, $F2, $F3
I3:   ADDI $R3, $R3, 8
I4:   LD $F4, 0(R2)
I5:   FADD $F5, $F4, $F2
I6:   FMULT $F6, $F1, $F4
I7:   ADDI $R5, $R5, 1
I8:   LD $R6, 0($R4)
I9:   SD $F6, 0($R5)
I10: SD $F5, 0($R1)
I11: ADD $R1, $R6, $R1
```

RAW F0 I1-I6

RAW F2 I2-I5

RAW F4 I4-I5

RAW F4 I4-I6

RAW F5 I5-I10

RAW F6 I6-I9

RAW R5 I7-I9

RAW R6 I8-I11

WAR R1 I10-I11

Exe.3 The conflicts

```
I1:  LD $F1, 0($R1)
I2:  FADD $F2, $F2, $F3
I3:  ADDI $R3, $R3, 8
I4:  LD $F4, 0(R2)
I5:  FADD $F5, $F4, $F2
I6:  FMULT $F6, $F1, $F4
I7:  ADDI $R5, $R5, 1
I8:  LD $R6, 0($R4)
I9:  SD $F6, 0($R5)
I10: SD $F5, 0($R1)
I11: ADD $R1, $R6, $R1
```

RAW F0 I1-I6

RAW F2 I2-I5

RAW F4 I4-I5

RAW F4 I4-I6

RAW F5 I5-I10

RAW F6 I6-I9

RAW R5 I7-I9

RAW R6 I8-I11

~~WAR R1 I10-I11~~

Recall: the Tomasulo pipeline

ISSUE	EXECUTION	WRITE
Get Instruction from Queue and Rename Registers	**Execute and Watch CDB;**	**Write on CDB;**
Structural RSs check; WAW and WAR solved by Renaming (!!!in-order-issue!!!);	**Check for Struct on FUs; RAW delaying; Struct check on CDB;**	**(FUs will hold results unless CDB free) RSs/FUs marked free**

Exe.3 Tomasulo CC0

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)						
I2: FADD $F2, $F2, $F3						
I3: ADDI $R3, $R3, 8						
I4: LD $F4, 0(R2)						
I5: FADD $F5, $F4, $F2						
I6: FMULT $F6, $F1, $F4						
I7: ADDI $R5, $R5, 1						
I8: LD $R6, 0($R4)						
I9: SD $F6, 0($R5)						
I10: SD $F5, 0($R1)						
I11: ADD $R1, $R6, $R1						

RAW F0 I1-I6 RAW F2 I2-I5 RAW F4 I4-I6 RAW F4 I4-I5

RAW F5 I5-I10 RAW F6 I6-I9 RAW R5 I7-I9 RAW R6 I8-I11

Exe.3 Tomasulo CC1

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	**1**				RS1	
I2: FADD $F2, $F2, $F3						
I3: ADDI $R3, $R3, 8						
I4: LD $F4, 0(R2)						
I5: FADD $F5, $F4, $F2						
I6: FMULT $F6, $F1, $F4						
I7: ADDI $R5, $R5, 1						
I8: LD $R6, 0($R4)						
I9: SD $F6, 0($R5)						
I10: SD $F5, 0($R1)						
I11: ADD $R1, $R6, $R1						

RAW F0 I1-I6 **RAW F2 I2-I5** **RAW F4 I4-I6** **RAW F4 I4-I5**

RAW F5 I5-I10 **RAW F6 I6-I9** **RAW R5 I7-I9** **RAW R6 I8-I11**

Exe.3 Tomasulo CC2

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	**1**	**2**			RS1	LDU1
I2: FADD $F2, $F2, $F3	**2**				RS4	
I3: ADDI $R3, $R3, 8						
I4: LD $F4, 0(R2)						
I5: FADD $F5, $F4, $F2						
I6: FMULT $F6, $F1, $F4						
I7: ADDI $R5, $R5, 1						
I8: LD $R6, 0($R4)						
I9: SD $F6, 0($R5)						
I10: SD $F5, 0($R1)						
I11: ADD $R1, $R6, $R1						

RAW F0 I1-I6 **RAW F2 I2-I5** **RAW F4 I4-I6** **RAW F4 I4-I5**

RAW F5 I5-I10 **RAW F6 I6-I9** **RAW R5 I7-I9** **RAW R6 I8-I11**

Exe.3 Tomasulo CC3

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	**1**	**2**			RS1	LDU1
I2: FADD $F2, $F2, $F3	**2**	**3**			RS4	FPU1
I3: ADDI $R3, $R3, 8	**3**				RS7	
I4: LD $F4, 0(R2)						
I5: FADD $F5, $F4, $F2						
I6: FMULT $F6, $F1, $F4						
I7: ADDI $R5, $R5, 1						
I8: LD $R6, 0($R4)						
I9: SD $F6, 0($R5)						
I10: SD $F5, 0($R1)						
I11: ADD $R1, $R6, $R1						

RAW F0 I1-I6 **RAW F2 I2-I5** **RAW F4 I4-I6** **RAW F4 I4-I5**

RAW F5 I5-I10 **RAW F6 I6-I9** **RAW R5 I7-I9** **RAW R6 I8-I11**

Exe.3 Tomasulo CC4

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	**1**	**2**			RS1	LDU1
I2: FADD $F2, $F2, $F3	**2**	**3**			RS4	FPU1
I3: ADDI $R3, $R3, 8	**3**	**4**			RS7	ALU1
I4: LD $F4, 0(R2)	**4**				RS2	
I5: FADD $F5, $F4, $F2						
I6: FMULT $F6, $F1, $F4						
I7: ADDI $R5, $R5, 1						
I8: LD $R6, 0($R4)						
I9: SD $F6, 0($R5)						
I10: SD $F5, 0($R1)						
I11: ADD $R1, $R6, $R1						

RAW F0 I1-I6 **RAW F2 I2-I5** **RAW F4 I4-I6** **RAW F4 I4-I5**

RAW F5 I5-I10 **RAW F6 I6-I9** **RAW R5 I7-I9** **RAW R6 I8-I11**

Exe.3 Tomasulo CC5

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	**1**	**2**	**5**		RS1	LDU1
I2: FADD $F2, $F2, $F3	**2**	**3**			RS4	FPU1
I3: ADDI $R3, $R3, 8	**3**	**4**		**Struct CDB**	RS7	ALU1
I4: LD $F4, 0(R2)	**4**	**5**			RS2	LDU2
I5: FADD $F5, $F4, $F2						
I6: FMULT $F6, $F1, $F4						
I7: ADDI $R5, $R5, 1						
I8: LD $R6, 0($R4)						
I9: SD $F6, 0($R5)						
I10: SD $F5, 0($R1)						
I11: ADD $R1, $R6, $R1						

RAW F0 I1-I6 **RAW F2 I2-I5** **RAW F4 I4-I6** **RAW F4 I4-I5**

RAW F5 I5-I10 **RAW F6 I6-I9** **RAW R5 I7-I9** **RAW R6 I8-I11**

Exe.3 Tomasulo CC6

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	**1**	**2**	**5**		RS1	LDU1
I2: FADD $F2, $F2, $F3	**2**	**3**	**6**		RS4	FPU1
