National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Computer Architecture**	**Course Code:**	**EE204**
Program:	**BS(Computer Science)**	**Semester:**	**Spring 18**
Duration:	**180 Minutes**	**Total Marks:**	**70**
Date:	**16-05-2018**	**Weight**	**45%**
Section:	**All**	**Page(s):**	**8**
Exam:	**Final**	**Student Id:**	

NOTE: Answer in the space provided. You can ask for rough sheets but they won't be graded.
In case some information is missing, make assumption and mention it at the top of your solution.

Question 1 [Marks: 2+2+2+2+2]

Provide reasoning for the following questions!

1. **We may need to add a cycle before and a cycle after branch instruction to solve hazards. Does both of these stalls are due to control hazard?**

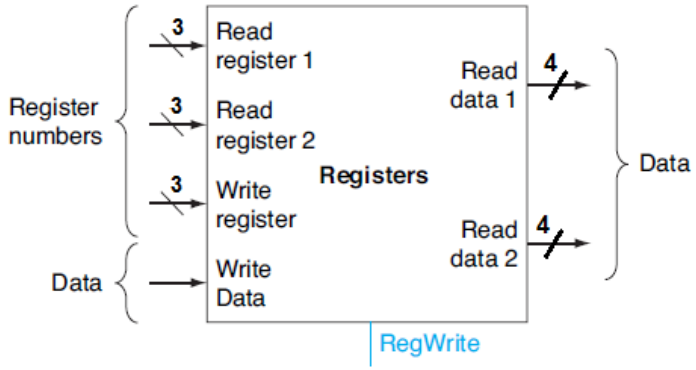
2. **Stalls can be added to solve hazards. Can we solve false dependencies (WAR, WAW) using stalls?**

3. **What is execution hazard in RAW? If forwarding is implemented, do we still need stalls to remove this hazard?**

4. **Why we need loop unrolling? Suppose a loop has 22 iterations and we want to unroll it by degree 3. In this scenario, how we will execute 22 iterations?**

5. **What is a VLIW processor? Does VLIW processor always provide speedup of more than 1?**

Question 2: [Marks: 2 +3 + 5]



This is the diagram of Register file which is built using different components (Registers, Decoders, multiplexers).

a) How many registers are there in this register file and what is the size of each register?

b) In order to implement complete functionality of this register file, which components are required, in what quantity and what would be the size of each component (number of inputs and outputs).

c) Draw circuit with all Registers and circuitry required to select a particular register by reading port **Read register 1** and providing data on **Read data 1**. Detail of other functionalities is not required!

Question 3: [Marks: 8+2+3+2]

Consider the following program fragment executing on a basic 5-stage MIPS pipeline **with no data forwarding**. All stages take 1 cycle, except the Execute stage, which takes a variable number of cycles, depending on the functional unit used:

Functional unit	Number of EX cycles
Integer ALU	1
Floating Point Add	2
Floating Point Load/Store	2
Floating Point Multiply	4

a. For each instruction, determine in which clock cycle the instruction is safely completed (i.e., when does its write back stage occur) after inserting stalls. **F** at the end of instruction op-code denotes floating point instruction e.g. MULT**F**.
Assume no data forwarding occurs. **Ignore false dependencies while scheduling.**

Instruction	Write Back Cycle
1: MULTF F1, F2, F3	
2: ADD R1, R1, R2	
3: LF F5, 0(R1)	
4: SUBF F5, F6, F1	
5: SF F5, 0(R1)	
6: MULTF F5, F3, F4	

Show timing in the table below:

Cycle / Inst	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
(1)	F	D																												
(2)																														
(3)																														
(4)																														
(5)																														
(6)																														

b. Identify false dependencies in the above code and tell whether they created any problem in the above code schedule or not.

c. Calculate the average CPI and throughput (in instructions per cycle) of the processor for the above code.

d. If we run this program on a processor with frequency of 1 MHZ, what would be its execution time?

Question No. 4 [Marks: 10]

Consider a machine with a byte addressable main memory of 64 Kbytes. Assume we have a 2-way set associative cache of 4K data bytes size and 8 bytes block size that is being used with this machine.

a. **Calculate the tag, index, and offset bits?**

b. **Into what set would bytes with each of the following addresses be stored**

Set number

- **0001 0001 0001 1011**
- **1010 1010 1010 1010**

c. **Suppose the byte with address 0001 1010 0001 1010 is stored in the cache. What are the addresses of the other bytes stored along with it in the same block?**

d. **Write two addresses that would map to same set in cache.**

e. **Calculate the size of the tag array?**

Question No. 5 [Marks: 15]

Suppose we have virtual memory in a system with page size of 4KB, Virtual address is 17 bits and physical address is 16 bits. TLB cache is fully associative with 4 entries and least recently used replacement policy. Following is the snapshot for Page table and TLB cache.

Page table:

Tag	Valid	Physical Page #
00000	0	0011
00001	1	1001
00010	1	0000
00011	1	0010
00100	1	1010
00101	0	0100
00110	1	1011
00111	0	0101
01000	1	1000
01001	1	0110
01010	1	1111
01011	1	1101
01100	1	0111
01101	0	1110
01110	1	1100
01111	1	0001

TLB cache:

Valid	LRU	Tag	Physical Page #
1	3	01111	0001
1	4	00011	0010
1	2	01000	1000
1	1	00100	1010

a. For each of the following addresses write the corresponding physical addresses and tell whether it is TLB hit/miss or page fault.

	Virtual address	Physical address	TLB hit/miss	Page fault
A	01110001000100100			
B	00011101001100000			
C	01110010100010001			
D	00101000111000101			
E	01000100110100011			

b. In case of a TLB miss, new entry comes in the TLB cache replacing the least recently used value. Value of 1 in LRU in above table represents the most recently used value and 4 represent the least recently used value. If the above addresses are accessed in same sequence, what would be the final state of TLB cache?

Valid	LRU	Tag	Physical Page #

Note: In case of a page fault, you can change valid bit to 1 in page table, after making sure that same physical page number is not assigned to another virtual page number. In that scenario first change the valid bit of that entry to 0.

Question No. 6 [Marks: 10]

Each instruction fetch means a reference to the instruction cache and 35% of all instructions reference data memory (Load/Store instructions).

With the first implementation (separate instruction and data cache):
- The average miss rate in the L1 instruction cache was 2%
- The average miss rate in the L1 data cache was 10%
- In both cases, the miss penalty (time to access main memory) is 90 CCs

For the new design, with unified memory(same cache for instruction and data), the average miss rate is 3% for the cache as a whole, and the miss penalty is again 90 CCs.

a)
Which design is better and by how much?

b) If we add L2 cache in the new design with access time equal to 10cc, it reduces the miss rate to 1.5 % from 3 %. What is new average memory access time?