Comp Org Homework 4

Aaron Verkleeren

December 2023

1 12.14.5:

In the exercise, we examine in detail how an instruction is executed in single-cycle datapath. Problems in this exercise refer to a clock cycle in which the processor fetches the following instruction word: 0xadac0014

1.1 What are the values of the ALU control unit's inputs for this instruction?

The ALU control unit's inputs depend on the opcode and function bits of the instruction. For the sw instruction, the opcode is 0b101011 (or 43 in decimal), which represents a store word operation.

In a MIPS processor, the ALU control inputs for this operation are:

$\boxed{\text{ALUOp: 0b00 (for store operations) ALUControl: 0b0010 (for addition)}}$

1.2 What is the new PC address after this instruction is executed? Highlight the path through which this value is determined.

The new PC address is determined by the PC + 4 for normal sequential execution or by a branch/jump condition. In a single-cycle datapath, the PC is updated directly for sequential execution. The PC + 4 is calculated and loaded into the PC register. Therefore, the new PC after executing this instruction will be the current $\boxed{\text{PC value + 4}}$

$\boxed{\text{Path: PC} \rightarrow \text{Adder (PC + 4)} \rightarrow \text{Banch Decider Mux} \rightarrow \text{PC}}$

1.3 For each mux, show the values of its inputs and outputs during the execution of this instruction. List values that are register outputs at Reg [xn]

For the first mux, the value at 0 is $t4 and the value at 1 is 0000, since the RegDst value is 1 (from op being sw), the output is 0000

For the second mux, the input at 0 is the value at $t4, and at 1 is the sign extended 20, or 0xFFFF000A, and since the ALUSrc is 1 for sw, the output is 0xFFFF000A

For the third mux, the input at 0 is the PC address + 4, and at 1 it is the sign extended immediate times 4, plus the PC + 4 value, and since PCSrc is 1, the output is PC + 4

************* idk what the last part means make sure you check ***

1.4 What are the input values for the ALU and the two add units?

The inputs for the ALU are the data from the read register 1 ($t5), and the sign extended immediate (0xFFFF000A)

The inputs for the first adder are the PC address, and 4

The inputs for the second adder are the PC address + 4, and the sign extended immediate multiplied by 4

1.5 What are the values of all inputs for the registers unit?

**** Check this one too ******

The values of all the inputs for the registers are as follows:

Read register 1: $t5

Read register 2: $t4

Write register: 0000

2 12.14.7

Problems in this exercise assume that the logic blocks used to implement a processor's datapath (COD Figure 4.21) have the following latencies:

I-mem/D-mem: 250ps, Register file: 150ps, Mux: 25ps, ALU: 200ps, Adder: 150ps, Single gate: 5ps, Register read: 30ps, Register setup: 20ps, Sign extend: 50ps, control: 50ps

"Register read" is the time needed after the rising clock edge for the new register value to appear on the output. This value applies to the PC only. "Register setup" is the amount of time a register's data input must be stable before the rising edge of the clock. This value applies to both the PC and Register File

2.1 What is the latency of an R-type instruction(i.e., how long must the clock period be to ensure that this instruction works correctly)?

Instruction fetch + register file + 2 mux + ALU + register read + register setup

$= 250 + 150 + 2(25) + 200 + 30 + 20 = \boxed{700\text{ps}}$

2.2 What is the latency of lw? (Check your answer carefully. Many students place extra muxes on the critical path.)

Instruction fetch + register file + 2 mux + D-mem + ALU + + register read + register setup

$= 250 + 150 + 2(25) + 250 + 200 + 30 + 20 = \boxed{950\text{ps}}$

2.3 What is the latency of sw? (Check your answer carefully. Many students place extra muxes on the critical path.)

Instruction fetch + register file + register read + mux + ALU + D-mem

$= 250 + 150 + 30 + 25 + 200 + 250 = \boxed{905\text{ps}}$

2.4 What is the latency of beq?

Instruction fetch + register file + register read + 2 mux + ALU + logic gate + register setup

$= 250 + 150 + 30 + 2(25) + 200 + 5 + 20 = \boxed{705\text{ps}}$

2.5 What is the latency of an arithmetic, logical, or shift I-type (non-load) instruction?

Instruction fetch + register file + register read + 2 mux + ALU + register setup

$= 250 + 150 + 30 + 2(25) + 200 + 20 = \boxed{700}$

2.6 What is the minimum clock period for this CPU?

To find the minimum clock period we must look at the instruction with the largest latency. Here that instruction is lw with a time of 950ps.

