ECE586 Homework No. 3 Solution

Problem No. 1

(a) The clock period for the pipelined processor is decided by the longest pipeline stage (1.75 ns for the EX stage).
   Pipeline register delay = 0.25 ns
   Therefore:
   Clock period for pipelined processor = 1.75 + 0.25 = 2 ns
   Clock rate = 1 / Clock period = 0.5 GHz

(b) Ideal CPI = 1
   The processor needs to incur a 2-cycle stall after every 6 instructions.
   Therefore:
   Effective CPI = 1 + (1/6)(2) = 1.33

(c) Speedup = \[ \frac{\text{Average time per instruction}_{\text{single-cycle}}}{\text{Average time per instruction}_{\text{pipelined}}} \]
   For the single-cycle processor: Clock period = 6 ns, CPI = 1
   For the pipelined processor: Clock period = 2 ns, CPI = 1.33
   Therefore, Speedup = \[ \frac{6 \times 1}{2 \times 1.33} = 2.25 \]

Problem No. 2

(a)

<table>
<thead>
<tr>
<th>1st Instruction</th>
<th>2nd Instruction</th>
<th>Register</th>
<th>Type of Dependence</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD</td>
<td>ADD</td>
<td>R1</td>
<td>RAW</td>
</tr>
<tr>
<td>LOAD</td>
<td>OR</td>
<td>R1</td>
<td>RAW</td>
</tr>
<tr>
<td>ADD</td>
<td>OR</td>
<td>R6</td>
<td>RAW</td>
</tr>
<tr>
<td>OR</td>
<td>SUB</td>
<td>R3</td>
<td>RAW</td>
</tr>
<tr>
<td>LOAD</td>
<td>OR</td>
<td>R3</td>
<td>WAR</td>
</tr>
<tr>
<td>OR</td>
<td>SUB</td>
<td>R6</td>
<td>WAR</td>
</tr>
<tr>
<td>ADD</td>
<td>SUB</td>
<td>R6</td>
<td>WAW</td>
</tr>
</tbody>
</table>

(b) Pipeline diagram without forwarding:

<table>
<thead>
<tr>
<th>Clock Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instruction</td>
</tr>
<tr>
<td>LOAD</td>
</tr>
<tr>
<td>ADD</td>
</tr>
<tr>
<td>OR</td>
</tr>
<tr>
<td>STORE</td>
</tr>
<tr>
<td>SUB</td>
</tr>
</tbody>
</table>
(c) Pipeline diagram with forwarding:

<table>
<thead>
<tr>
<th>Instruction</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOAD</td>
<td>IF</td>
<td>ID</td>
<td>EX</td>
<td>MEM</td>
<td>WB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD</td>
<td>IF</td>
<td>ID</td>
<td>stall</td>
<td>EX</td>
<td>MEM</td>
<td>WB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OR</td>
<td>IF</td>
<td>ID</td>
<td>stall</td>
<td>EX</td>
<td>MEM</td>
<td>WB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STORE</td>
<td>IF</td>
<td>ID</td>
<td>EX</td>
<td>MEM</td>
<td>WB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUB</td>
<td>IF</td>
<td>ID</td>
<td>EX</td>
<td>MEM</td>
<td>WB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(d) Non-pipelined: Each instruction takes 5 cycles. There are 5 instructions in the sequence.
Execution time = 5 * 5 = 25 cycles

Pipeline without forwarding: Execution time = 14 cycles (shown in the part (b) pipeline diagram)
Pipeline with forwarding: Execution time = 10 cycles (shown in the part (c) pipeline diagram)
Therefore:
Speedup without forwarding compared to non-pipelined execution = 25/14 = \(1.786\)
Speedup with forwarding compared to non-pipelined execution = 25/10 = \(2.5\)

**Problem No. 3**

Percentage of branches = 20%
Branch misprediction rate = 15%

If the pipeline never stalls, one instruction is completed every clock cycle. But, in the presence of branches, the pipeline would be stalled each time the branch predictor makes an incorrect prediction.

