CS M151B: Homework 4

Jonathan Woong

804205763

Spring 2017

Discussion 1A

Monday 1st May, 2017

Problem 1

Compilers can have a profound impact on the performance of an application. Assume that for a program, compiler A results in a dynamic instruction count of 1.0E9 and has an execution time of 1.1 s, while compiler B results in a dynamic instruction count of 1.2E9 and an execution time of 1.5 s.

a. Find the average CPI for each program given that the processor has a clock cycle of 1 ns.

$$1 \text{ ns } = 1.0 \times 10^{-9} \text{ s}$$

$$\text{execution time} = \text{instruction count} \times \text{CPI} \times \text{T}_{\text{cycle}}$$

$$\text{CPI} = \frac{\text{execution time}}{\text{instruction count} \times \text{T}_{cycle}}$$

$$\text{CPI}_A = \frac{1.1}{(1 \times 10^9)(1 \times 10^{-9})} = \boxed{1.1}$$

$$\text{CPI}_B = \frac{1.5}{(1.2 \times 10^9)(1 \times 10^{-9})} = \boxed{1.25}$$

b. Assume the compiled program runs on two different processors. If the execution times on the two processors are the same, how much faster is the clock of the processor running compiler A's code versus the clock of the processor running on compiler B's code?

$$n = \frac{\text{instruction count}_B}{\text{instruction count}_A} = \frac{1.2 \times 10^9}{1.0 \times 10^9} = 1.2 \rightarrow \boxed{20\% \text{ faster}}$$

c. A new compiler is developed that uses only 6.0E8 instructions and has an average CPI of 1.1. What is the speedup of using this new compiler versus using compiler A or B on the original processor?

$$\text{execution time}_C = (6.0 \times 10^8)(1.1)(1 \times 10^{-9}) = 0.66$$

$$\text{speedup}_A = \frac{\text{execution time}_A}{\text{execution time}_C} = \frac{1.1}{0.66} = 1.67 \rightarrow \boxed{67\% \text{ speedup}}$$

$$\text{speedup}_B = \frac{\text{execution time}_B}{\text{execution time}_C} = \frac{1.5}{0.66} = 2.27 \rightarrow \boxed{127\% \text{ speedup}}$$

Problem 2

Consider the performance of a processor with a clock frequency of 2.4GHz on a suite of benchmarks. Two different sets of measurements are made: A) the programs are compiled by a compiler, and B) the programs are written in assembly by a highly-skilled programmer. The highly-skilled programmer manages to get a reduction of a factor of 2.3 in the CPI and a performance improvement of a factor of 9.7 compared to the compiled programs. What is the reduction in instruction count obtained by the highly-skilled programmer?

$$\begin{aligned}
9.7 &= \frac{\text{performance}_B}{\text{performance}_A} \\
&= \frac{\text{execution time}_A}{\text{execution time}_B} \\
&= \frac{\text{instruction count}_A \times \text{CPI}_A \times \text{T}_{\text{cycle}}}{\text{instruction count}_B \times \text{CPI}_B \times \text{T}_{\text{cycle}}} \\
&= \frac{\text{instruction count}_A}{\text{instruction count}_B \times 2.3} \\
\boxed{22.31} &= \frac{\text{instruction count}_A}{\text{instruction count}_B}
\end{aligned}$$

Problem 3

When a processor executes a program P_A, the execution time is 3.8 seconds and the CPI is 2.9. When the same processor executes program P_B, the execution time is 8.5 seconds and the CPI is 3.7. A typical workload consists of executing P_A, then executing P_B. What is the CPI for this typical workload?

$$\text{execution time} = \text{instruction count} \times \text{CPI} \times \text{T}_{\text{cycle}}$$

$$\text{instruction count} = \frac{\text{execution time}}{\text{CPI} \times \text{T}_{\text{cycle}}}$$

$$\text{instruction count}_A = \frac{3.8}{2.9 \times \text{T}_{\text{cycle}}} = \frac{1.31}{\text{T}_{\text{cycle}}}$$

$$\text{instruction count}_B = \frac{8.5}{3.7 \times \text{T}_{\text{cycle}}} = \frac{2.30}{\text{T}_{\text{cycle}}}$$

$$\text{instruction count}_A + \text{instruction count}_B = \frac{3.61}{\text{T}_{\text{cycle}}}$$

$$\text{execution time}_A + \text{execution time}_B = 3.8 + 8.5 = 12.3 \text{ s}$$

$$\text{CPI}_{A+B} = \frac{\text{execution time}_{A+B}}{\text{instruction count}_{A+B} \times \text{T}_{cycle}} = \frac{12.3}{\frac{3.61}{\text{T}_{\text{cycle}}} \times \text{T}_{cycle}} = \frac{12.3}{3.61} = \boxed{3.41}$$

Problem 4

Consider a processor that achieves an overall average CPI of 3.6 when executing a program P1. 23% of the instructions of P1 perform floating point operations. For these floating point instructions, the average CPI is 5. By improving the design of the floating point ALU, the average CPI for floating point instructions is reduced from 5 to 3.

A) What is the overall average CPI for the improved processor?