I3: ADDI $R3, $R3, 8	**3**	**4**		**Struct CDB**	RS7	ALU1
I4: LD $F4, 0(R2)	**4**	**5**			RS2	LDU2
I5: FADD $F5, $F4, $F2	**5**			**RAW $F4, RAW $F2**	RS5	
I6: FMULT $F6, $F1, $F4	**6**				RS6	
I7: ADDI $R5, $R5, 1						
I8: LD $R6, 0($R4)						
I9: SD $F6, 0($R5)						
I10: SD $F5, 0($R1)						
I11: ADD $R1, $R6, $R1						

RAW F0 I1-I6 **RAW F2 I2-I5** **RAW F4 I4-I6** **RAW F4 I4-I5**

RAW F5 I5-I10 **RAW F6 I6-I9** **RAW R5 I7-I9** **RAW R6 I8-I11**

Exe.3 Tomasulo CC7

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	1	2	5		RS1	LDU1
I2: FADD $F2, $F2, $F3	2	3	6		RS4	FPU1
I3: ADDI $R3, $R3, 8	3	4	7	Struct CDB	RS7	ALU1
I4: LD $F4, 0(R2)	4	5			RS2	LDU2
I5: FADD $F5, $F4, $F2	5			RAW $F4, RAW $F2	RS5	
I6: FMULT $F6, $F1, $F4	6			RAW $F4	RS6	
I7: ADDI $R5, $R5, 1	7				RS8	
I8: LD $R6, 0($R4)						
I9: SD $F6, 0($R5)						
I10: SD $F5, 0($R1)						
I11: ADD $R1, $R6, $R1						

RAW F0 I1-I6 RAW F2 I2-I5 RAW F4 I4-I6 RAW F4 I4-I5

RAW F5 I5-I10 RAW F6 I6-I9 RAW R5 I7-I9 RAW R6 I8-I11

Exe.3 Tomasulo CC8

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	1	2	5		RS1	LDU1
I2: FADD $F2, $F2, $F3	2	3	6		RS4	FPU1
I3: ADDI $R3, $R3, 8	3	4	7	Struct CDB	RS7	ALU1
I4: LD $F4, 0(R2)	4	5	8		RS2	LDU2
I5: FADD $F5, $F4, $F2	5			RAW $F4, RAW $F2	RS5	
I6: FMULT $F6, $F1, $F4	6			RAW $F4	RS6	
I7: ADDI $R5, $R5, 1	7	8			RS8	ALU1
I8: LD $R6, 0($R4)	8				RS1	
I9: SD $F6, 0($R5)						
I10: SD $F5, 0($R1)						
I11: ADD $R1, $R6, $R1						

RAW F0 I1-I6 RAW F2 I2-I5 RAW F4 I4-I6 RAW F4 I4-I5

RAW F5 I5-I10 RAW F6 I6-I9 RAW R5 I7-I9 RAW R6 I8-I11

Exe.3 Tomasulo CC9

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	1	2	5		RS1	LDU1
I2: FADD $F2, $F2, $F3	2	3	6		RS4	FPU1
I3: ADDI $R3, $R3, 8	3	4	7	Struct CDB	RS7	ALU1
I4: LD $F4, 0(R2)	4	5	8		RS2	LDU2
I5: FADD $F5, $F4, $F2	5	9		RAW $F4, RAW $F2	RS5	FPU1
I6: FMULT $F6, $F1, $F4	6	9		RAW $F4	RS6	FPU2
I7: ADDI $R5, $R5, 1	7	8	9		RS8	ALU1
I8: LD $R6, 0($R4)	8	9			RS1	LDU1
I9: SD $F6, 0($R5)	9			RAW $F6	RS2	
I10: SD $F5, 0($R1)						
I11: ADD $R1, $R6, $R1						

RAW F0 I1-I6 RAW F2 I2-I5 RAW F4 I4-I6 RAW F4 I4-I5

RAW F5 I5-I10 RAW F6 I6-I9 RAW R5 I7-I9 RAW R6 I8-I11

Exe.3 Tomasulo CC10

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	1	2	5		RS1	LDU1
I2: FADD $F2, $F2, $F3	2	3	6		RS4	FPU1
I3: ADDI $R3, $R3, 8	3	4	7	Struct CDB	RS7	ALU1
I4: LD $F4, 0(R2)	4	5	8		RS2	LDU2
I5: FADD $F5, $F4, $F2	5	9		RAW $F4, RAW $F2	RS5	FPU1
I6: FMULT $F6, $F1, $F4	6	9		RAW $F4	RS6	FPU2
I7: ADDI $R5, $R5, 1	7	8	9		RS8	ALU1
I8: LD $R6, 0($R4)	8	9			RS1	LDU1
I9: SD $F6, 0($R5)	9			RAW $F6	RS2	
I10: SD $F5, 0($R1)	10			RAW $F5	RS3	
I11: ADD $R1, $R6, $R1						

RAW F0 I1-I6 RAW F2 I2-I5 RAW F4 I4-I6 RAW F4 I4-I5

RAW F5 I5-I10 RAW F6 I6-I9 RAW R5 I7-I9 RAW R6 I8-I11

Exe.3 Tomasulo CC11

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	1	2	5		RS1	LDU1
I2: FADD $F2, $F2, $F3	2	3	6		RS4	FPU1
I3: ADDI $R3, $R3, 8	3	4	7	Struct CDB	RS7	ALU1
I4: LD $F4, 0(R2)	4	5	8		RS2	LDU2
I5: FADD $F5, $F4, $F2	5	9		RAW $F4, RAW $F2	RS5	FPU1
I6: FMULT $F6, $F1, $F4	6	9		RAW $F4	RS6	FPU2
I7: ADDI $R5, $R5, 1	7	8	9		RS8	ALU1
I8: LD $R6, 0($R4)	8	9			RS1	LDU1
I9: SD $F6, 0($R5)	9			RAW $F6	RS2	
I10: SD $F5, 0($R1)	10			RAW $F5	RS3	
I11: ADD $R1, $R6, $R1	11			RAW $R6	RS7	ALU1

RAW F0 I1-I6 RAW F2 I2-I5 RAW F4 I4-I6 RAW F4 I4-I5

RAW F5 I5-I10 RAW F6 I6-I9 RAW R5 I7-I9 RAW R6 I8-I11

Exe.3 Tomasulo CC12

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	1	2	5		RS1	LDU1
I2: FADD $F2, $F2, $F3	2	3	6		RS4	FPU1
I3: ADDI $R3, $R3, 8	3	4	7	Struct CDB	RS7	ALU1
I4: LD $F4, 0(R2)	4	5	8		RS2	LDU2
I5: FADD $F5, $F4, $F2	5	9	12	RAW $F4, RAW $F2	RS5	FPU1
I6: FMULT $F6, $F1, $F4	6	9		RAW $F4, Struct CDB	RS6	FPU2
I7: ADDI $R5, $R5, 1	7	8	9		RS8	ALU1
I8: LD $R6, 0($R4)	8	9		Struct CDB	RS1	LDU1
I9: SD $F6, 0($R5)	9			RAW $F6	RS2	
I10: SD $F5, 0($R1)	10			RAW $F5	RS3	
I11: ADD $R1, $R6, $R1	11			RAW $R6	RS7	ALU1

RAW F0 I1-I6 RAW F2 I2-I5 RAW F4 I4-I6 RAW F4 I4-I5

RAW F5 I5-I10 RAW F6 I6-I9 RAW R5 I7-I9 RAW R6 I8-I11

Exe.3 Tomasulo CC13

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	1	2	5		RS1	LDU1
I2: FADD $F2, $F2, $F3	2	3	6		RS4	FPU1
I3: ADDI $R3, $R3, 8	3	4	7	Struct CDB	RS7	ALU1
I4: LD $F4, 0(R2)	4	5	8		RS2	LDU2
I5: FADD $F5, $F4, $F2	5	9	12	RAW $F4, RAW $F2	RS5	FPU1
I6: FMULT $F6, $F1, $F4	6	9	13	RAW $F4, Struct CDB	RS6	FPU2
