CSEE W3827
Fundamentals of Computer Systems
Homework Assignment 5
Solutions

Prof. Martha A. Kim
Columbia University
Due December 5, 2013 at 10:10 AM

Write your name and UNI on your solutions
Show your work for each problem; we are more interested in how you get the answer than whether you get the right answer.
1. (20 pts.) Imagine that the MIPS ISA were modified to disallow non-zero offsets in address calculations. All loads or stores of the form

\[ \text{lw } \$x, \ a(\$y) \]

would need to become a two-instruction sequence such as:

\[
\begin{align*}
\text{addi } \$y, \ \$y, \ a \\
\text{lw } \$x, \ \$y
\end{align*}
\]

(a) Show the modifications to the datapath and control to support this change. You should start from the implementation shown below and be sure not to change the behavior of any instructions other than the \textit{lw} and \textit{sw}.

(b) Assuming that SPECINT2000 consists of 25% loads and 10% stores, of which 40% and 50% respectively have non-zero offsets, what is percent change in SPECINT2000 execution time? You may assume no change in clock period as a result of your changes.
Part b: Because this modification does not affect the CPI or clock period of this processor, any change in runtime will be due solely to a change in the dynamic instruction count.

Let $I$ be the number of dynamic instructions ($\text{lw}$, $\text{sw}$, and other) in the original version of SPEC:

$$I = (0.25 \times I) + (0.1 \times I) + (0.65 \cdot I)$$

The number of instructions after this change, $I_{after}$ would be:

$$I_{after} = (1.4 \times 0.25 \times I) + (1.5 \times 0.1 \times I) + (0.65 \cdot I)$$
$$= (0.35 \times I) + (0.15 \times I) + (0.65 \cdot I)$$
$$= 1.15 \times I$$

Therefore, these changes will cause SPECT to take 15% more time.
2. (15 points) Assuming the following control signals are (individually) stuck at 0, meaning they always have the value 0. For each, give an example of an instruction that would malfunction as a result.

(a) *MemToReg*

Data from memory will never be written to the register file, therefore *lw* will fail.

(b) *ALUOp*₁₀

*ALUOp = 00* forces the ALU to add, so any instruction that needs the ALU to do something else will fail, e.g., *and*, *sub*, or *beq*.

(c) *PCSrc*

*PCSrc = 0* forces *PC’ = PC + 4* no matter what, so taken branches will fail, e.g., *beq $0, $0, L*. 
3. (20 points) The table below describes the performance of two processors, A and B, and two compilers, C and D, on a standard benchmark program. Which is the best compiler-machine combination? Provide quantitative support for your choice.

<table>
<thead>
<tr>
<th>Processor</th>
<th>GHz</th>
<th>Compiler C</th>
<th>Avg. CPI</th>
<th>Compiler D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor A</td>
<td>3.4</td>
<td>7000</td>
<td>1.2</td>
<td>5000</td>
</tr>
<tr>
<td>Processor B</td>
<td>2.6</td>
<td>1500</td>
<td>2.2</td>
<td>1000</td>
</tr>
</tbody>
</table>

We can compute the execution time of each combination from the information provided above:

\[
T_{CA} = 7000 \times \frac{1.2}{3.4} = 2,470ns
\]
\[
T_{DA} = 5000 \times \frac{1.5}{3.4} = 2,205ns
\]
\[
T_{CB} = 1500 \times \frac{2.2}{2.6} = 1,270ns
\]
\[
T_{DB} = 1000 \times \frac{4.0}{2.6} = 1,540ns
\]

Compiler C on processor B is the best combination as it results in the fastest execution of the benchmark.
4. (20 points) Assuming the single-cycle processor examined in class, and the two sets of latencies shown below (processors A and B), indicate the critical path for each instruction listed below. All latencies provided are in units of picoseconds.

<table>
<thead>
<tr>
<th></th>
<th>I-Mem</th>
<th>Adder</th>
<th>Mux</th>
<th>ALU</th>
<th>Regfile</th>
<th>D-Mem</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Processor A</strong></td>
<td>400</td>
<td>100</td>
<td>30</td>
<td>120</td>
<td>200</td>
<td>350</td>
<td>100</td>
</tr>
<tr>
<td><strong>Processor B</strong></td>
<td>500</td>
<td>150</td>
<td>100</td>
<td>180</td>
<td>220</td>
<td>1000</td>
<td>65</td>
</tr>
</tbody>
</table>

(a) An and instruction on processor A.

\[
CP_{\text{and},A} = IMem + Regfile + Mux_{\text{SrcB}} + ALU + Mux_{WD3} \\
= 400 + 200 + 30 + 120 + 30 \\
= 780\text{ps}
\]

Other paths (e.g., PC increment, control, are all shorter as they do not include I-Mem)

(b) A beq instruction on processor A.

\[
CP_{\text{beq},A} = IMem + Regfile + Mux_{\text{SrcB}} + ALU + Mux_{PC} \\
= 400 + 200 + 30 + 120 + 30 \\
= 780\text{ps}
\]
(c) A sw instruction on processor B.

\[ CP_{sw,B} = IMem + \text{Max}(\text{Regfile}, \text{Mux}_{SrcB}) + \text{ALU} + \text{DMem} \]
\[ = 500 + \text{Max}(220, 100) + 180 + 1000 \]
\[ = 1900\text{ps} \]

(d) A lw instruction on processor B.

\[ CP_{lw,B} = IMem + \text{Max}(\text{Regfile}, \text{Mux}_{SrcB}) + \text{ALU} + \text{DMem} + \text{Mux}_{WD3} \]
\[ = 500 + \text{Max}(220, 100) + 180 + 1000 + 100 \]
\[ = 2000\text{ps} \]
5. (25 points) Consider the following MIPS program:

```
addi $s0, $0, 0  # i = 0
add $s1, $0, $0   # sum = 0
addi $t0, $0, 10  # $t0 - 10
loop:
    slt $t1, $s0, $t0  # if (i < 10), $t1 == 1, else $t1 = 0
    beq $t1, $0, done  # if (i >= 10), branch to done
    add $s1, $s1, $s0   # sum += i
    addi $s0, $s0, 1    # i++
    j loop
done:
```

(a) How many cycles does it take to execute the following program on the multicycle MIPS processor?

First let us analyze the dynamic instruction count. We can do so by breaking it into three parts:

- **Initialization**: addi, add, addi. A 3 instruction sequence that on the multicycle processor takes $4 + 4 + 4 = 12$ cycles.

- **Main loop iteration**: slt, beq, add, addi, j. A 5 instruction sequence that takes $4 + 3 + 4 + 4 + 3 = 18$ cycles. As it will be executed 10 times (for $i = 0, 1, \ldots, 9$) this totals to 50 instructions or 180 cycles.

- **Exit loop iteration**: slt, beq. When $i = 10$ these two instructions from the loop will be executed, the branch
taken, and the program terminated. These last 2 instructions take $4 + 3 = 7$ cycles.

So, our total cycles for this program is $12 + (10 \times 180) + 7 = 199$ cycles.

(b) What is the CPI of this program?

CPI is cycles per instruction. We already have cycles from above, but need to compute the dynamic instruction total $3 + (10 \times 5) + 2 = 55$. So CPI = $199/55 = 3.62$. 