For the 5-stage pipeline:
Branch misprediction penalty = 2 cycles
CPI = Ideal CPI + (Percentage of branches)(Branch misprediction rate)(Branch misprediction penalty)
CPI = \(1 + (20\%)(15\%)(2) = 1.06\)

For the 10-stage pipeline:
Branch misprediction penalty = 5 cycles
CPI = Ideal CPI + (Percentage of branches)(Branch misprediction rate)(Branch misprediction penalty)
CPI = \(1 + (20\%)(15\%)(5) = 1.15\)

Speedup of 10-stage pipeline over 5-stage pipeline = \(\frac{(Execution\ time)_{5-stage}}{(Execution\ time)_{10-stage}}\)

Speedup = \(\frac{(Clock\ period\ *\ CPI)_{5-stage}}{(Clock\ period\ *\ CPI)_{10-stage}}\ = \frac{1\ ns\ *\ 1.06}{0.6\ ns\ *1.15} = 1.536\)
Problem No. 4

(a) The clock period for a pipelined processor is decided by the longest pipeline stage.

For the 5-stage pipeline:

Longest pipeline stage = 1.8 ns
Pipeline register delay = 0.2 ns
Therefore:
Clock cycle time = 1.8 + 0.2 = 2 ns
Clock rate = 1 / Clock cycle time = 500 MHz

For the 10-stage pipeline:

All the pipe stages in the original 5-stage pipeline are split into half. Therefore:
Longest pipeline stage = 1.8 / 2 = 0.9 ns
Pipeline register delay = 0.2 ns
Clock cycle time = 0.9 + 0.2 = 1.1 ns
Clock rate = 1 / Clock cycle time = 909 MHz

(b) Pipeline diagram for the 10-stage pipeline is shown below. Operand forwarding is indicated by arrows.

There are two RAW dependences in the code sequence: (i) Between LOAD and SUB instructions due to register R1 (forwarding happens from the output of MEM2 stage to the input of EX1 stage), (ii) Between SUB and STORE instructions for register R6 (forwarding happens from the output of EX2 stage to the input of EX1 stage).

| Instruction | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   | 10  | 11  | 12  | 13  | 14  | 15  | 16  |
|-------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| LOAD        | IF1 | IF2 | ID1 | ID2 | EX1 | EX2 | MEM1| MEM2| WB1 | WB2 |     |     |     |     |     |     |
| ADD         | IF1 | IF2 | ID1 | ID2 | EX1 | EX2 | MEM1| MEM2|     |     |     |     | WB1 | WB2 |     |     |
| SUB         | IF1 | IF2 | ID1 | ID2 | stall| stall|     |     |     |     |     | EX1 | EX2 | MEM1| MEM2| WB1 | WB2 |
| STORE       | IF1 | IF2 | ID1 | stall| stall| ID2 | stall|     | EX1 | EX2 | MEM1| MEM2|     |     |     |     |

As shown in the pipeline diagram, the code sequence takes 16 cycles to complete on the 10-stage pipeline. The clock period for the 10-stage pipeline is 1.1 ns.

Therefore:

Execution time in nanoseconds = Clock period in nanoseconds * Number of cycles
= 1.1 * 16 = 17.6 ns
Problem No. 5

Clock rate = 2.6 GHz

Instruction throughput = Instructions completed per second = Instructions per cycle * Cycles per second
Instruction throughput = (1/CPI) * Clock rate = Clock rate/CPI

To calculate the CPI, we need to quantify the stall cycles. Since there are no control hazards (100% branch prediction accuracy) and the pipeline employs full forwarding hardware, the only source of stall cycles are load instructions which are immediately followed by dependent instructions. For each such load instruction, the pipeline needs to be stalled for 1 cycle. Therefore:

\[ \text{CPI} = 1 + (25\%)(40\%)(1) = 1.1 \]

Instruction throughput = Clock rate/CPI = \((2.6 \times 10^9)/(1.1) = 2.364 \text{ billion instructions } / \text{ second}\)

Problem No. 6

(a) (ii) 1.5x to 2.0x
(b) (iv) Nearly smooth and linear with some dips
(c) The performance simulator enabled the designers to model the initial straw-man architecture and perform a sensitivity analysis of various micro-architectural optimizations, without the overhead of having to build real hardware.
(d) (i) Added a clock cycle to data cache lookup, changing it from 2 to 3 cycles,
(ii) Reworking the instruction fetch/decode/rename pipeline, resulting in an additional two stages in the fetch pipeline
These changes resulted in an increase in average CPI. However, they also enabled clock frequency to be increased by 50%. Since the increase in frequency was significantly higher than the increase in average CPI, these trade-offs resulted in a net gain in performance.