I7: ADDI $R5, $R5, 1	7	8	9		RS8	ALU1
I8: LD $R6, 0($R4)	8	9		Struct CDB	RS1	LDU1
I9: SD $F6, 0($R5)	9			RAW $F6	RS2	
I10: SD $F5, 0($R1)	10	13		RAW $F5	RS3	LDU2
I11: ADD $R1, $R6, $R1	11			RAW $R6	RS7	ALU1

RAW F0 I1-I6 RAW F2 I2-I5 RAW F4 I4-I6 RAW F4 I4-I5

RAW F5 I5-I10 RAW F6 I6-I9 RAW R5 I7-I9 RAW R6 I8-I11

Exe.3 Tomasulo CC14

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	1	2	5		RS1	LDU1
I2: FADD $F2, $F2, $F3	2	3	6		RS4	FPU1
I3: ADDI $R3, $R3, 8	3	4	7	Struct CDB	RS7	ALU1
I4: LD $F4, 0(R2)	4	5	8		RS2	LDU2
I5: FADD $F5, $F4, $F2	5	9	12	RAW $F4, RAW $F2	RS5	FPU1
I6: FMULT $F6, $F1, $F4	6	9	13	RAW $F4, Struct CDB	RS6	FPU2
I7: ADDI $R5, $R5, 1	7	8	9		RS8	ALU1
I8: LD $R6, 0($R4)	8	9	14	Struct CDB	RS1	LDU1
I9: SD $F6, 0($R5)	9	14		RAW $F6	RS2	LDU3
I10: SD $F5, 0($R1)	10	13		RAW $F5	RS3	LDU2
I11: ADD $R1, $R6, $R1	11			RAW $R6	RS7	ALU1

RAW F0 I1-I6 RAW F2 I2-I5 RAW F4 I4-I6 RAW F4 I4-I5

RAW F5 I5-I10 RAW F6 I6-I9 RAW R5 I7-I9 RAW R6 I8-I11

Exe.3 Tomasulo CC15

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	1	2	5		RS1	LDU1
I2: FADD $F2, $F2, $F3	2	3	6		RS4	FPU1
I3: ADDI $R3, $R3, 8	3	4	7	Struct CDB	RS7	ALU1
I4: LD $F4, 0(R2)	4	5	8		RS2	LDU2
I5: FADD $F5, $F4, $F2	5	9	12	RAW $F4, RAW $F2	RS5	FPU1
I6: FMULT $F6, $F1, $F4	6	9	13	RAW $F4, Struct CDB	RS6	FPU2
I7: ADDI $R5, $R5, 1	7	8	9		RS8	ALU1
I8: LD $R6, 0($R4)	8	9	14	Struct CDB	RS1	LDU1
I9: SD $F6, 0($R5)	9	14		RAW $F6	RS2	LDU3
I10: SD $F5, 0($R1)	10	13		RAW $F5	RS3	LDU2
I11: ADD $R1, $R6, $R1	11	15		RAW $R6	RS7	ALU1

RAW F0 I1-I6 RAW F2 I2-I5 RAW F4 I4-I6 RAW F4 I4-I5

RAW F5 I5-I10 RAW F6 I6-I9 RAW R5 I7-I9 RAW R6 I8-I11

Exe.3 Tomasulo CC16

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	1	2	5		RS1	LDU1
I2: FADD $F2, $F2, $F3	2	3	6		RS4	FPU1
I3: ADDI $R3, $R3, 8	3	4	7	Struct CDB	RS7	ALU1
I4: LD $F4, 0(R2)	4	5	8		RS2	LDU2
I5: FADD $F5, $F4, $F2	5	9	12	RAW $F4, RAW $F2	RS5	FPU1
I6: FMULT $F6, $F1, $F4	6	9	13	RAW $F4, Struct CDB	RS6	FPU2
I7: ADDI $R5, $R5, 1	7	8	9		RS8	ALU1
I8: LD $R6, 0($R4)	8	9	14	Struct CDB	RS1	LDU1
I9: SD $F6, 0($R5)	9	14		RAW $F6	RS2	LDU3
I10: SD $F5, 0($R1)	10	13	16	RAW $F5	RS3	LDU2
I11: ADD $R1, $R6, $R1	11	15		RAW $R6, Struct CDB	RS7	ALU1

RAW F0 I1-I6 RAW F2 I2-I5 RAW F4 I4-I6 RAW F4 I4-I5

RAW F5 I5-I10 RAW F6 I6-I9 RAW R5 I7-I9 RAW R6 I8-I11

Exe.3 Tomasulo CC17

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	1	2	5		RS1	LDU1
I2: FADD $F2, $F2, $F3	2	3	6		RS4	FPU1
I3: ADDI $R3, $R3, 8	3	4	7	Struct CDB	RS7	ALU1
I4: LD $F4, 0(R2)	4	5	8		RS2	LDU2
I5: FADD $F5, $F4, $F2	5	9	12	RAW $F4, RAW $F2	RS5	FPU1
I6: FMULT $F6, $F1, $F4	6	9	13	RAW $F4, Struct CDB	RS6	FPU2
I7: ADDI $R5, $R5, 1	7	8	9		RS8	ALU1
I8: LD $R6, 0($R4)	8	9	14	Struct CDB	RS1	LDU1
I9: SD $F6, 0($R5)	9	14	17	RAW $F6	RS2	LDU3
I10: SD $F5, 0($R1)	10	13	16	RAW $F5	RS3	LDU2
I11: ADD $R1, $R6, $R1	11	15		RAW $R6, Struct CDB	RS7	ALU1

RAW F0 I1-I6 RAW F2 I2-I5 RAW F4 I4-I6 RAW F4 I4-I5

RAW F5 I5-I10 RAW F6 I6-I9 RAW R5 I7-I9 RAW R6 I8-I11

Exe.3 Tomasulo CC18

- 3 RESERVATION STATIONS (RS1, RS2, RS3) + 3 LOAD/STORE unit (LDU1, LDU2, LDU3) with latency 3
- 3 RESERVATION STATIONS (RS4, RS5, RS6) + 3 FPUs (FPU1, FPU2, FPU3) with latency 3
- 2 RESERVATION STATIONS (RS7, RS8) + 1 Integer ALU (ALU1) with latency 1

Instruction	ISSUE	START EXE	WB	Hazards Type	RSi	Unit
I1: LD $F1, 0($R1)	1	2	5		RS1	LDU1
I2: FADD $F2, $F2, $F3	2	3	6		RS4	FPU1
I3: ADDI $R3, $R3, 8	3	4	7	Struct CDB	RS7	ALU1
I4: LD $F4, 0(R2)	4	5	8		RS2	LDU2
I5: FADD $F5, $F4, $F2	5	9	12	RAW $F4, RAW $F2	RS5	FPU1
I6: FMULT $F6, $F1, $F4	6	9	13	RAW $F4, Struct CDB	RS6	FPU2
I7: ADDI $R5, $R5, 1	7	8	9		RS8	ALU1
I8: LD $R6, 0($R4)	8	9	14	Struct CDB	RS1	LDU1
I9: SD $F6, 0($R5)	9	14	17	RAW $F6	RS2	LDU3
I10: SD $F5, 0($R1)	10	13	16	RAW $F5	RS3	LDU2
I11: ADD $R1, $R6, $R1	11	15	18	RAW $R6, Struct CDB	RS7	ALU1

RAW F0 I1-I6 RAW F2 I2-I5 RAW F4 I4-I6 RAW F4 I4-I5

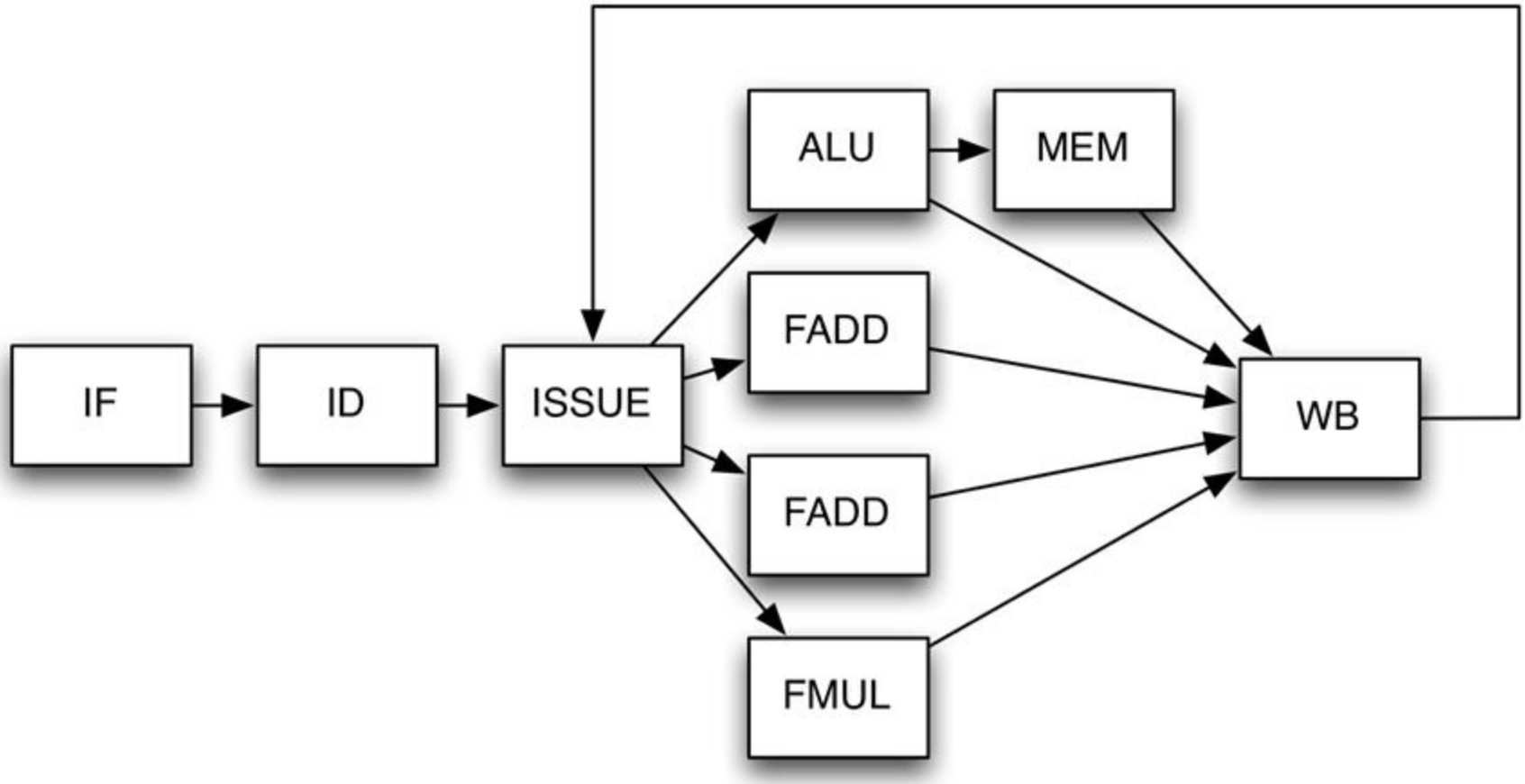
RAW F5 I5-I10 RAW F6 I6-I9 RAW R5 I7-I9 RAW R6 I8-I11

Scoreboard vs Tomasulo

	Instruction	ISSUE	READ OPERAND	EXE COMPLETE	WB
I1	LD $F1, 0($R1)	1	2	5	6
I2	FADD $F2, $F2, $F3	2	3	7	8
I3	ADDI $R3, $R3, 8	3	4	5	7
I4	LD $F4, 0(R2)	4	5	8	9
I5	FADD $F5, $F4, $F2	5	10	14	15
I6	FMULT $F6, $F1, $F4	6	10	14	16
I7	ADDI $R5, $R5, 1	7	8	9	10
I8	LD $R6, 0($R4)	8	9	12	13
I9	SD $F6, 0($R5)	9	17	20	21
I10	SD $F5, 0($R1)	10	16	19	20
I11	ADD $R1, $R6, $R1	11	14	15	17

