COIS 2300H Assignment 3

This assignment is worth 10% of your final grade. ‘

Trent Username: punyajamishra

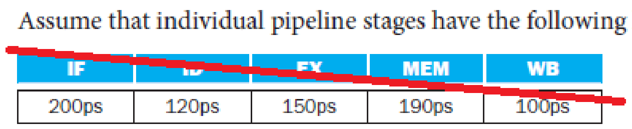
Your name as it appears on blackboard: Punyaja Mishra

Theory Part:

Question 1 Resource and Control Hazards in Pipeline execution (12 marks total):

(Based on section 4.5 in the textbook):

Assume that individual pipeline stages have the following latencies (use the ones supplied by Sri/Alaadin):



Assume that individual pipeline stages have the following latencies:

~~IF~~	~~ID~~	~~EX~~	~~MEM~~	~~WB~~
~~200ps~~	~~120ps~~	~~150ps~~	~~190ps~~	~~100ps~~

The textbook version:

The **Sri/Alaadin** version (same question, different numbers) (Based on figure 4.28):

Instruction Fetch (IF)	Instruction Decode (ID)	Execute (EX)	Memory (MEM)	Write Back (WB)
250ps	100ps	175ps	150ps	200ps

A) (4.10.1) Fill in the chart below with the name of each pipeline stage to the right of the instruction provided in the first column. In the bottom row, write how long each pipeline stage will take in cycles.

- Assume that all branches are perfectly predicted (this eliminates all control hazards) and that no delay slots are used.
- If we only have one memory (for both instructions and data), there is a structural hazard every time we need to fetch an instruction in the same cycle in which another instruction accesses data. To guarantee forward progress, this **hazard must always be resolved in favor of the instruction that accesses data.**

If you need to use a spreedsheet to answer this, and paste that in (or attach it) that's ok too.

For a structural hazard :

Time ⇨

Pipeline slots	0	1	2	3	4	5	6	7	8	9	10	11	12
`sw 16,12(r6)`	IF	ID	EX	MEM	WB								
`lw r16,8(r6)`		IF	ID	EX	MEM	WB							
`beq r5,r4,Label` **`# Assume r5!=r4`**			IF	ID	EX	MEM	WB						
`add r5,r1,r4`				-----	-----	IF	ID	EX	MEM	WB			
`slt r5,r15,r4`					-----	-----	IF	ID	EX	MEM	WB		
Time in ps	250	250	250	175	200	250	250	175	175	200	200		

$\frac{\quad}{1}$

B) What is the total execution time of the above instructions in ps? Place your answer in the box.

Addition of all time :

2375 ps

$\frac{\quad}{1}$

C) We have seen that data hazards can be eliminated by adding nops to the code. Can you do the same with this structural hazard? Why?

Yes, we can remove the structural hazard by adding two *nops* (no operation instruction) between *beq* and *add* by using a special hardware that recognises and fetch the nops instruction because it is for the two bubbles that caused the hazard.

$\frac{\quad}{1}$

D) (4.10.2) If we change load/store instructions to use a register (without an off set) as the address, these instructions no longer need to use the ALU. As a result, MEM and EX stages can be overlapped and the pipeline has only 4 stages.

- For this problem, assume that all branches are perfectly predicted (this eliminates all control hazards) and that no delay slots are used.
- The length of the new MEM/EX stage shall be the longer of the two.

Fill in the table below to reflect the changed ISA.

If you need to use a spreedsheet to answer this, and paste that in (or attach it) that's ok too.

Pipeline slots	0	1	2	3	4	5	6	7	8	9	10	11	12
`sw 16,12(r6)`	IF	ID	EM	WB									
`lw r16,8(r6)`		IF	ID	EM	WB								
`beq r5,r4,Label # Assume r5!=r4`			IF	ID	EM	WB							
`add r5,r1,r4`				IF	ID	EM	WB						
`slt r5,r15,r4`					IF	ID	EM	WB					
Time in ps	250	250	250	250	250	200	200	200					

$\overline{}$ 1

E) Assuming this change does not affect clock cycle time, what speedup is achieved in this instruction sequence as compared to question A?

Total Time in D=250*5+200*3=1850ps
Speedup = total time in A/total time in D = 2375/1830 =1.297 times speedup.

