

Guidelines

If you found any ambiguity in any of the questions or there appears to be a lack of information, then write an assumption on the answer sheet to explain your side interpretation of the problem and solve accordingly.

Problem I: Assembly Program Encoding and Caller-Callee concept:**10 Points**

A microprocessor has 32 general-purpose registers. R31 is the Link/Return_Address register(LR), and R30 is the stack pointer(SP). Registers R27-R31 are callee-saved, and Register R1-R5 are caller-saved registers. It is recommended to use R1, R2, R3 for passing arguments to the caller function and R4, R5 to return the result from the called subroutine.

ISA followed by the processor and available to the programmer for the Algorithm encoding.

Instruction (Mnemonic, Operands)	Operation Performed
MOV R1, R2	Copy the content of right_hand side specified register(R2) into left_hand_side specified register(R1).
MVI R1, #Imm	Copy the Immediate value into the specified register(R1).
ADD R1, R2	Add the content of specified registers and store the result in Left-side specified register(R1).
ADI R1, #Imm	Add the content of specified registers with Immediate value and store in the specified register(R1).
MUL R1, R2	Multiply reg R1, R2 and store the result into the left_hand_side specified register(R1).
LD R1, [R2]	Load the content from the address, which is stored inside the specified register(R2) into specified register(R1).
ST R1, [R2]	Store the content of register R1 to the address which is stored in specified register(R2).
PUSH R1	Push the content of the specified register(R1) into stack memory and decrement the stack pointer.
POP R1	Increment the stack pointer to the next location and then pop the content from the top of the stack into the specified register(R1).
BL address	Branch and Link to the specified address.
HALT	Stop the program execution immediately.
RET	Return from the subroutine and jump to the address specified in the link register.

Please encode the below-mentioned algorithm into an assembly program.

Algorithm of the program to be converted: Partial Logistic Map.

Note: “mem” is an arbitrary-chosen memory location. The initial value of $x(x_{initial})$ is 3. And the parameter a, b have the value 1,2 respectively.

----- Program pseudocode begins -----

```

|
new_mem_location = mem
old_x = x_initial

xn_dot = xn_dot_calculator(old_x, a, b)
new_mem_location = new_mem_location + 4
new_x = old_x + xn_dot
old_x = new_x
Store “new_x” at [new_mem_loacaton]

xn_dot = xn_dot_calculator(old_x, a, b)
new_mem_location = new_mem_location + 4
new_x = old_x + xn_dot
old_x = new_x
Store “new_x” at [new_mem_loacaton]
Stop

```

```

xn_dot_calculator(old_x, a, b)
    temp = a
    temp = temp * old_x
    temp = temp + b
    temp = temp * old_x
    Return (temp)

```

----- Program pseudocode ends -----

Problem II: Processor Pipeline and execution time:

7.5 Points

Consider a five-stage pipeline where pipeline stages F, D, E, M, and W take 3ns, 4ns, 6ns, 8ns, and 6ns, respectively. There are intermediate storage buffer after each stage and delay of each buffer is 2ns. Assuming each pipeline stage takes one clock cycle to complete, determine the minimum clock period needed to ensure that the pipeline works properly. Also, determine the time needed for the implementation of the following program.

```
----- Assembly Program begins -----
MUL R4, R3, R7
SUB R1, R4, R2
ADD R3, R5, R6
ADD R2, R1, R7
SUB R9, R2, R8
HALT
----- Assembly Program ends -----
```

Evaluation Note: One mark for each correct instruction pipeline (no marks for HALT). One mark for correct minimum clock period. Execution time calculation account for 1.5 marks.

Problem II: Data Hazard:

5 Points

Find all possible types of Data hazards (RAW, WAR, WAW, RAR) in the below-mentioned assembly program.
For any instruction, the syntax is:

```
INST Dest.Register, Src.Register1, Src.Register2
OR
INST Dest.Register/Src.Register, #ImmValue
----- Assembly Program begins -----
1) AND R1,R2,R3
2) SUBI R2, #3
3) AND R1,R6,R3
4) ADD R5,R6,R7
5) MUL R4,R1,R2
6) ST  R2, [R3]
7) MUL R7, R8, R2
----- Assembly Program ends -----
```

Problem IV: Processor Performance evaluation

1.5 + 1 = 2.5 Points

Two Processors are following the same ISA.
Processor-A operates at 10GHz and executes 0.2 cycles per instruction.
Processor-B operates at 12 GHz and executes 0.5 cycles of instruction.
Calculate the number of Instructions executed by each processor in one second, And which one will serve better in terms of Instructions execution performance.