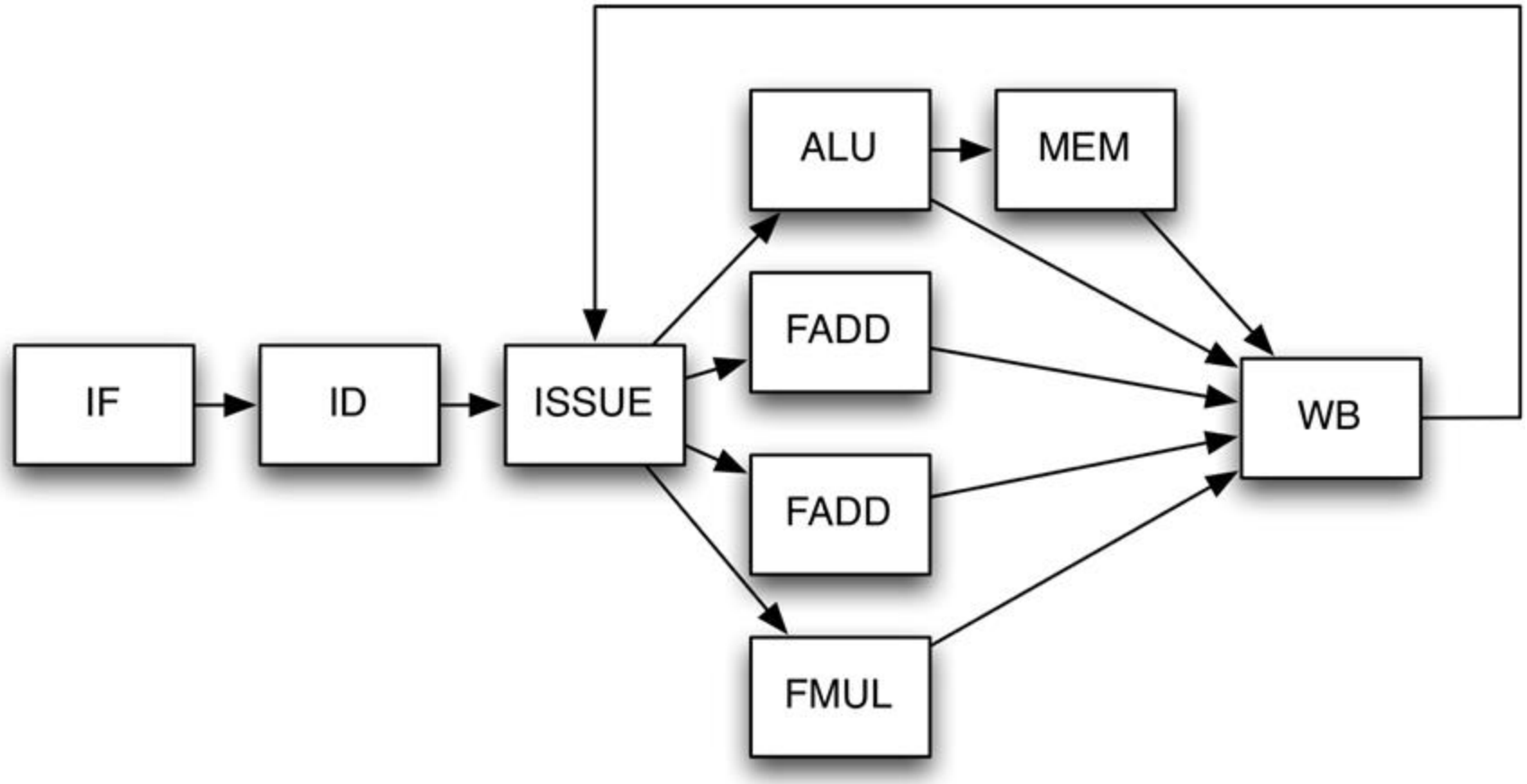
ISSUE	START EXE	WB
1	2	5
2	3	6
3	4	7
4	5	8
5	9	12
6	9	13
7	8	9
8	9	14
9	14	17
10	13	16
11	15	18

Exe: Complex Pipeline

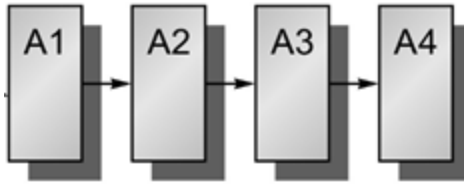


Exe: Complex Pipeline

In this problem we will examine the execution of a code segment on the following single-issue out-of-order processor:

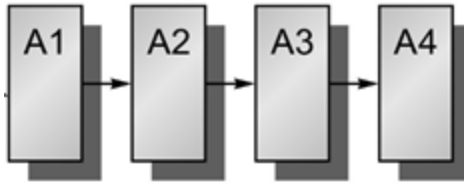


You can assume that



- All functional units are pipelined
- ALU operations take 1 cycle
- Memory operations take 2 cycles (includes time in ALU)
- Floating-point add instructions take 2 cycles
- Floating-point multiply instructions take 3 cycles
- There is no register renaming. No forwarding
- Instructions are fetched, decoded and issued in order
- The ISSUE stage is a buffer of unlimited length that holds instructions waiting to start execution
- An instruction will only enter the issue stage if it does not cause a WAR or WAW hazard
- Only one instruction can be issued at a time, and in the case multiple instructions are ready, the oldest one will go first
- Program Counter calculation for branches and jumps has been anticipated in the ISSUE stage

You can assume that

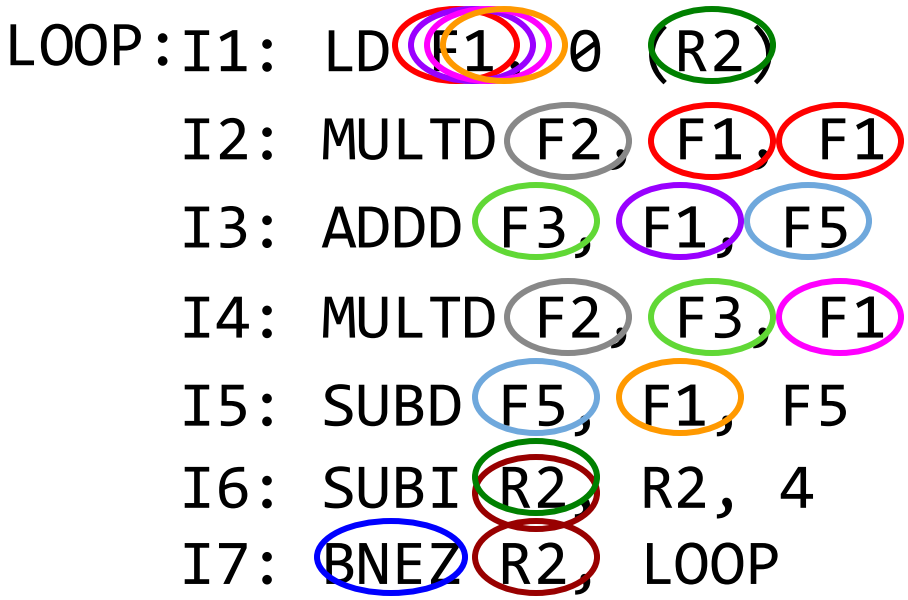


- All functional units are pipelined
- ALU operations take 1 cycle
- Memory operations take 2 cycles (includes time in ALU)
- Floating-point add instructions take 2 cycles
- Floating-point multiply instructions take 3 cycles
- There is no register renaming. No forwarding
- Instructions are fetched, decoded and issued in order
- The ISSUE stage is a buffer of unlimited length that holds instructions waiting to start execution
- An instruction will only enter the issue stage if it does not cause a WAR or WAW hazard
- Only one instruction can be issued at a time, and in the case multiple instructions are ready, the oldest one will go first
- Program Counter calculation for branches and jumps has been anticipated in the ISSUE stage

Exe Complex Pipeline: the Code

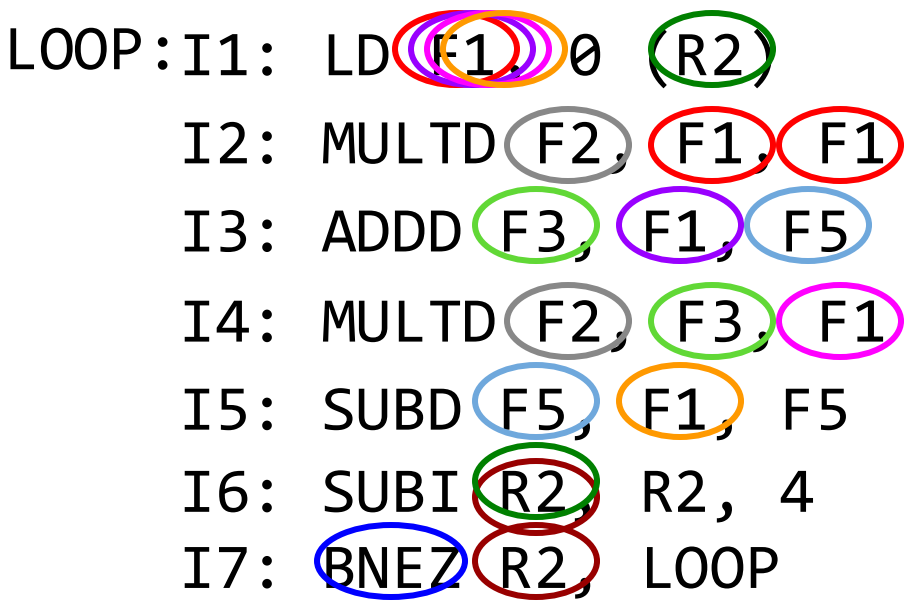
```
LOOP:I1: LD F1, 0 (R2)
     I2: MULTD F2, F1, F1
     I3: ADDD F3, F1, F5
     I4: MULTD F2, F3, F1
     I5: SUBD F5, F1, F5
     I6: SUBI R2, R2, 4
     I7: BNEZ R2, LOOP
```

Exe Complex Pipeline: the Conflicts



RAW F1 I1-I2
RAW F1 I1-I3
RAW F1 I1-I4
RAW F1 I1-I5
RAW F3 I3-I4
RAW R2 I6-I7
WAW F2 I2-I4
WAR F5 I3-I5
WAR R2 I1-I6
CNTRL

Exe Complex Pipeline: the Arch.



RAW F1 I1-I2
RAW F1 I1-I3
RAW F1 I1-I4
RAW F1 I1-I5
RAW F3 I3-I4
RAW R2 I6-I7
WAW F2 I2-I4
WAR F5 I3-I5
WAR R2 I1-I6
CNTRL

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 0

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)																				
2	MULTD F2,F1,F1																				RAW F1 I1-I2
3	ADDD F3,F1,F5																				RAW F1 I1-I3
4	MULTD F2,F3,F1																				RAW F1 I1-I4 RAW F3 I3-I4 WAW F2 I2-I4
5	SUBD F5,F1,F5																				RAW F1 I1-I5 WAR F5 I3-I5
6	SUBI R2,R2,4																				WAR R2 I1-I6
7	BNEZ R2, LOOP																				RAW R2 I6-I7
8	(New Instruction)																				CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 1

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F																			
2	MULTD F2,F1,F1																				RAW F1 I1-I2
3	ADDD F3,F1,F5																				RAW F1 I1-I3
4	MULTD F2,F3,F1																				RAW F1 I1-I4 RAW F3 I3-I4 WAW F2 I2-I4
5	SUBD F5,F1,F5																				RAW F1 I1-I5 WAR F5 I3-I5
6	SUBI R2,R2,4																				WAR R2 I1-I6
7	BNEZ R2, LOOP																				RAW R2 I6-I7
8	(New Instruction)																				CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 2

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F	D																		
2	MULTD F2,F1,F1		F																		RAW F1 I1-I2
3	ADDD F3,F1,F5																				RAW F1 I1-I3
4	MULTD F2,F3,F1																				RAW F1 I1-I4 RAW F3 I3-I4 WAW F2 I2-I4
5	SUBD F5,F1,F5																				RAW F1 I1-I5 WAR F5 I3-I5
6	SUBI R2,R2,4																				WAR R2 I1-I6
7	BNEZ R2, LOOP																				RAW R2 I6-I7
8	(New Instruction)																				CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 3

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F	D	IS																	
2	MULTD F2,F1,F1		F	D																	RAW F1 I1-I2
3	ADDD F3,F1,F5			F																	RAW F1 I1-I3
4	MULTD F2,F3,F1																				RAW F1 I1-I4 RAW F3 I3-I4 WAW F2 I2-I4
