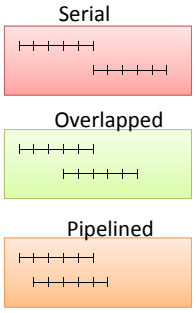
Pipeline and Hazards
Data Forwarding and Pipeline Scheduling

Dr. A. Sahu
CSE IIT Guwahati

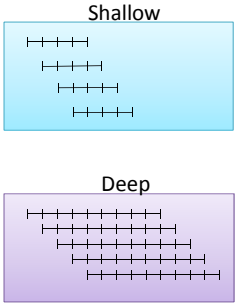
Types of Pipelined processors

- Degree of overlap
  - Serial, Overlapped, Pipelined,
  - Super-pipelined/Superscalar
- Depth
  - Shallow, Deep
- Structure
  - Linear, Non - linear
- Scheduling of operations
  - Static, Dynamic

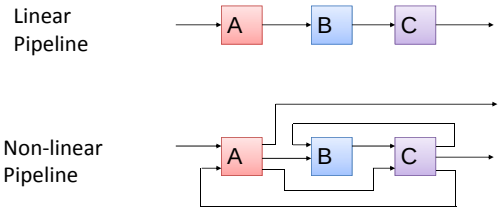
Degree of overlap



Depth



Pipeline Structure



Sequence: A, B, C, B, C, A, C, A

Scheduling/timing alternatives

- Static
  - same sequence of stages for all instructions
  - all actions in order
  - if one instruction stalls, all subsequent instructions are delayed
- Dynamic
  - above conditions are relaxed
  - higher throughput is achieved

Dynamic Scheduling

- type 1 : beginnings (decode) and endings (put away) in order
- type 2 : only beginnings in order
- type 3 : no order restrictions except dependencies
- type 1 extended : beginnings in order, references that effect memory state are in order

  [note that a memory reference may lead to page fault]

Pipelining and CPI

Type	CPI
Serial	5 – 6
Overlapped	3
Pipelined (static)	1.5 – 2
Pipelined (dynamic)	1.2 – 1.5
Multiple instruction issue	< 1.0

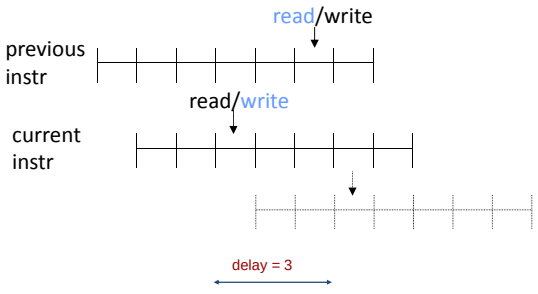
A Sahu slide 7

Hazards in Pipelining

- Data dependencies => Data hazards
  - RAW (read after write)
  - WAR (write after read)
  - WAW (write after write)
- Resource conflicts => Structural hazards
  - use of same resource in different stages
- Procedural dependencies => Control hazards
  - conditional and unconditional branches, calls/returns

