


# National University of Computer and Emerging Sciences, Lahore Campus

	Course:	Computer Architecture	Course Code:	EE204
	Program:	BS(Computer Science)	Semester:	Spring 2020
	Date:	08-07-2020	Total Marks:	60
	Section:	ALL	Weight	50%
	Exam:	Final	Page(s):	5
	Time	180 + 30 Minutes		

## Instructions:

Don't forget to write your ID on top of every page.

Solution must be hand written with necessary working clearly shown.

Upload a single PDF (not word documents) containing all answers. Please place your solution in order i.e. Q1 followed by Q2 ...

Time line is strict so upload your answers within the given time limit.

You are not allowed to have any form of communication with any other person during the exam. **The question statements in red color require student specific answers and might be used to identify communication during examination**

Attempt as many questions as you can and don't be worried if the exam seems lengthy. Relative grading will take care of such a problem.

## Problem 1[Memory Cache Design]

Assume that we have a machine with a cache that can store 8K data bytes and is 4-way set-associative with LRU-policy used for data block replacement. Further assume that the line/block size of the cache is 64 byte and the machine uses 36 bit virtual addresses and 32-bit physical addresses. The cache is virtually indexed and physically tagged.

### Part a)

[2 + 3 + 1 + 2 Points]

Answer the following question for the given cache design

- How many sets does it have?
- How many bits will be used for **index, offset, and tag** in this machine?
- Consider a reference to address X. Which set does this map to in the cache? Assume that X is the four digit number in your University ID. For example, if your ID is 16L-4167 then take X = 4167**
- Including both **tag and data bits (and 1 valid bit per line)** how many total SRAM bits are needed to implement this cache?

### Part b)

[2 + 2 + 2 Points]

**Now assume that you are accessing/processing elements of a large one dimensional array A [ ] of integers such that the first element of the array is stored at address 0x100h and that an integer is stored in 4 bytes.**

**For each of the following cache HIT and cache MISS pattern, write a C++ code segment that accesses the array elements such that the given pattern of HIT and MISS is generated during the data access. If a pattern cannot be generated during memory accesses then give a concise reason to support your answer as to why such patterns cannot be generated?**

- MISS, HIT, HIT, MISS
- (HIT)<sup>10</sup> {i.e. HIT 10 times}
- MISS, (HIT)<sup>100</sup>

**Part c) {Virtual Memory}****[1 + 2 + 3 Points]**

Assume that on a system with 36-bit virtual addresses and 32-bit physical address the page size is 8KB. Furthermore, assume that each page table entry consists of a **physical page number**, 1 **valid bit**, 1 **dirty bit**

- i) How many pages a process can have?
- ii) What is the size of the page table on such a system?
- iii) What are some of the advantages of using virtual memory w.r.t process creation, loading and linking? {Search the net and explain in your own words}

**Problem 2[Multiplication Using BOOTH's Encoding]****[5 Points]**

Demonstrate multiplication of a ten bit number **X** with itself using Booth's encoding method for multiplication. Take **X** to be the value represented by the least significant three digits of your four digit university ID. For example, if your ID is 16L-4167 then take **X = 167**.

**Problem 3[MIPS and MIPS Pipeline]**

Assume that the following code segment is executed on the five stage pipeline with fully implemented hazard detection and forwarding

```
lw    $2, 100($3)
sub   $4, $3, $5
add   $5, $3, $7
addi  $7, $6, $1
sw    $7, 10($2)
```

**Part a)****[5 Points]**

Write the above code segment using 32 bit binary encoding of MIPS architecture.

**Part b)****[7 Points]**

Assume that the execution of this code started in the 1<sup>st</sup> cycle, write the values of the following control signals in the 5<sup>th</sup>, 6<sup>th</sup> and 7<sup>th</sup> cycle. Some descriptions of these control signals can be found on pages 417 and 418 of the third edition of course textbook.

Instruction	Execution/address calculation stage control lines				Memory access stage control lines			Write-back stage control lines	
	Reg Dst	ALU Op1	ALU Op0	ALU Src	Branch	Mem Read	Mem Write	Reg Write	Mem to Reg

**Part c)****[8 Points]**

Assume that following code segment is executed on the five stage pipeline with fully implemented hazard detection and forwarding

```
Or  $1, $1, $3
lw  $2, 20($1)
lw  $3, 0($1)
sw  $2, 40($1)
add $4, $5, $12
add $4, $5, $5
add $4, $6, $6
```

Assuming that the execution of code started in 1<sup>st</sup> cycle, write the values of each of the following conditions in the 5<sup>th</sup>, 6<sup>th</sup>, and 7<sup>th</sup> cycle

- MEM/WB.RegisterRd = ID/EX.RegisterRs
- MEM/WB.RegisterRd = ID/EX.RegisterRt
- MEM/WB.RegisterRd = ID/EX.RegisterRs
- MEM/WB.RegisterRd = ID/EX.RegisterRt