$\overline{}$ 1

F) **A**ssuming stall-on-branch and no delay slots, determine the execution time if branch outcomes are determined at the Execute stage by filling in the chart below.
If you need to use a spreedsheet to answer this, and paste that in (or attach it) that's ok too.

Pipeline slots	0	1	2	3	4	5	6	7	8	9	10	11	12
`sw 16,12(r6)`	IF	ID	EX	MEM	WB								
`lw r16,8(r6)`		IF	ID	EX	MEM	WB							
`beq r5,r4,Label # Assume r5!=r4`			IF	ID	EX	MEM	WB						
`add r5,r1,r4`				------	-----	IF	ID	EX	MEM	WB			
`slt r5,r15,r4`							IF	ID	EX	MEM	WB		
Time in ps	250	250	250	175	200	250	250	175	175	200	200		

$\overline{}$ 1

G) **A**ssuming stall-on-branch and no delay slots, determine the execution time if branch outcomes are determined at the ID stage by filling in the chart below.
If you need to use a spreedsheet to answer this, and paste that in (or attach it) that's ok too.

Pipeline slots	0	1	2	3	4	5	6	7	8	9	10	11	12
`sw 16,12(r6)`	IF	ID	EX	MEM	WB								
`lw r16,8(r6)`		IF	ID	EX	MEM	WB							
`beq r5,r4,Label` `# Assume r5!=r4`			IF	ID	EX	MEM	WB						
`add r5,r1,r4`				------	IF	ID	EX	MEM	WB				
`slt r5,r15,r4`						IF	ID	EX	MEM	WB			
Time in ps	250	250	250	175	250	250	200	175	200	200			

1

H) (4.10.3) What speedup is achieved on this code if branch outcomes are determined in the ID stage, relative to the execution where branch outcomes are determined in the EX stage?

Total Time in F = 2375, Total Time in G = 2200
Speedup = total time in F/Total time in G = 2375/2200 = 1.079 ps

1

I) (4.10.4) Repeat the speedup calculation from D, but take into account the (possible) change in clock cycle time. When EX and MEM are done in a single stage, most of their work can be done in parallel. As a result, the resulting EX/MEM stage has a latency that is the larger of the original two, plus 80 ps needed for the work that could not be done in parallel.

The clock cycles required as in D is 7 to complete. Latency of new stage (EX/MEM) is 175+80=255 ps which is longer than other latency. That is why the minimum clock cycles in this case is 255 ps.
The latency of 8 clock cycles is 250 ps
So, speedup = 8(250)/7(255) = 1.120

1

J) (4.10.5) Repeat the speedup calculation from F, taking into account the following (possible) change in clock cycle time: assume that the latency ID stage increases by 50% and the latency of the EX stage decreases by 10ps when branch outcome resolution is moved from EX to ID.

Increase 50% for ID = 100 + (50/100)100 = 150
Decrease 10ps for EX = 175-10 = 165
New latencies : IF=250 ID=150,EX=165,MEM=150,WB=200
Highest latency is still 250 and so it is minimum clock cycle time, so the speedup = 11(250)/10(250) = 1.1

1

K) (4.10.6) What is the new clock cycle time and execution time of this instruction sequence if `beq` address computation is moved to the MEM stage?

- Assume stall-on-branch and no delay slots
- Assume that the latency of the EX stage is reduced by 20 ps and the latency of the MEM stage is unchanged when branch outcome resolution is moved from EX to MEM.

If you need to use a spreedsheet to answer this, and paste that in (or attach it) that's ok too.

latency of EX = 155 ps

Pipeline slots	0	1	2	3	4	5	6	7	8	9	10	11	12
`sw 16,12(r6)`	IF	ID	EX	MEM	WB								
`lw r16,8(r6)`		IF	ID	EX	MEM	WB							
`beq r5,r4,Label` **`# Assume r5!=r4`**			IF	ID	EX	MEM	WB						
`add r5,r1,r4`				----	---	----	IF	ID	EX	MEM	WB		
`slt r5,r15,r4`								IF	ID	EX	MEM	WB	
Time in ps	250	250	250	155	200	200	250	250	155	155	200	200	

1

What is the speedup from this change?