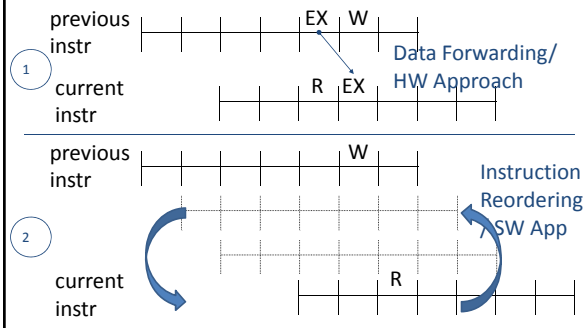
A Sahu slide 8

Data Hazards



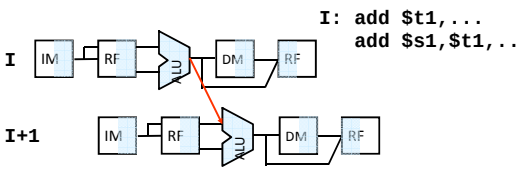
A Sahu slide 9

Handling Data Hazards

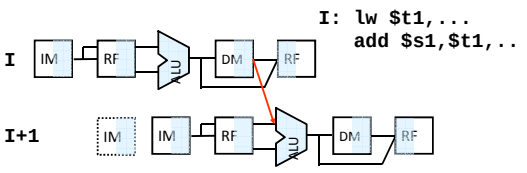


A Sahu slide 10

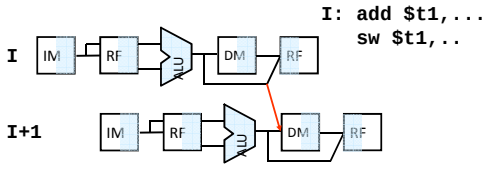
Data forwarding path P1



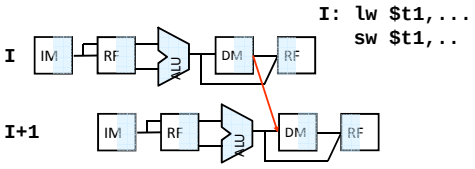
Data forwarding path P2



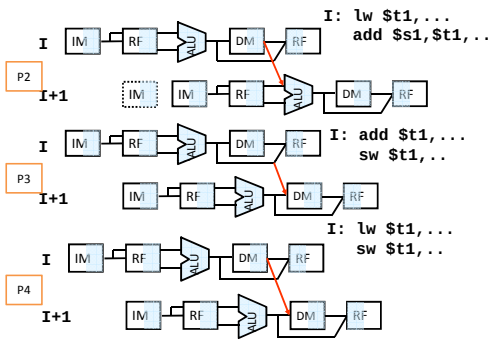
Data forwarding path P3



Data forwarding path P4



Data forwarding paths

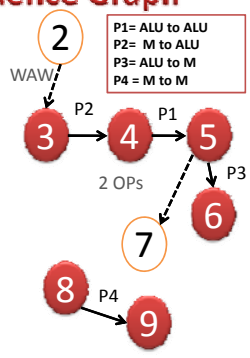


Data forwarding path list

- P1
  from ALU out (EX/DM) to ALU in1/2
- P2
  from DM/ALU out (DM/WB) to ALU in1/2
- P3/P4
  from DM/ALU out (DM/WB) to DM in

Drawing Dependence Graph

```
1        move   $t0 $zero
2        addi   $t2, $zero,100
3   L:   lw     $t2 0($7)
4        add    $t1 $t2 $s1
5        add    $a $t1 $s5
6        sw     $a  32($s3)
7        add    $6 $3 $a
8        addi   $t0 $t0 1
9        lw     $7 0($8)
10       sw     $7 8($0)
11       add     $s9 $s9 1
12       beq    $t0 $t2          L
13       hlt
```



A Sahu 17

Reference

Patterson, D.A., and Hennessy, J.L. , "Computer Organization and Design: The Hardware/Software Interface

Chapter 6.4/6.5, third edition

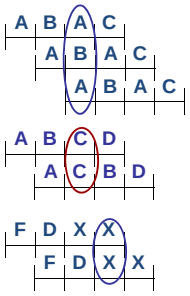
Ebook can be found

A Sahu slide 18

Structural Hazards

Caused by Resource Conflicts

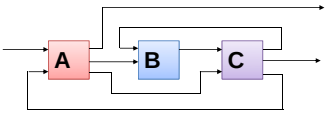
- Use of a hardware resource in more than one cycle
- Different sequences of resource usage by different instructions
- Non-pipelined multi-cycle resources



A Sahu slide 19

Analysis of Structural Hazards

Non-linear Pipeline



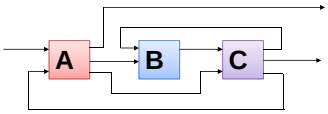
Reservation Table for X (*Required Resources of instruction in Cycle*)