5	SUBD F5,F1,F5																				RAW F1 I1-I5 WAR F5 I3-I5
6	SUBI R2,R2,4																				WAR R2 I1-I6
7	BNEZ R2, LOOP																				RAW R2 I6-I7
8	(New Instruction)																				CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 4

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F	D	IS	E1																
2	MULTD F2,F1,F1		F	D	IS s																RAW F1 I1-I2
3	ADDD F3,F1,F5			F	D																RAW F1 I1-I3
4	MULTD F2,F3,F1				F																RAW F1 I1-I4 RAW F3 I3-I4 WAW F2 I2-I4
5	SUBD F5,F1,F5																				RAW F1 I1-I5 WAR F5 I3-I5
6	SUBI R2,R2,4																				WAR R2 I1-I6
7	BNEZ R2, LOOP																				RAW R2 I6-I7
8	(New Instruction)																				CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 5

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F	D	IS	E1	E2															
2	MULTD F2,F1,F1		F	D	IS s	IS s															RAW F1 I1-I2
3	ADDD F3,F1,F5			F	D	IS s															RAW F1 I1-I3
4	MULTD F2,F3,F1				F	D s															RAW F1 I1-I4 RAW F3 I3-I4 WAW F2 I2-I4
5	SUBD F5,F1,F5					F s															RAW F1 I1-I5 WAR F5 I3-I5
6	SUBI R2,R2,4																				WAR R2 I1-I6
7	BNEZ R2, LOOP																				RAW R2 I6-I7
8	(New Instruction)																				CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 6

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F	D	IS	E1	E2	W														
2	MULTD F2,F1,F1		F	D	IS s	IS s	IS														~~RAW F1 I1-I2~~
3	ADDD F3,F1,F5			F	D	IS s	IS s														~~RAW F1 I1-I3~~
4	MULTD F2,F3,F1				F	D s	D s														~~RAW F1 I1-I4~~ RAW F3 I3-I4 WAW F2 I2-I4
5	SUBD F5,F1,F5					F s	F s														~~RAW F1 I1-I5~~ WAR F5 I3-I5
6	SUBI R2,R2,4																				~~WAR R2 I1-I6~~
7	BNEZ R2, LOOP																				RAW R2 I6-I7
8	(New Instruction)																				CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 7

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F	D	IS	E1	E2	W														
2	MULTD F2,F1,F1		F	D	IS s	IS s	IS	E1													~~RAW F1 I1-I2~~
3	ADDD F3,F1,F5			F	D	IS s	IS s	IS													~~RAW F1 I1-I3~~
4	MULTD F2,F3,F1				F	D s	D s	D s													~~RAW F1 I1-I4~~ RAW F3 I3-I4 WAW F2 I2-I4
5	SUBD F5,F1,F5					F s	F s	F s													~~RAW F1 I1-I5~~ WAR F5 I3-I5
6	SUBI R2,R2,4																				~~WAR R2 I1-I6~~
7	BNEZ R2, LOOP																				RAW R2 I6-I7
8	(New Instruction)																				CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 9

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F	D	IS	E1	E2	W														
2	MULTD F2,F1,F1		F	D	ISs	ISs	IS	E1	E2	E3											~~RAW F1 I1 I2~~
3	ADDD F3,F1,F5			F	D	ISs	ISs	IS	E1	E2											~~RAW F1 I1 I3~~ ~~Structural on WB~~
4	MULTD F2,F3,F1				F	Ds	Ds	Ds	Ds	D											~~RAW F1 I1-I4~~ RAW F3 I3-I4 ~~WAW F2 I2 I4~~
5	SUBD F5,F1,F5					Fs	Fs	Fs	Fs	F											~~RAW F1 I1 I5~~ WAR F5 I3-I5
6	SUBI R2,R2,4																				~~WAR R2 I1-I6~~
7	BNEZ R2, LOOP																				RAW R2 I6-I7
8	(New Instruction)																				CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 10

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F	D	IS	E1	E2	W														
2	MULTD F2,F1,F1		F	D	IS s	IS s	IS	E1	E2	E3	W										~~RAW F1 I1 I2~~
3	ADDD F3,F1,F5			F	D	IS s	IS s	IS	E1	E2	E2 s										~~RAW F1 I1 I3~~ ~~Structural on WB~~
4	MULTD F2,F3,F1				F	D s	D s	D s	D s	D	IS s										~~RAW F1 I1-I4~~ RAW F3 I3-I4 ~~WAW F2 I2 I4~~
5	SUBD F5,F1,F5					F s	F s	F s	F s	F	D										~~RAW F1 I1 I5~~ WAR F5 I3-I5
6	SUBI R2,R2,4										F										~~WAR R2 I1-I6~~
7	BNEZ R2, LOOP																				RAW R2 I6-I7
8	(New Instruction)																				CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 11

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F	D	IS	E1	E2	W														
2	MULTD F2,F1,F1		F	D	IS s	IS s	IS	E1	E2	E3	W										~~RAW F1 I1 I2~~
3	ADDD F3,F1,F5			F	D	IS s	IS s	IS	E1	E2	E2 s	W									~~RAW F1 I1 I3~~ ~~Structural on WB~~
4	MULTD F2,F3,F1				F	D s	D s	D s	D s	D	IS s	IS									~~RAW F1 I1-I4~~ ~~RAW F3 I3 I4~~ ~~WAW F2 I2 I4~~
5	SUBD F5,F1,F5					F s	F s	F s	F s	F	D	IS s									~~RAW F1 I1 I5~~ ~~WAR F5 I3 I5~~
6	SUBI R2,R2,4										F	D									~~WAR R2 I1-I6~~
7	BNEZ R2, LOOP											F									RAW R2 I6-I7
8	(New Instruction)																				CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 12

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F	D	IS	E1	E2	W														
