B.Sc. Engg. SWE 3rd Semester — 12 March 2019 (Afternoon)

ISLAMIC UNIVERSITY OF TECHNOLOGY (IUT)
ORGANISATION OF ISLAMIC COOPERATION (OIC)
Department of Computer Science and Engineering (CSE)

MID SEMESTER EXAMINATION — **WINTER SEMESTER, 2018-2019**

DURATION: 1 Hour 30 Minutes — **FULL MARKS: 75**

CSE 4305: Computer Organization and Architecture

Programmable calculators are not allowed. Do not write anything on the question paper.

There are **4 (four)** questions. Answer any **3 (three)** of them.

Figures in the right margin indicate marks.

---

1. a) Define Throughput and Response Time. How are they affected by: 8
   i. Replacing the old processor with a new one.
   ii. Adding more processors.

   b) What is Power Wall and how has it led to the inception of Multiprocessors? What are the hurdles of parallel programming for multiprocessors? 9

   c) Consider three different processors P1, P2 and P3 executing the same instruction set with the clock rates and CPIs given in the following table: 8

Table 1: Table for question 1 (c)

Processor	Clock Rate	CPI
P1	3 Ghz	1.5
P2	1.5 Ghz	3.0
P3	2 Ghz	2.5

Answer the following questions:

i. Which processor has the highest performance?

ii. If each processor executes a program in 20 seconds, find the number of cycles and the number of instructions.

2. a) What are the different instruction formats in MIPS architecture? Give appropriate examples for each. Also mention the name and size of the fields of each of the instruction formats. 10

   b) Convert the following assembly language into its corresponding machine code and mention the value of each field of the R-format instruction. 5

```
add $s6, $t0, $s1
```

   c) Convert the C code given in Figure 1 into Assembly code. The arguments are to be stored in registers *$a0* to *$a2*, variable *i* in register *$s0*, base register of 'save' array in *$s6* and result in register *$v0*. You can also use other registers other than these if required. Make sure to add comments for each line of the assembly language: 10

```
int Mid (int j, k, p){
      int i = p;
      while (save[i]< k) j++;
}
```

Figure 1.

3. a) Explain briefly the functions of a linker and loader. 5

b) Describe the followings with appropriate example: 10

i. PC-Relative Addressing

ii. Base Addressing

c) The following problems deal with translating from MIPS to C. Assume that the variables *f*, *g*, *h*, *i*, and *j* are assigned to registers *$s0*, *$s1*, *$s2*, *$s3*, and *$s4*, respectively. Assume that the base address of the arrays *A* and *B* are in registers $s6 and *$s7*, respectively. For the MIPS assembly instructions below, what is the corresponding *C* statement? 10

a.
```
sll $s2, $s4, 1
add $s0, $s2, $s3
add $s0, $s0, $s1
```

b.
```
sll $t0, $s0, 2      # $t0 = f * 4
add $t0, $s6, $t0    # $t0 = &A[f]
sll $t1, $s1, 2      # $t1 = g *4
add $t1, $s7, $t1    # $t1 = &B[g]
lw $s0, 0($t0)       # f = A[f]
addi $t2, $t0, 4
lw $t0, 0($t2)
add $t0, $t0, $s0
sw $t0, 0($t1)
```

4. a) Define pipelining and its advantages. Use appropriate examples. 6

b) Consider the MIPS datapath shown in Figure 2. Suppose the latencies (time needed to do their work) for the logic blocks are given in Table 2: 9

Table 2: Latencies for the logic blocks

I-Mem	Add	Mux	ALU	Regs	D-Mem	Sign-extend	Shift-left
400ps	120ps	40ps	120ps	150ps	350ps	20ps	0ps

Answer the following questions:

i. What is the clock cycle time if the only type of instructions supported are the ALU instructions (`add`, `sub` etc.)?

ii. What is the clock cycle time if only `sw` instruction were to be supported?

iii. What is the clock cycle time if you must support `add`, `beq`, `lw` and `sw` instruction

c) Consider the following sequence of MIPS instructions: 5+5

```
lw $t1, 0($t0)
lw $t2, 4($t0)
add $t3, $t1, $t2
sw $t3, 12($t0)
lw $t4, 8($t0)
add $t5, $t1, $t4
sw $t5, 16($t0)
```

Answer the followings:

i. What are data hazards and how can we overcome them?

ii. Identify the hazard(s) in this instruction sequence and rearrange them so that it will execute without any stalls on a 5-stage pipelined processor.

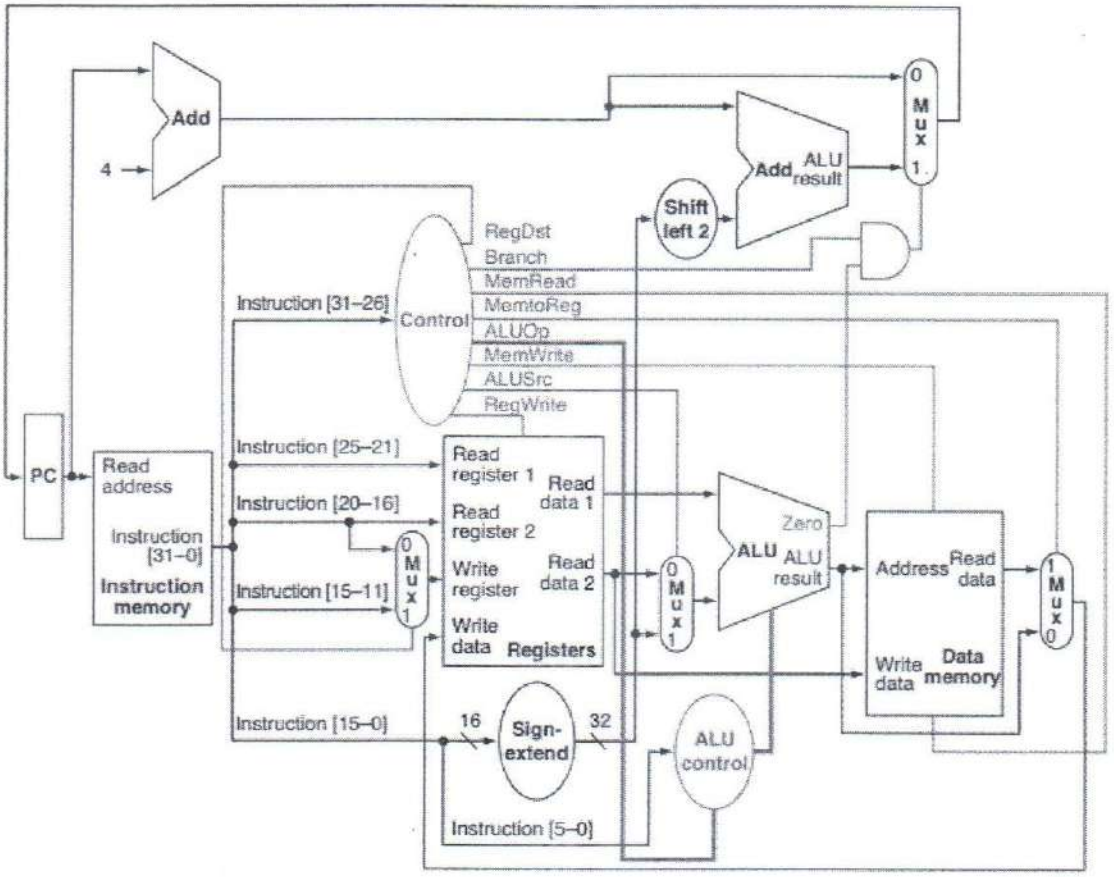

Figure 2.