5 Pipelined Processor

• temporal overlapping of processing, assembly line
• 5.1 Basic concept
• 5.2 Design space of pipelines
• 5.3 Overview of pipelined instruction processing
• 5.4 Pipelined execution of integer and Boolean instructions
• 5.5 Pipelined processing of loads and stores

5.1.1 Principle of pipelining

Principle of pipelining e.g.

Processing of a sequence of instructions using a basic pipeline

<table>
<thead>
<tr>
<th>Cycle</th>
<th>In</th>
<th>In processing</th>
<th>Out (Finished)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Cycle</td>
<td>Instr 1 →</td>
<td>F₁</td>
<td></td>
</tr>
<tr>
<td>2. Cycle</td>
<td>Instr 2 →</td>
<td>F₂, D₁</td>
<td></td>
</tr>
<tr>
<td>3. Cycle</td>
<td>Instr 3 →</td>
<td>F₃, D₂, E₁</td>
<td></td>
</tr>
<tr>
<td>4. Cycle</td>
<td>Instr 4 →</td>
<td>F₄, D₃, E₂, WB₁ → Instr 1</td>
<td></td>
</tr>
<tr>
<td>5. Cycle</td>
<td>Instr 5 →</td>
<td>F₅, D₄, E₃, WB₂ → Instr 2</td>
<td></td>
</tr>
</tbody>
</table>

Pipelined and unpipelined processing

5.1.2 General structure of pipelines
Structure and pipelined operation of the Fx unit of the IBM Power1

Pipeline Performance Measures
- Cycle time: $t_c$
  - is determined by the worst-case processing time of the longest stage
- Repetition Rate: $R$
  - the shortest possible time interval between subsequent independent instructions in the pipeline
- Performance potential of a pipeline: $P$
  \[ P = \frac{1}{R \times t_c} \]
  - PowerPC603 FP double Mul. e.g. $R = 2$, $t_c = 12$ nsec
  \[ P = \frac{1}{(2 \times 12\text{nsec})} = 44.6\text{ MFLOPS} \]

Performance: RAW-dependent
- Latency:
  - specifies the amount of time that the result of a particular instruction takes to become available in the pipeline for a subsequent dependent instruction.
- Define-use latency (10 to 100 cycles)
  - `mul r1, r2, r3`
  - `add r5, r1, r4`
- Load-use latency (1 to 3 cycles)
  - `load r1, x`
  - `add r5, r1, r2`
- Stalled: the immediately following RAW-dependent instruction has to be stalled in the pipeline for n-1 cycle

Improve Performance
- Multiple-operation instructions
  - HP PA 7100
  \[ \text{FMPYADD}\ RM1,\ RM2,\ RM3,\ RA1,\ RA2 \]
  \[ \text{RM3=RM1*RM2 RA2=RA1+RA2} \]
  - PowerPC
    - FMA for performing $(A*C) + B$

5.1.4 Application scenarios of pipelines

5.2 Design space of pipelines
- key aspect of the design space of pipeline

Layout of a pipeline (in general)

Basic layout of the pipeline

Method of dependency resolution
5.2.2 Basic layout of a pipeline

- Design space of the overall stage layout

Increasing parallelism by raising the number of pipeline stages

Eight-stage pipeline

Problems arise for more stages

- Data and control dependencies occur more frequently
  - stalled and wait for data
  - reload pipe in case of branch
- Subtask becomes less balanced (in execution time)
  - Cycle time is determined by the worst-case processing time of the longest stage
- In most case
  - 5-10 stages

Pipelines e.g. DEC α 21064

Layout of the stage sequence

Used in most cases
Used for performing certain complex operations such as multiplication and division
Bypasses (data forwarding in RAW)
• Unless special arrangements are made,
• the results of the operation instruction is written into
the register file, or into the memory,
• and then it is fetched from there as a source operand.

Principle of bypassing in define-use and load-use conflicts

Possibilities for the timing of pipeline operation

5.3 Overview: pipelined instruction processing

Declaration of Logical Pipeline: e.g. Powerpc 601

Detailed Specification of each of the pipeline: e.g. //
Implementation of instruction pipelines (v.s. logical)

Layout of physical pipelines

Multiplicity of pipelines

Preserveing sequential consistency

Preserveing sequential consistency, implementation e.g.
Case studies: Pentium

- Logic layout of Pentium’s pipelines

![Logic layout of Pentium’s pipelines](image)

Case studies: PowerPC 604

![PowerPC 604 pipelines](image)

5.4 (Specific) Pipelines execution:
Integer and Boolean instructions (FX)

![Diagram of pipelines execution](image)

RISC pipelines 4 or 5 stages

![Diagram of RISC pipelines](image)

Traditional FX pipeline of RISC processors

![Diagram of traditional FX pipeline](image)

Logical to Physical: e.g.
PowerPC 601 using a single universal FX unit

![Diagram of logical to physical](image)

Table 5.2 Variations in pipeline cycle duration in traditional FX pipelines

<table>
<thead>
<tr>
<th></th>
<th>P</th>
<th>D</th>
<th>E</th>
<th>WB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Most processors</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>MC 68010</td>
<td>1</td>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
</tr>
<tr>
<td>SuperSparc</td>
<td>1</td>
<td>3/2</td>
<td>1</td>
<td>1/2</td>
</tr>
</tbody>
</table>
**Layout 5 stages e.g.:**
- FX and L/S pipelines in the MIPS R4200

**CISC pipeline 6 or 5 stages**
- Example pipelines in CISC processors

**Traditional CISC pipeline:**
The execution of register-memory instructions

**CISC pipeline:**
Execution of register-register and load/store instructions

**CISC pipeline 5 stage: recycling E/C stage**

**Implementation of FX units: how many**
Trend in increasing the performance

5.5 (Specific) Pipelines execution: loads and stores

5.5.3 Load-use delay: RICS pipelines

Load-use delay: MIPS

Load-use delay: CISC

Handling Load-use delay

• Basic approaches to cope with a load-use delay
Remove Load-use delay

Remove Load-use delay: bringing forward the calculation of virtual address: for slow cache

Used by early MIPS processors, such as the 36000, 26000.