Course Title and Code Number:
Computer Architectures (CS x35)
Fourth Year (Communications and Electronics)
Time Allowed: 90 Mins

اسم المقرر والرقم الكودي له:
معماريات الحاسب (CS x35)
السنة الدراسية الرابعة (اتصالات و الكترونيات)
الزمن: ٩٠ دقيقة

Name: **Seat number:**

Answer only three questions in the dedicated space: **(25 marks)**

Question 1: **(8 marks)**

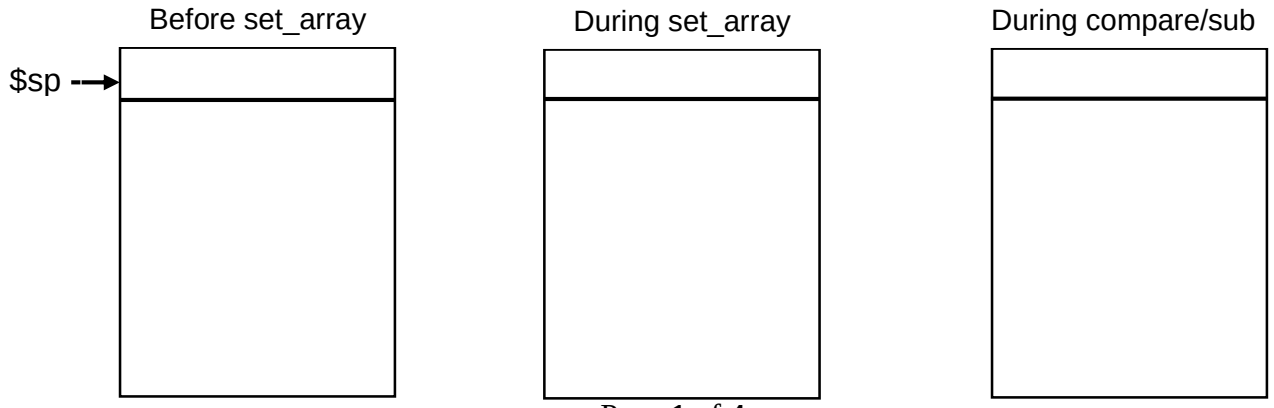
Consider the following C code snippet.

```
// C code
void setArray(int num) {
int i;
int array[3];
for (i = 0; i < 3; i = i + 1)
{
array[i] = compare(num, i);
      }
}
```

```
int compare(int a, int b) {
if (sub(a, b) >= 0)
return 1;
else
return 0;
}
int sub(int a, int b) {
return a - b;
}
```

a) Implement the C code snippet in MIPS assembly language. Use $s0 to hold the variable i. Handle the stack pointer appropriately. The array is stored on the stack of the setArray function.

b) Assume setArray is the first function called. Draw the status of the stack before calling setArray and during each function call. Indicate the names of registers and variables stored on the stack, mark the location of $sp, and clearly mark each stack frame.



Question 2: **(8 marks)**

a) The following instructions, in MIPS assembly, represent a control structure very common in high-level programming languages (Java, Ada, C ...). Which structure is it?

```
slt $t0, $a0, $a1
bne $t0, $zero, cout
     ... instructions...
cout :
```

---

b) Write a MIPS assembly program equivalent to the following pseudo-instructions. If necessary, you can use register $t0 to memorize intermediary values. No other register can be used.

i. `add ($s0),$s1,($s2)` `#mem[$s0]=$s1+mem[$s2]`

This MIPS instruction does not exist, because it uses an addressing mode not supported by RISC processors.

ii. `SWAP $s0` `#bits 31-16 <-> bits 15-0`

This instruction allows us to swap the 16 most significant bits with the 16 least significant ones of a 32-bit word.

iii. `PUSH $s0`

This instruction is not a MIPS instruction either. It decrements the stack pointer (SP), then saves $s0 at this address.