Let F = float, NF = non-float, I = improved, NI = non-improved

$$\begin{aligned}
\text{CPI}_{NI} &= (0.77)\text{CPI}_{NF} + (0.23)\text{CPI}_F \\
3.6 &= (0.77)\text{CPI}_{NF} + (0.23)(5) = (0.77)\text{CPI}_{NF} + 1.15 \\
2.45 &= (0.77)\text{CPI}_{NF} \\
\text{CPI}_{NF} &= 3.18 \\
\text{CPI}_I &= (0.77)\text{CPI}_{NF} + (0.23)\text{CPI}_F \\
&= 2.45 + (0.23)(3) \\
\text{CPI}_I &= \boxed{3.14}
\end{aligned}$$

B) If the original processor executed P1 in 190 seconds, how long will it take the improved processor to execute P1?

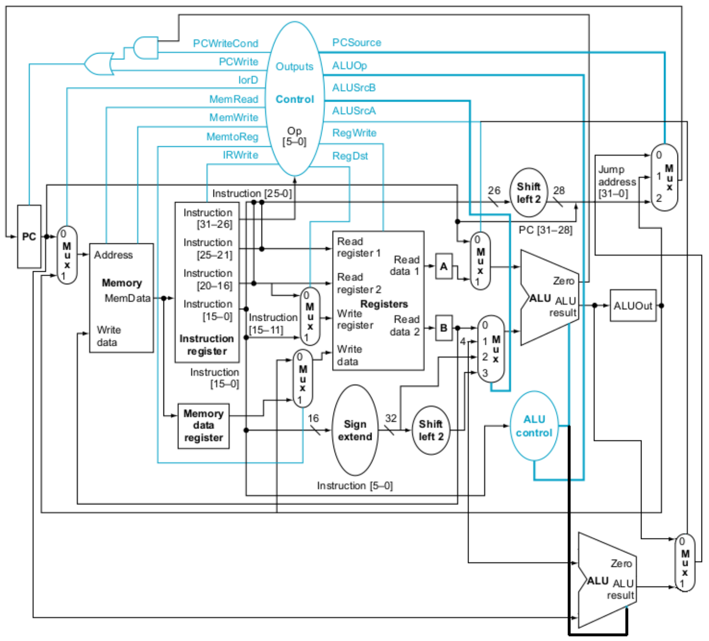
$$\begin{aligned}
\text{execution time}_{NI} &= \text{instruction count} \times \text{CPI}_{NI} \times \text{T}_{\text{cycle}} \\
190 &= \text{instruction count} \times 3.6 \times \text{T}_{\text{cycle}} \\
52.78 &= \text{instruction count} \times \text{T}_{\text{cycle}} \\
\text{execution time}_I &= \text{instruction count} \times \text{CPI}_I \times \text{T}_{\text{cycle}} \\
&= (3.14)(52.78) \\
&= \boxed{165.72}
\end{aligned}$$

Problem 5

Consider the multicycle MIPS implementation from chapter 5, 3rd Ed, (Figure 5.28). Assume that the processor executes a long sequence of consecutive `lw` instructions or a long sequence of consecutive `sw` instructions. For each of these cases we would like to use pipelining to reduce the CPI by 1. The approach is to fetch the next instruction (instruction fetch step in Figure 5.30) during the execution of the previous instruction.

A) Explain the basic idea of your modifications in 2-4 clear sentences. I add another ALU (with inputs PC and 4) that will increment the PC by 4 when ALUSrcA is 1, ALUSrcB is 10, and ALUOp is 00 (memory address computation stage). If the previous `lw` or `sw` is in the memory address computation stage, the PC will increment by 4 and become an input to the MUX controlled by ALUSrcA. Since ALUSrcA must be 1 for the memory address computation stage, the incremented PC will be passed into the MUX controlled by PCSource signal. When the previous instruction is computing memory address, we simply need to set MemRead = 1, IorD = 0, IRWrite = 1, PCWrite = 1, and PCSource = 00 to fetch the next instruction. When ALUSrcA is 0, the current implementation behaves as if no changes were made.

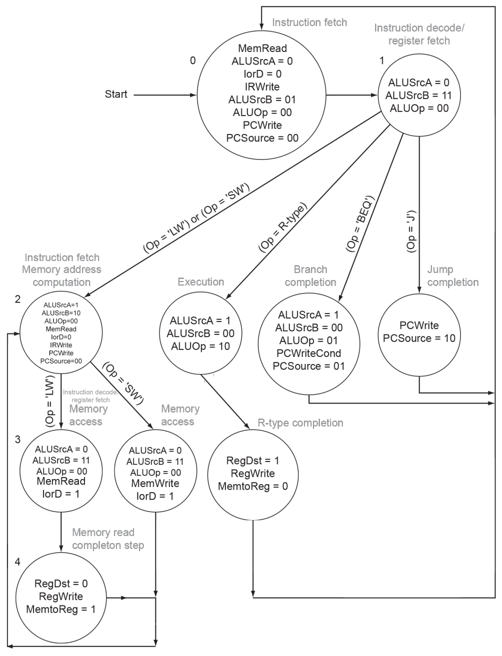
B) Show the necessary modifications to the datapath. If modifications are required, show the entire modified datapath (Figure 5.28 modified as necessary). If you need to modify any of the datapath building blocks, draw the modified building blocks and explain the modifications.