2	MULTD F2,F1,F1		F	D	IS s	IS s	IS	E1	E2	E3	W										~~RAW F1 I1 I2~~
3	ADDD F3,F1,F5			F	D	IS s	IS s	IS	E1	E2	E2 s	W									~~RAW F1 I1 I3~~ ~~Structural on WB~~
4	MULTD F2,F3,F1				F	D s	D s	D s	D s	D	IS s	IS	E1								~~RAW F1 I1-I4~~ ~~RAW F3 I3 I4~~ ~~WAW F2 I2 I4~~
5	SUBD F5,F1,F5					F s	F s	F s	F s	F	D	IS s	IS								~~RAW F1 I1 I5~~ ~~WAR F5 I3 I5~~
6	SUBI R2,R2,4										F	D	IS s								~~WAR R2 I1-I6~~
7	BNEZ R2, LOOP											F	D								RAW R2 I6-I7
8	(New Instruction)												F s								CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 14

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F	D	IS	E1	E2	W														
2	MULTD F2,F1,F1		F	D	IS s	IS s	IS	E1	E2	E3	W										~~RAW F1 I1 I2~~
3	ADDD F3,F1,F5			F	D	IS s	IS s	IS	E1	E2	E2 s	W									~~RAW F1 I1 I3~~ Structural on WB
4	MULTD F2,F3,F1				F	D s	D s	D s	D s	D	IS s	IS	E1	E2	E3						~~RAW F1 I1-I4~~ ~~RAW F3 I3 I4~~ ~~WAW F2 I2 I4~~
5	SUBD F5,F1,F5					F s	F s	F s	F s	F	D	IS s	IS	E1	E2						~~RAW F1 I1 I5~~ ~~WAR F5 I3 I5~~ Structural on WB
6	SUBI R2,R2,4										F	D	IS s	IS	E1						~~WAR R2 I1-I6~~
7	BNEZ R2, LOOP											F	D	IS s	IS s						RAW R2 I6-I7
8	(New Instruction)												F s	F s	F s						CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 15

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F	D	IS	E1	E2	W														
2	MULTD F2,F1,F1		F	D	IS s	IS s	IS	E1	E2	E3	W										~~RAW F1 I1 I2~~
3	ADDD F3,F1,F5			F	D	IS s	IS s	IS	E1	E2	E2 s	W									~~RAW F1 I1 I3~~ Structural on WB
4	MULTD F2,F3,F1				F	D s	D s	D s	D s	D	IS s	IS	E1	E2	E3	W					~~RAW F1 I1-I4~~ ~~RAW F3 I3 I4~~ ~~WAW F2 I2 I4~~
5	SUBD F5,F1,F5					F s	F s	F s	F s	F	D	IS s	IS	E1	E2	E2 s					~~RAW F1 I1 I5~~ ~~WAR F5 I3 I5~~ Structural on WB
6	SUBI R2,R2,4										F	D	IS s	IS	E1	E1 s					~~WAR R2 I1-I6~~ Structural on WB
7	BNEZ R2, LOOP											F	D	IS s	IS s	IS s					RAW R2 I6-I7
8	(New Instruction)												F s	F s	F s	F s					CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 16

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F	D	IS	E1	E2	W														
2	MULTD F2,F1,F1		F	D	IS s	IS s	IS	E1	E2	E3	W										~~RAW F1 I1 I2~~
3	ADDD F3,F1,F5			F	D	IS s	IS s	IS	E1	E2	E2 s	W									~~RAW F1 I1 I3~~ Structural on WB
4	MULTD F2,F3,F1				F	D s	D s	D s	D s	D	IS s	IS	E1	E2	E3	W					~~RAW F1 I1-I4~~ ~~RAW F3 I3 I4~~ ~~WAW F2 I2 I4~~
5	SUBD F5,F1,F5					F s	F s	F s	F s	F	D	IS s	IS	E1	E2	E2 s	W				~~RAW F1 I1 I5~~ ~~WAR F5 I3 I5~~ Structural on WB
6	SUBI R2,R2,4										F	D	IS s	IS	E1	E1 s	E1 s				~~WAR R2 I1-I6~~ Structural on WB
7	BNEZ R2, LOOP											F	D	IS s	IS s	IS s	IS s				RAW R2 I6-I7
8	(New Instruction)												F s	F s	F s	F s	F s				CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 17

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F	D	IS	E1	E2	W														
2	MULTD F2,F1,F1		F	D	IS s	IS s	IS	E1	E2	E3	W										~~RAW F1 I1 I2~~
3	ADDD F3,F1,F5			F	D	IS s	IS s	IS	E1	E2	E2 s	W									~~RAW F1 I1 I3~~ Structural on WB
4	MULTD F2,F3,F1				F	D s	D s	D s	D s	D	IS s	IS	E1	E2	E3	W					~~RAW F1 I1-I4~~ ~~RAW F3 I3 I4~~ ~~WAW F2 I2 I4~~
5	SUBD F5,F1,F5					F s	F s	F s	F s	F	D	IS s	IS	E1	E2	E2 s	W				~~RAW F1 I1 I5~~ ~~WAR F5 I3 I5~~ Structural on WB
6	SUBI R2,R2,4										F	D	IS s	IS	E1	E1 s	E1 s	W			~~WAR R2 I1-I6~~ Structural on WB
7	BNEZ R2, LOOP											F	D	IS s	IS s	IS s	IS s	IS			~~RAW R2 I6 I7~~
8	(New Instruction)												F s	F s	F s	F s	F s	F s			CNTRL

Pipeline Schema

ALU OP: 1 cycle
MEM OP: 2 cycles
FP ADD: 2 cycles
FP MULT: 3 cycles

CC 19

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	Notes
1	LOOP: LD F1,0(R2)	F	D	IS	E1	E2	W														
2	MULTD F2,F1,F1		F	D	IS s	IS s	IS	E1	E2	E3	W										~~RAW F1 I1 I2~~
3	ADDD F3,F1,F5			F	D	IS s	IS s	IS	E1	E2	E2 s	W									~~RAW F1 I1 I3~~ ~~Structural on WB~~
4	MULTD F2,F3,F1				F	D s	D s	D s	D s	D	IS s	IS	E1	E2	E3	W					~~RAW F1 I1-I4~~ ~~RAW F3 I3 I4~~ ~~WAW F2 I2 I4~~
5	SUBD F5,F1,F5					F s	F s	F s	F s	F	D	IS s	IS	E1	E2	E2 s	W				~~RAW F1 I1 I5~~ ~~WAR F5 I3 I5~~ ~~Structural on WB~~
6	SUBI R2,R2,4										F	D	IS s	IS	E1	E1 s	E1 s	W			~~WAR R2 I1-I6~~ ~~Structural on WB~~
7	BNEZ R2, LOOP											F	D	IS s	IS s	IS s	IS s	IS	E1	W	~~RAW R2 I6 I7~~
8	(New Instruction)												F s	F s	F s	F s	F s	F s	F	D	~~CNTRL~~

Thank you for your attention
Questions?