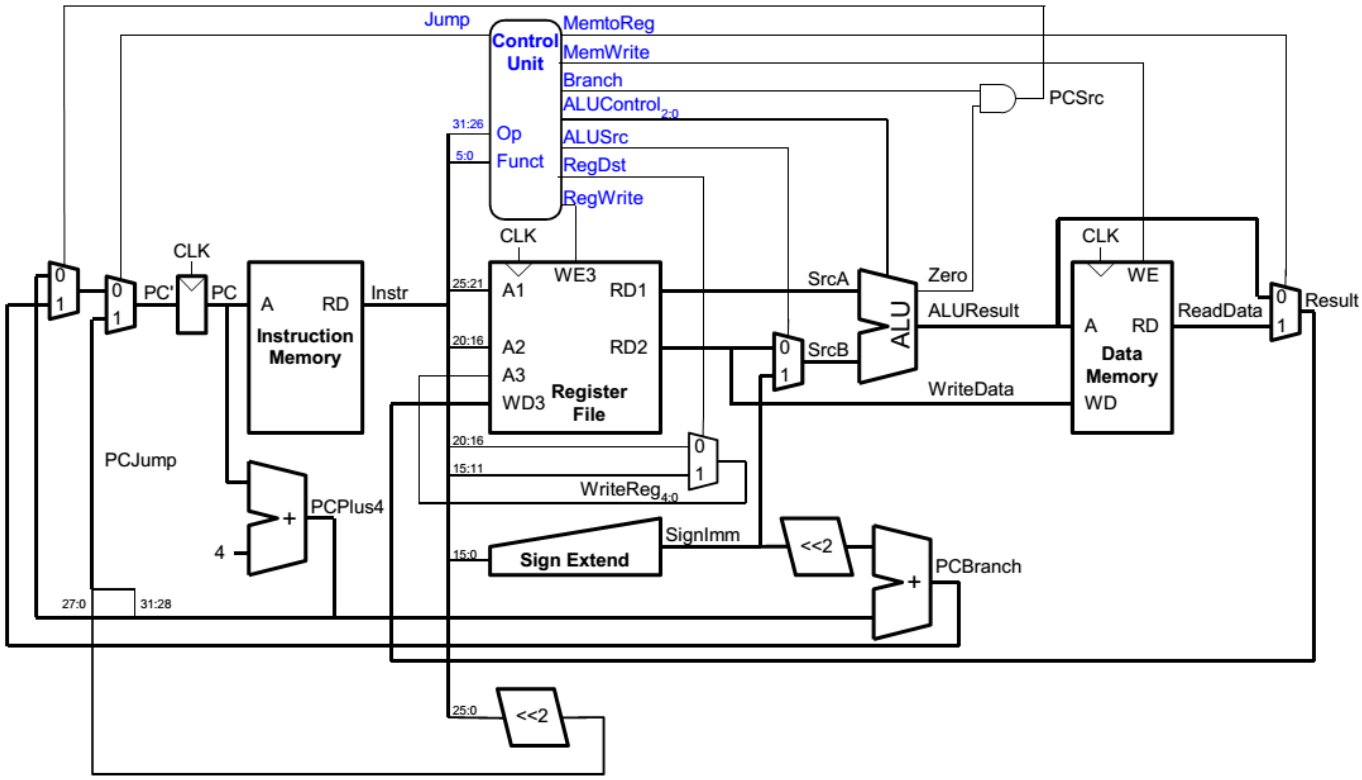
Question 3: **(8 marks)**

Modify the single-cycle MIPS processor to implement one of the following instructions. Indicate the changes to the datapath on the following figure. Name any new control signals and complete the table to indicate changes required to the control signals.

a. `jal label` `#jump and link $ra = PC + 4, PC = JTA`

b. `lh rt, imm(rs)` `#load halfword [rt] = SignExt ([Address]15:0)`

c. `jr rs` `#jump register PC = [rs]`



Instruction	$Op_{5:0}$	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	$ALUOp_{1:0}$	Jump		
R-type	000000	1	1	0	0	0	0	10	0		
lw	100011	1	0	1	0	0	1	00	0		
sw	101011	0	X	1	0	1	X	00	0		
beq	000100	0	X	0	1	0	X	01	0		
j	000100	0	X	X	X	0	X	XX	1		
jal	000011										
lh	100001										
jr	001000										

Question 4: **(8 marks)**

a) An architecture engineer is contemplating building the single-cycle MIPS processor in a 65 nm CMOS manufacturing process. He has determined that the logic elements have the delays given in Table1. Help him compute the execution time for a program with 100 billion instructions.

Element	Parameter	Delay (ps)
register clk-to-Q	t_{pcq}	30
register setup	t_{setup}	20
multiplexer	t_{mux}	25
ALU	t_{ALU}	200
memory read	t_{mem}	250
register file read	$t_{RF\text{read}}$	150
register file setup	$t_{RF\text{setup}}$	20

---

b) The engineer is allowed to optimize only a single element to minimize its delay to the half. Which element should he choose and what will be the program execution time after this step.

Good Luck

Dr. Mohammed Morsy

The MIPS Processor Instruction Format

Field	0	rs	rt	rd	shamt	funct
Bit positions	31:26	25:21	20:16	15:11	10:6	5:0

a. R-type instruction

Field	35 or 43	rs	rt	address
Bit positions	31:26	25:21	20:16	15:0

b. Load or store instruction

Field	4	rs	rt	address
Bit positions	31:26	25:21	20:16	15:0

c. Branch instruction