Latency of EX = 175-20 = 155 ps
Speedup = 11(250)/9(250) = 1.99

1

Question 2 (10 marks total):

(4.19) This exercise explores energy efficiency and its relationship with performance. Problems in this exercise assume the following energy consumption for activity in Instruction memory, Registers, and Data memory. You can assume that the other components of the datapath spend a negligible amount of energy.

Assume that components in the datapath have the following latencies. You can assume that the other components of the datapath have negligible latencies.

I-Mem	Control	Register Read or Write	ALU	D-Mem Read or Write
~~200ps~~	~~150ps~~	~~90ps~~	~~90ps~~	~~250ps~~

Use the following Sri Supplied values instead of the textbook values.

I-Mem	Control	Register Read/Write	ALU	D-Mem R/W
150ps	**175ps**	**80ps**	**75 ps**	**225ps**

A) **(4.19.1** [10])(Based on sections 4.3, 4.6, 4.14) How much energy is spent to execute an ADD instruction in a single-cycle design?

For add instruction, I-mem, register read 2 times and register write once.
Total energy spent =140 + 2*70 + 60
= 340 pJ

$\frac{\quad}{1}$

B) How much energy is spent to execute an ADD instruction in the 5-stage pipelined design?

IF : I-mem
ID: 2 reads of registers
Wb: register write for output
Total energy consumed = 140 + 2*70 + 60
= 340 ps

$\frac{\quad}{1}$

C) **(4.19.2** [10]) (Based on sections 4.6, 4.14) What is the worst-case MIPS instruction in terms of energy consumption, and what is the energy spent to execute it?

Instruction: Load uses both 2 read and writes so it is the worst case

Energy used = 1*I-mem + 2*R-read + 1*R-write + 1*D-Mem Read
= 1*140 + 2*70 + 1*60 + 1*140
= 480 pJ

$\frac{\quad}{1}$

D) **(4.19.3** [10]) (Based on sections 4.6, 4.14) If energy reduction is paramount, how would you change the pipelined design?

Add control signal so reading registers is not done unnecessarily and reduce LW instruction to only use 1 register read and not 2 register reads so 80ps reduction. The memory load is eliminated from pipelining stages.

$\overline{}$ 1

E) What is the percentage reduction in the energy spent by an LW instruction after the change in part D?

Reduction = 480 – 340
= 140 pJ
Percentage reduction = (140/480)*100 = 0.29167 = 29.167%

$\overline{}$ 1

F) **(4.19.4** [10]) (Based on sections 4.6, 4.14) What is the performance impact of your changes from part D?

The latency increases for ID stage as there is only one read instead of two and so this can delay the clock cycle time anywhere the ID stage is slowest in the process stage. Register reads also won't be overlapped. The store instruction will also get benefitted as it is needed to perform the memory write, this operation can be skipped by using forwarding from memory. The reduction in energy will be there if store instruction is followed by other instructions.

$\overline{}$ 1

G) **(4.19.5** [10]) (Based on sections 4.6, 4.14) We can eliminate the MemRead control signal and have the data memory be read in every cycle, i.e., we can permanently have MemRead=1. Explain why the processor still functions correctly after this change.

MemRead is used for loading instructions, unloading instructions from register, and for stalls like instructions. When MemRead is set to 1 permanently, there will be memory read possible in every process. Clock cycle time already includes for memory reads in MEM stage so this wouldn't lead to major changes to clock cycle time. Also, no energy changes because when load instructions is in MEM stage, without loading the instructions, the memory read occurs.

$\overline{}$ 1

H) What is the effect of the change in part G on clock frequency and energy consumption?

Clock cycle time already includes for memory reads in MEM stage so this wouldn't lead to major changes to clock cycle time.
Since clock frequency = 1/clock cycle time, and there are minor changes in clock cycle time, thus there would be minor increase in clock frequency.

Also, no energy changes because when load instructions is in MEM stage, without loading the instructions, the memory read occurs.

$\overline{}$ 1

I) (**(4.19.6** [10]) **(Based on sections 4.6 and 4.14)** Because the frequency of the clock is limited by the component in the data path with the longest execution time, some components will be idle for some of each clock pulse, even when they are active during the cycle.