Alessandro Verosimile <alessandro.verosimile@polimi.it>

Acknowledgements

Davide Conficconi, E. Del Sozzo, Marco D. Santambrogio, D. Sciuto

Part of this material comes from:

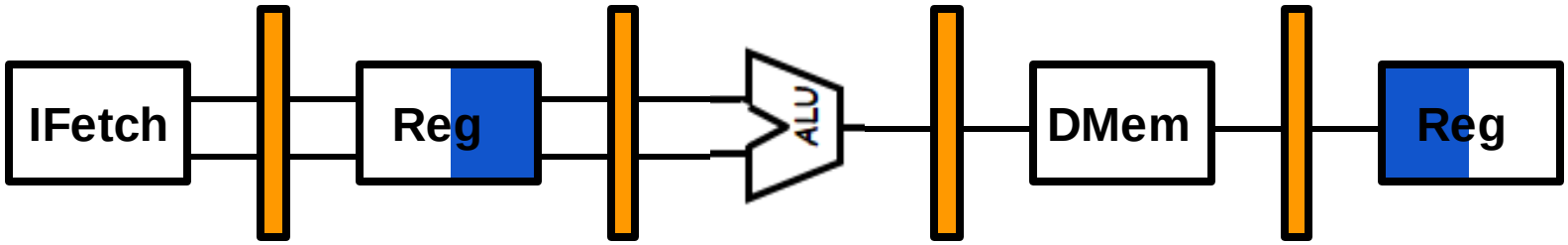
- “Computer Organization and Design” and “Computer Architecture A Quantitative Approach” Patterson and Hennessy books
- News and paper cited throughout the lecture

Exe 3: Simple Pipelining

Exe 3 Simple Pipelining : the Code

```
I1:  addi $s3, $s2, 2
I2:  add $s5, $s4, $s3
I3:  sw $s5, 4($s3)
I4:  sub $s7, $s5, $s6
I5:  lw $s6, 4($s7)
```

Exe 3: Simple Pipelining: the Architecture



Exe 3.1 Simple Pipelining : Conflicts

	Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
I1:	`addi $s3, $s2, 2`	**F**	**D**	**E**	**M**	**W**										
I2:	`add $s5, $s4, $s3`		**F**	**D**	**E**	**M**	**W**									
I3:	`sw $s5, 4($s3)`			**F**	**D**	**E**	**M**	**W**								
I4:	`sub $s7, $s5, $s6`				**F**	**D**	**E**	**M**	**W**							
I5:	`lw $s6, 4($s7)`					**F**	**D**	**E**	**M**	**W**						

Draw the pipeline schema showing all the conflicts/dependencies.
Solve the resulting RAW hazards without using rescheduling and path forwarding.

Exe 3.1 Simple Pipelining : solve as is

Istr	CK1	CK2	CK3	CK4	CK5	CK6	CK7	CK8	CK9	CK10	CK11
I1											
I2											
I3											
I4											
I5											

Istr	CK12	CK13	CK14	CK15	CK16	CK17	CK18	CK19	CK20	CK21	CK22
I1											
I2											
I3											
I4											
I5											

```
I1:  addi $s3, $s2, 2
I2:  sub $s4, $s3, $s1
I3:  add $s5, $s4, $s1
I4:  lw $s6, 4($s4)
I5:  sub $s7, $s4, $s6
```

Exe 3.2 Simple Pipelining : Rescheduling

Reschedule the instructions to **reduce the stalls**; Draw the pipeline schema showing all the data conflicts/dependencies.

Exe 3.3 Simple Pipelining : FWD Paths

Istr	CK1	CK2	CK3	CK4	CK5	CK6	CK7	CK8	CK9	CK10	CK11
I1											
I2											
I3											
I4											
I5											
Istr	CK12	CK13	CK14	CK15	CK16	CK17	CK18	CK19	CK20	CK21	CK22
I1											
I2											
I3											
I4											
I5											

```
I1:  addi $s3, $s2, 2
I2:  sub $s4, $s3, $s1
I3:  add $s5, $s4, $s1
I4:  lw $s6, 4($s4)
I5:  sub $s7, $s4, $s6
```

Exe 3.1 Simple Pipelining : Conflicts

	Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
I1:	`addi $s3, $s2, 2`	**F**	**D**	**E**	**M**	**W**										
I2:	`add $s5, $s4, $s3`		**F**	**D**	**E**	**M**	**W**									
I3:	`sw $s5, 4($s3)`			**F**	**D**	**E**	**M**	**W**								
I4:	`sub $s7, $s5, $s6`				**F**	**D**	**E**	**M**	**W**							
I5:	`lw $s6, 4($s7)`					**F**	**D**	**E**	**M**	**W**						

Draw the pipeline schema showing all the conflicts/dependencies.
Solve the resulting RAW hazards without using rescheduling and path forwarding.

Exe 3.1 Simple Pipelining : Conflicts

	Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
I1:	`addi $s3, $s2, 2`	F	D	E	M	W										
I2:	`add $s5, $s4, $s3`		F	D	E	M	W									
I3:	`sw $s5, 4($s3)`			F	D	E	M	W								
I4:	`sub $s7, $s5, $s6`				F	D	E	M	W							
I5:	`lw $s6, 4($s7)`					F	D	E	M	W						

Draw the pipeline schema showing all the conflicts/dependencies.
Solve the resulting RAW hazards without using rescheduling and path forwarding.

Exe 3.1 Simple Pipelining : Hazards

	Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
I1:	`addi $s3, $s2, 2`	F	D	E	M	W										
I2:	`add $s5, $s4, $s3`		F	D	E	M	W									
I3:	`sw $s5, 4($s3)`			F	D	E	M	W								
I4:	`sub $s7, $s5, $s6`				F	D	E	M	W							
I5:	`lw $s6, 4($s7)`					F	D	E	M	W						

Draw the pipeline schema showing all the conflicts/dependencies.
Solve the resulting RAW hazards without using rescheduling and path forwarding.