	1	2	3	4	5	6	7	8
A	X					X		X
B		X		X				
C			X		X		X	

A Sahu slide 20

Analysis of Structural Hazards

Multi-functional Pipeline



Reservation Table for X / for Y

	1	2	3	4	5	6	7	8
A	Y				Y	X		X
B		X	Y	X				
C		Y	X	Y	X	Y	X	

A Sahu slide 21

Collisions with Initiation Interval =2

	1	2	3	4	5	6	7	8	9	10	11
A	1		2		3	1	4	1,2	5	2,3	6
B		1		1,2		2,3		3,4		4,5	
C			1		1,2		1-3		2-4		

Collisions

1-3 means 1,2,3

A Sahu slide 22

With Initiation Interval =3

	1	2	3	4	5	6	7	8	9	10	11
A	1			2		1	3	1	2		2
B		1		1	2		2	3		3	
C			1		1	2	1	2	3	2	2

A Sahu slide 23

Collisions with Initiation Interval =5

	1	2	3	4	5	6	7	8	9	10	11
A	1					1,2		1			2,3
B		1		1			2		2		
C			1		1		1	2		2	

Collisions

A Sahu slide 24

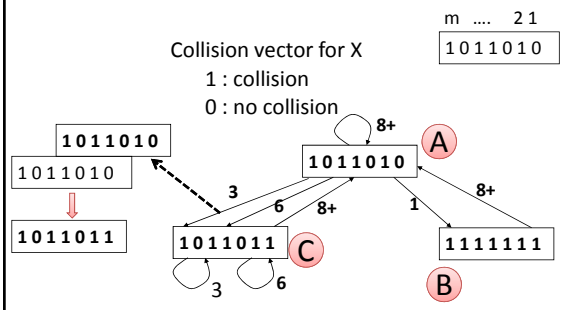
Latency Sequences and Cycles

No Collision for 1, 8, 3 and 6 interval

1, 8, 1, 8, …. (1, 8) avg = 4.5
3, 3, 3, 3, …. (3) avg = 3
6, 6, 6, 6, …. (6) avg = 6

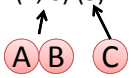
Minimum Average Latency?

Collision Free Scheduling for X



Latency Cycles from State Diagram

Latency Cycles
(1, 8) (1, 8, 6, 8) (3) (6) (3, 8) (3, 6, 3)
Simple Latency Cycles (no figure repeats)
(1, 8) (3) (6) (3, 8) (6, 8)
Greedy Latency Cycles
(1, 8) (3) - from different starting states



Minimum Average Latency (MAL)

MAL ≥ max no. of check marks in any row
MAL ≤ avg latency of any greedy cycle

avg latency of any greedy cycle ≤
no. of 1's in initial collision vector + 1

Upper Bound on MAL

- Consider a greedy cycle $(k_1,k_2,..,k_n)$
- Let p = no. of 1's in initial collision vector

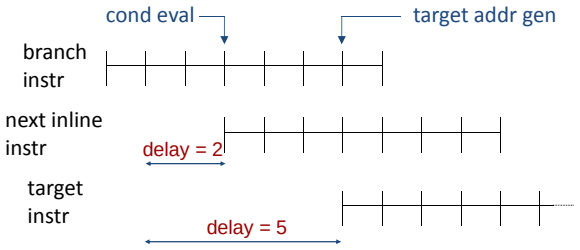
$\Rightarrow k_1 \le p + 1$
$k_2 \le 2\,p - k_1 + 2$
$k_3 \le 3\,p - k_1 - k_2 + 3$
….
$k_n \le n\,p - k_1 - k_2 \ldots - k_{n-1} + n$
$\Rightarrow k_1 + k_2 \ldots + k_n \le n\,p + n \quad \Rightarrow$ **MAL ≤ p + 1**

$k_i \le p+1$,
$k_1+k2 \le 2p+2$

Reference

Kai Hwang, " Advanced Computer Architecture: Parallelism, Scalability, Programmability
Chapter 6

Control Hazards



- the order of *cond eval* and *target addr gen* may be different
- *cond eval* may be done in previous instruction

A Sahu slide 31

Improving Branch Performance

- **Branch Elimination**
  - **replace branch with other instructions**
- **Branch Speed Up**
  - **reduce time for computing CC and TIF**
- **Branch Prediction**
  - **guess the outcome and proceed, undo if necessary**
- **Branch Target Capture**
  - **make use of history**