Minimum clock period = $\boxed{950}$

3 12.14.10

3.1 What is the speedup achieved by adding this improvement?

Clock Time prior to improvements = 5 + 30 + 20 + 50 + 50 + 250 + 150 + 25 + 200 + 150 = 930ps

Clock Time after improvements = 30 + 20 + 50 + 50 + 250 + 160 + 25 + 200 + 150 + 5 = 940ps

# of LDUR and STUR instructions after doubling $25 - (25 * 0.12) = 22$

52 + 22 + 11 + 12 = 97 total instructions after improvement

Speed up = $\frac{930}{100} / \frac{930}{97} \approx \boxed{0.96}$

3.2 Compare the change in performance to the change in cost.

Calculate Cost before improvements =

$$1000 + 200 + 10 + 100 + 30 + 2000 + 5 + 100 + 1 + 500 = 3946$$

Calculate Cost per unit clock cycle before improvement =

$$\text{Cost / Clock cycle time} = \frac{3946}{930} = 4.24$$

Cost after improvements =

$$1000 + 400 + 10 + 100 + 30 + 2000 + 5 + 100 + 1 + 500 = 4146$$

Cost per unit clock cycle after improvement =

$$\text{Cost / Clock cycle time} = \frac{4146}{940} = 4.38$$

Change in costs =

$$4.38 - 4.24 = 0.14$$

Performance before improvements =

$$\frac{1}{100*930*10^{-12}} = 10.75*10^6$$

Performance after improvements =

$$\frac{1}{97*940*10^{-12}} = 10.967*10^6$$

Change in performance =

$$10.967*10^6 - 10.75*10^6 = 0.217*10^6$$

Cost Ratio $= \frac{4.38}{4.24} = 1.03$

Performance Ratio $= \frac{10.967}{10.75} = 1.02$

We see that the performance ratio is $\boxed{1.02}$ while the cost ratio is $\boxed{1.03}$

The Cost Ratio is higher than the Performance Ratio by $\boxed{0.01}$

3.3 Given the cost/performance ratios you just calculated, describe a situation where it makes sense to add more registers and describe a situation where it doesn't make sense to add more registers.

When focusing on performance enhancement, expanding the number of registers is a a good choice. This is because additional registers can help streamline the execution of instructions, leading to improved system performance, although at a higher cost.

On the other hand, if minimizing expenses is the primary objective, increasing the number of registers might not be the best approach. Fewer registers help in cost-cutting but can lead to a decrease in performance. This is due to an increase in the number of required instructions and a negative impact on latency.

In essence, the decision to add more registers aligns well with performance-focused strategies, despite the accompanying rise in costs. Conversely, for cost-sensitive scenarios, adding more registers could lead to a compromise in performance, making it a less favorable option.

4 12.14.16

4.1 What is the clock cycle time in a pipelined and nonpipelined processor?

For a pipelined processor

Clock cycle time of pipelined processor = $\boxed{350\text{ps}}$ (Equal to slowest instruction which is Instruction Decode)

For a non-pipelined processor

IF + ID + EX + MEM + WB = 250ps + 350ps + 150ps + 300ps + 200ps = $\boxed{1250\text{ps}}$

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

For a pipelined processor

Total latency = number of cycles * clock cycle time = $5 * 350ps$ = $\boxed{\text{1750ps}}$

For a non-pipelined processor

Total latency = IF + ID + EX + MEM + WB = 250ps + 350ps + 150ps + 300ps + 200ps = $\boxed{\text{1250ps}}$

4.3 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?

Splitting the stage that takes the longest time(ID in this case) would be the best way of reducing clock cycle time. Doing this would mean that our new cycle clock time would be whichever stage takes the longest now after splitting ID, which is now MEM.

So the new clock cycle time of the pipelined processor will be $\boxed{\text{300ps}}$

4.4 Assuming there are no stalls or hazards, what is the utilization of the data memory?

If we assume that there are no stalls or hazards in memory, then only two instructions will be used in memory, the load and store instructions.

Total utilization = Load instruction % used + store instruction % used

Total utilization = 20% + 15% = $\boxed{35\%}$

4.5 Assuming there are no stalls or hazards, what is the utilization of the write-register port of the "Registers" unit?

Utilization of the write-register port = LW utilization % used + ALU utilization % used = 20% + 45% = $\boxed{\text{65\% Clock Cycles}}$