C) Are any new control signals required? If so, list them with an explanation and identify them on the datapath diagram.

No new control signals are required

D) Modify the control unit state diagram (Figure 5.37) to reflect the changes.



To visualize the execution flow, see tables below:

Instruction	CC 1	CC 2	CC 3	CC 4	CC 5	CC 6	CC 7	CC 8
`lw`	IF	ID	EXE	MA	WB			
`lw`			IF	ID		EXE	MA	WB

Two instructions completed in 8 cycles = 4 instructions per cycle.

Instruction	CC 1	CC 2	CC 3	CC 4	CC 5	CC 6
`sw`	IF	ID	EXE	MA		
`sw`			IF	ID	EXE	MA

Two instructions completed in 6 cycles = 3 instructions per cycle.

Problem 6

What is the total latency of an LW instruction in a pipelined and non-pipelined processor?

800 ps for both. Pipelining does not affect latency, it only affects throughput.

If asking about average latency for pipelined processor with many LW instructions, 200 ps.

Problem 7

If we can split one stage of the pipelined datapath into two new stages, each with half the latency of the original stage, which stage would you split and what is the new clock cycle time of the processor?

Split the memory access stage into two, the new clock cycle time will be 400 ps.

Problem 8

Refer to the following sequence of instructions:

i1 : **or** r1 , r2 , r3

i2 : **or** r2 , r1 , r4

i3 : **or** r1 , r1 , r2

For each depdendence indicate:

- Which instruction is dependent on (must be executed after) which instruction. For this purpose, label the instructions within each of the program segments: i1, i2, i3.
- What is the type of depdendence: RAW, WAR, WAW
- What is the storage location (for example, the register number) through which there is the dependence

i2 depends on i1 (RAW) r1

i3 depends on i1 (RAW, WAW) r1

i3 depends on i2 (RAW) r2

Problem 9

Assume there is no forwarding in this pipelined processor. Indicate hazards and add `nop` instructions to eliminate them.

Instruction	CC 1	CC 2	CC 3	CC 4	CC 5	CC 6	CC 7
`i1: or r1,r2,r3`	IF	ID	EXE	MA	WB		
`i2: or r2,r1,r4`		IF	ID	EXE	MA	WB	
`i3: or r1,r1,r2`			IF	ID	EXE	MA	WB

The cells colored above are where hazards occur.

For i2: the r1 operand that is required has not yet been updated by i1.

For i3: the r2 operand that is required has not yet been updated by i2.

Instruction	CC 1	CC 2	CC 3	CC 4	CC 5	CC 6	CC 7	CC 8	CC 9	CC 10	CC 11	CC 12	CC 13
`i1: or r1,r2,r3`	IF	ID	EXE	MA	WB								
`i2: or r2,r1,r4`		IF	ID	`nop`	`nop`	`nop`							
			IF	ID	`nop`	`nop`	`nop`						
				IF	ID	EXE	MA	WB					
`i3: or r1,r1,r2`					IF	ID	`nop`	`nop`	`nop`				
						IF	ID	`nop`	`nop`	`nop`			
							IF	ID	EXE	MA	WB		

The WB for the instructions above occur during the first half of the cycle, and the ID for the instructions occur during the second half of the cycle.

Problem 10

Assume there is full forwarding. Indicate hazards and add `NOP` instructions to eliminate them.

Instruction	CC 1	CC 2	CC 3	CC 4	CC 5	CC 6	CC 7	CC 8	CC 9
`i1: or r1,r2,r3`	IF	ID	EXE	MA	WB				
`i2: or r2,r1,r4`		IF	ID	`NOP`	`NOP`	`NOP`			
			IF	ID	EXE	MA	WB		
`i3: or r1,r1,r2`				IF	ID	`NOP`	`NOP`	`NOP`	
					IF	ID	EXE	MA	WB

Problem 11

Consider the pipelined MIPS implementation shown in Figure 4.51 (pade 304). For each of the following two instructions separately, for each of the pipeline stages, specify the values of the control signals that are asserted for the instruction, when the instruction reaches that stage.

a. `sw   $14,44($3)`

Stage	RegDst	RegWrite	ALUOp	ALUSrc	PCSrc	MemRead	MemWrite	MemtoReg
IF	0	0	00	0	0	0	0	0
ID	0	0	00	0	0	0	0	0
EXE	0	0	00	1	0	0	0	0
MA	0	0	00	0	0	0	1	0
WB	0	0	00	0	0	0	0	0

b. `sub   $21,$9,$2`

Stage	RegDst	RegWrite	ALUOp	ALUSrc	PCSrc	MemRead	MemWrite	MemtoReg
IF	0	0	00	0	0	0	0	0
ID	0	0	00	0	0	0	0	0
EXE	0	0	10	0	0	0	0	0
MA	0	0	00	0	0	0	0	0
WB	0	1	00	0	0	0	0	0