If an idle unit spends 10% of the power it would spend if it were active, what is the energy spent by the instruction memory in each cycle?

Idle Energy = 10% * Imem ActiveEnergy * (clockcycle time – Imem Latency)/Imem Latency
= 0.1 * 140 pJ * (225 – 150)/150
= 14 * 75/150
= 7 pJ
I-Mem energy = I-mem Active Energy + Idle Energy
= 140 pJ + 7 pJ
= 147 pJ is the energy spent by instruction memory in each cycle
Control 175ps of 225
ALU 75ps of 225 again
Register 70ps of 225 (@ 70pj in the worst case, 10% = 7 pj so 155/70 * 7 pj = 15.5 pj to idle)

$\overline{}$ 1

J) What percentage of the overall energy spent by the instruction memory does this idle energy in part I represent?

So ~22.5 pj in idle power loss. Since all possible units can be executing at once, that means 530 pj of power could be in use all at once (not counting idleness). So 22.5/552.5 = 4.07% of power is lost to idle.
Percentage of Idle Energy/ I-Mem energy = (7/147)*100
= 4.07%

$\overline{}$ 1

Question 3: 2 marks

A) We haven't discussed it in class, so you'll need to do a tiny bit of research. How many words must be accessed for DDR4-3600 with CAS latency 16 to be faster than DDR-4 4200 with CAS latency 19? (show your calculation – the Wikipedia page on CAS latency has both the formula and the answer buried in a table). In this case a word is most likely 64 bits of memory, but that doesn't actually matter.

Access time for first word by DDR4-3600 CAS latency 16 = (16/3600)*2000=8.89nanosecond
For 4th word=9.72ns, for 8th word=10.83ns
Access time for first word by DDR4-4200 CAS latency 19 = (19/4200)*2000=9.05nanosecond
For 4th word=9.76ns,for 8th word= 10.71ns
So at most 4 words must be accessed by DDR4-3600 to be faster than DDR4-4200

B) How many words must be accessed for the DDR4-3600 memory to be slower?

$\frac{}{1}$

Access time for first word by DDR4-3600 CAS latency 16 = (16/3600)*2000=8.89nanosecond
For 4th word=9.72ns, for 8th word=10.83ns
Access time for first word by DDR4-4200 CAS latency 19 = (19/4200)*2000=9.05nanosecond
For 4th word=9.76ns,for 8th word= 10.71ns
So at least 8 words must be accessed by DDR4-3600 to be slower than DDR4-4200

Programming questions (16 marks total):

Question 3: 2D array Programming (12 marks):

I generated two sample arrays for this question using
https://www.random.org/integers/?num=16&min=1&max=10&col=4&base=10&format=html&rnd=new

2	1	9	2	8	7	1	2
7	9	10	10	2	7	3	6
3	4	3	4	7	5	6	8
2	5	4	6	9	4	8	9

$$A = \begin{pmatrix} 2 & 1 & 9 & 2 \\ 7 & 9 & 10 & 10 \\ 3 & 4 & 3 & 4 \\ 2 & 5 & 4 & 6 \end{pmatrix}, B = \begin{pmatrix} 8 & 7 & 1 & 2 \\ 2 & 7 & 3 & 6 \\ 7 & 5 & 6 & 8 \\ 9 & 4 & 8 & 9 \end{pmatrix},$$

Create those matrices in MIPS and calculate A+B and A*B, calculate the Transpose of A.

Test your program with at least one other pair of random matrices but limit yourself to 4x4.

Take screen captures of each result and include them in a word document along with a description of what each screen capture is testing.

(row*rowsize)+(col*colsize)

Question 4: Floating point Programming (4 marks)

Write a program in MIPS to prove that for (single precision) floating points a+(b+c) does not necessarily = (a+b)+c (you should be able to do this with a = 0.01, b = 0.02, c=0.03) – tip, turn on values being displayed in hexadecimal on in program settings.

Calculate the error between the two values in decimal (feel free to use a calculator for this one).

Take screen captures of your test results and include them in a word document along with a description of what each screen capture is testing.

Yes, I realise Programming Q1 and Q2 are not even remotely close to each other in complexity.