RISC Architecture: Pipelining

Virendra Singh
Associate Professor
Computer Architecture and Dependable Systems Lab
Department of Electrical Engineering
Indian Institute of Technology Bombay
http://www.ee.iitb.ac.in/~viren/
E-mail: viren@ee.iitb.ac.in
ILP: Instruction Level Parallelism

• Single-cycle and multi-cycle datapaths execute one instruction at a time.
• How can we get better performance?
• Answer: Execute multiple instruction at a time:
  • Pipelining – Enhance a multi-cycle datapath to fetch one instruction every cycle.
  • Parallelism – Fetch multiple instructions every cycle.
Traffic Flow
Automobile Team Assembly

1 car assembled every four hours
6 cars per day
180 cars per month
2,040 cars per year
Automobile Assembly Line

First car assembled in 4 hours (pipeline latency) thereafter 1 car per hour
21 cars on first day, thereafter 24 cars per day
717 cars per month
8,637 cars per year
Throughput: Team Assembly

Time of assembling one car = \( n \) hours

where \( n \) is the number of nearly equal subtasks, each requiring 1 unit of time

Throughput = \( \frac{1}{n} \) cars per unit time
Throughput: Assembly Line

Time to complete first car \( = n \) time units (latency)

Cars completed in time \( T \) \( = T - n + 1 \)

Throughput \( = 1 - \frac{(n - 1)}{T} \) car per unit time

Throughput (assembly line) \( = \frac{1 - (n - 1)}{T} = \frac{n(n - 1)}{T} \rightarrow \frac{n}{T} \) as \( T \rightarrow \infty \)
Some Features of Assembly Line

Electrical parts delivered (JIT)

Task 1
1 hour
Mechanical

Task 2
1 hour
Electrical

Task 3
1 hour
Painting

Task 4
1 hour
Testing

Stall assembly line to fix the cause of defect

3 cars in the assembly line are suspects, to be removed (flush pipeline)

Defect found
Pipelining in a Computer

- Divide datapath into nearly equal tasks, to be performed serially and requiring non-overlapping resources.
- Insert registers at task boundaries in the datapath; registers pass the output data from one task as input data to the next task.
- Synchronize tasks with a clock having a cycle time that just exceeds the time required by the longest task.
- Break each instruction down into a fixed number of tasks so that instructions can be executed in a staggered fashion.
## Single-Cycle Datapath

<table>
<thead>
<tr>
<th>Instruction class</th>
<th>Instr. fetch (IF)</th>
<th>Instr. Decode (also reg. file read) (ID)</th>
<th>Execution (ALU Operation) (EX)</th>
<th>Data access (MEM)</th>
<th>Write Back (Reg. file write) (WB)</th>
<th>Total time</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw</td>
<td>2ns</td>
<td>1ns</td>
<td>2ns</td>
<td>2ns</td>
<td>1ns</td>
<td>8ns</td>
</tr>
<tr>
<td>sw</td>
<td>2ns</td>
<td>1ns</td>
<td>2ns</td>
<td>2ns</td>
<td></td>
<td>8ns</td>
</tr>
<tr>
<td>R-format add, sub, and, or, slt</td>
<td>2ns</td>
<td>1ns</td>
<td>2ns</td>
<td></td>
<td>1ns</td>
<td>8ns</td>
</tr>
<tr>
<td>B-format, beq</td>
<td>2ns</td>
<td>1ns</td>
<td>2ns</td>
<td></td>
<td></td>
<td>8ns</td>
</tr>
</tbody>
</table>

No operation on data; idle time equalizes instruction length to a fixed clock period.
Thank You