**Problem 4****[5 Points]**

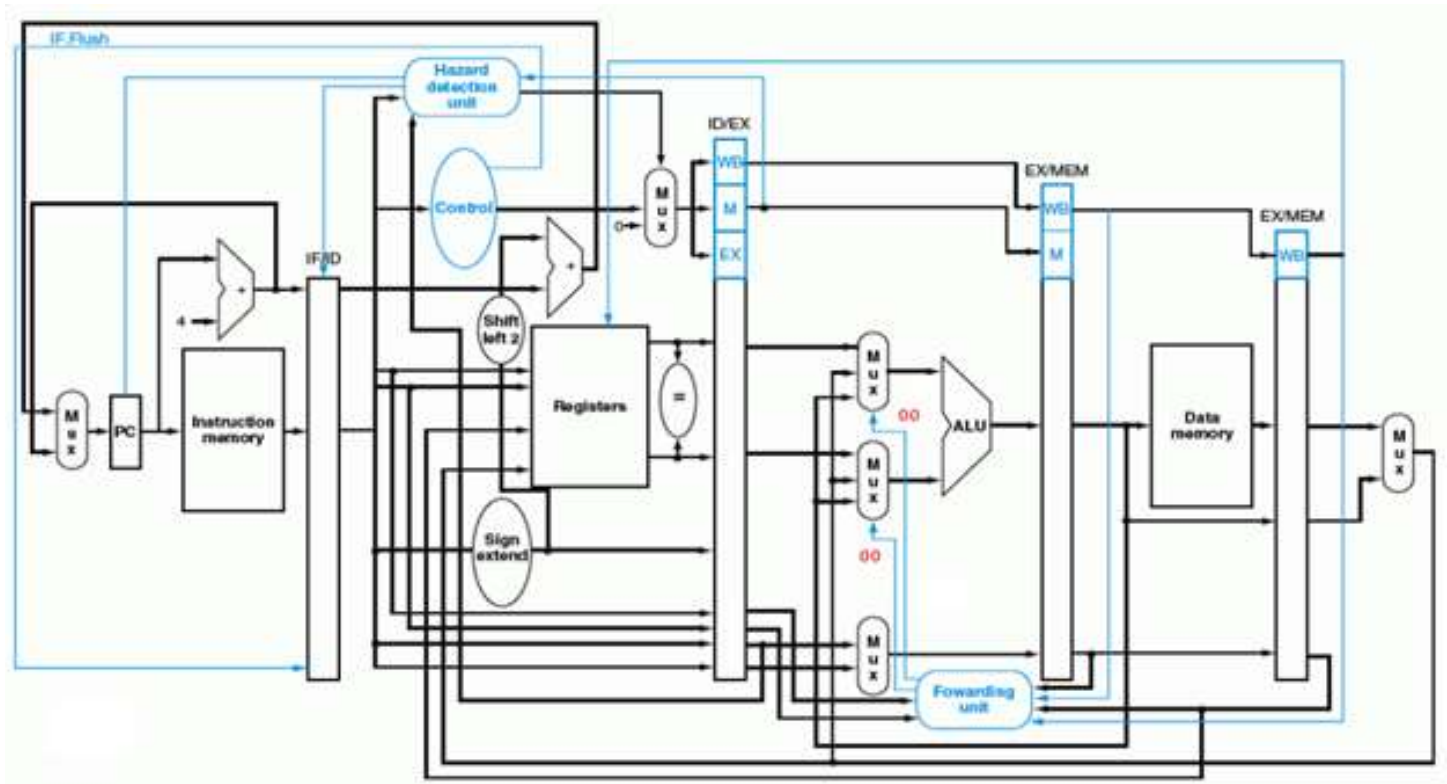
Use some form of diagram to show execution of the following three instructions on a MIPS data path shown on the last page of this exam and also given at page 614 of the textbook (3<sup>rd</sup> Edition)

lw \$4, 100(\$2)
sub \$6, \$4, \$3
add \$2, \$3, \$5

Clearly show all stalls and forwarding taking place during the execution of the three instructions and determine the number of cycles needed to completely execute the three instructions

### Problem 5[Hazard Detection and Forwarding]

In an implementation of a MIPS processor the two signals from the forwarding unit to the two processors are permanently stuck at **00** as shown in the figure below.



Part a)

[5 Points]

Assume that the following code segment is executed on this system

```

Loop: lw  $1, 20($2)
      add $1, $1, $3
      sub $6, $1, $5
      sw  $6, 20($2)
      beq $6, $7, Loop
    
```

Given the following initial values stored in various registers and assuming that all the **Data memory locations are initialized to your four digit number in your University ID**. For example, if your ID is 16L-4167 then the value 4167 is stored at each memory location.

Register Name --	\$1	\$2	\$3	\$4	\$5	\$6	\$7
>							
Register Value --	0	0	10	0	5	0	6
>							

Determine the final register values after the first iteration of the above code sequence

Part b)

[5 Points]

Repeat **part a**, assuming that the two signals from the forwarding unit to the two processors are permanently stuck at **01** instead