RAW $s3 I1-I2

RAW $s3 I1-I3

RAW $s5 I2-I3

RAW $s5 I2-I4

RAW $s7 I4-I5

Exe 3.1 Simple Pipelining : solve as is

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16
1	addi $s3, $s2, 2	F	D	E	M	W											
2	add $s5, $s4, $s3		F	D	E	M	W										
3	sw $s5, 4($s3)			F	D	E	M	W									
4	sub $s7, $s5, $s6				F	D	E	M	W								
5	lw $s6, 4($s7)					F	D	E	M	W							

RAW $s3 I1-I2

RAW $s3 I1-I3

RAW $s5 I2-I3

RAW $s5 I2-I4

RAW $s7 I4-I5

Exe 3.1 Simple Pipelining : solve as is

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16
1	addi $s3, $s2, 2	F	D	E	M	W											
2	add $s5, $s4, $s3		F	D(s)	D(s)	D	E	M	W								
3	sw $s5, 4($s3)			F(s)	F(s)	F	D(s)	D(s)	D	E	M	W					
4	sub $s7, $s5, $s6						F(s)	F(s)	F	D	E	M	W				
5	lw $s6, 4($s7)									F	D(s)	D(s)	D	E	M	W	

RAW $s3 I1-I2

RAW $s3 I1-I3

RAW $s5 I2-I3

RAW $s5 I2-I4

RAW $s7 I4-I5

Exe 3.2 Simple Pipelining : Rescheduling

Reschedule the instructions to **reduce the stalls**; Draw the pipeline schema showing all the data conflicts/dependencies.

Exe 3.2 Simple Pipelining : Rescheduling

Reschedule the instructions to reduce the stalls; Draw the pipeline schema showing all the data conflicts/dependencies.

```
I1:  addi $s3, $s2, 2
I2:  add $s5, $s4, $s3
I3:  sw $s5, 4($s3)
I4:  sub $s7, $s5, $s6
I5:  lw $s6, 4($s7)
```

RAW $s3 I1-I2

RAW $s3 I1-I3

RAW $s5 I2-I3

RAW $s5 I2-I4

RAW $s7 I4-I5

Exe 3.2 Simple Pipelining : Rescheduling

Reschedule the instructions to reduce the stalls; Draw the pipeline schema showing all the data conflicts/dependencies.

```
I1:  addi $s3, $s2, 2
I2:  add $s5, $s4, $s3
I4:  sub $s7, $s5, $s6
I3:  sw $s5, 4($s3)
I5:  lw $s6, 4($s7)
```

RAW $s3 I1-I2

RAW $s3 I1-I3

RAW $s5 I2-I3

RAW $s5 I2-I4

RAW $s7 I4-I5

Exe 3.2 Simple Pipelining : Rescheduling RAW

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16
1	addi $s3, $s2, 2	F	D	E	M	W											
2	add $s5, $s4, $s3		F	D	E	M	W										
4	sub $s7, $s5, $s6			F	D	E	M	W									
3	sw $s5, 4($s3)				F	D	E	M	W								
5	lw $s6, 4($s7)					F	D	E	M	W							

RAW $s3 I1-I2

~~RAW $s3 I1-I3~~

RAW $s5 I2-I3

RAW $s5 I2-I4

RAW $s7 I4-I5

Exe 3.2 Simple Pipelining : Rescheduling Execution

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16
1	addi $s3, $s2, 2	F	D	E	M	W											
2	add $s5, $s4, $s3		F	D(s)	D(s)	D	E	M	W								
4	sub $s7, $s5, $s6			F(s)	F(s)	F	D(s)	D(s)	D	E	M	W					
3	sw $s5, 4($s3)						F(s)	F(s)	F	D	E	M	W				
5	lw $s6, 4($s7)									F	D(s)	D	E	M	W		

RAW $s3 I1-I2

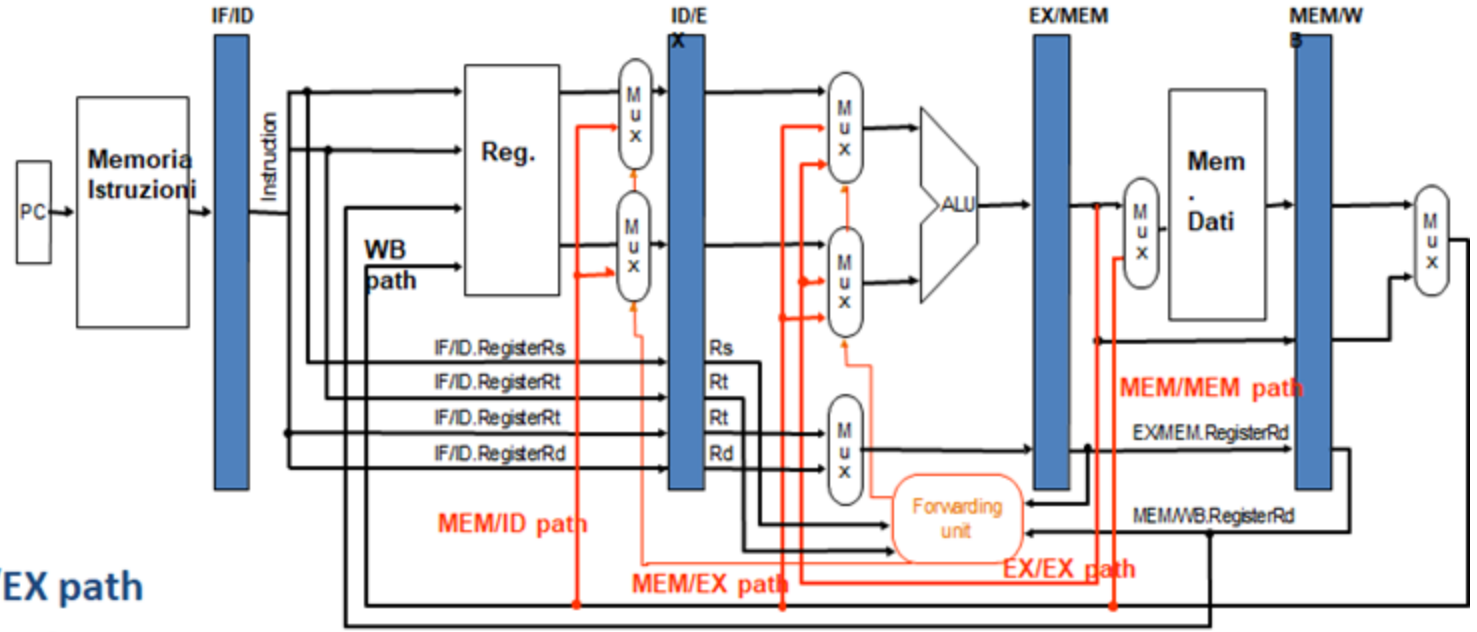
~~RAW $s3 I1-I3~~

RAW $s5 I2-I3

RAW $s5 I2-I4

RAW $s7 I4-I5

Exe 3.3: Forwarding paths



- EX/EX path
- MEM/EX path
- MEM/ID path
- MEM/MEM path

The forwarding paths have been included in the pipeline. Start from the code in (a) and draw the pipeline schema showing all the forwarding paths that have to be used to solve the hazards.

Exe 3.3 Simple Pipelining : FWD Paths

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16
1	addi $s3, $s2, 2	F	D	E	M	W											
2	add $s5, $s4, $s3		F	D	E	M	W										
3	sw $s5, 4($s3)			F	D	E	M	W									
4	sub $s7, $s5, $s6				F	D	E	M	W								
5	lw $s6, 4($s7)					F	D	E	M	W							

RAW $s3 I1-I2

RAW $s3 I1-I3

RAW $s5 I2-I3

RAW $s5 I2-I4

RAW $s7 I4-I5

Exe 3.3 Simple Pipelining : FWD Paths

	Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	FWD Path
1	addi $s3, $s2, 2	F	D	E	M	W											
2	add $s5, $s4, $s3		F	D	E	M	W										EX-EX
3	sw $s5, 4($s3)			F	D	E	M	W									M-EX M-M
4	sub $s7, $s5, $s6				F	D	E	M	W								M-EX
5	lw $s6, 4($s7)					F	D	E	M	W							EX-EX

RAW $s3 I1-I2

RAW $s3 I1-I3

RAW $s5 I2-I3

RAW $s5 I2-I4

RAW $s7 I4-I5