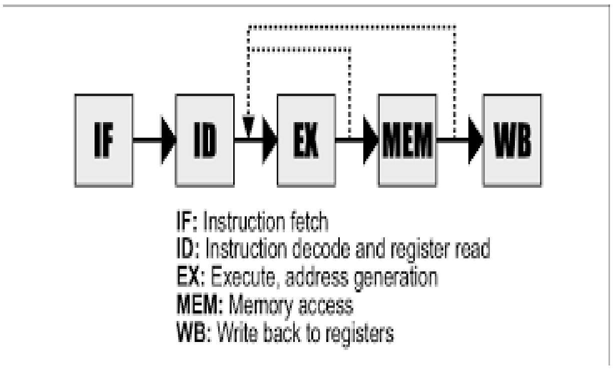
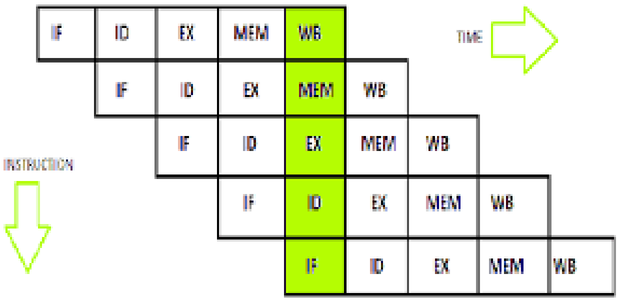
1. Why do we need processor pipelining?
   Processor pipelining improves performance by breaking down instructions into stages and executing multiple instructions simultaneously. This allows for higher throughput and more efficient use of CPU resources.
2. What will happen if we don't use any pipelining at all?
   Without pipelining, each instruction would be processed one at a time, leading to longer execution times and less efficient CPU utilization. This would result in lower overall performance and throughput.
3. Draw a 5-stage pipeline and explain the operation in each stage.



IF: Instruction fetch
ID: Instruction decode and register read
EX: Execute, address generation
MEM: Memory access
WB: Write back to registers



IF	ID	EX	MEM	WB				
	IF	ID	EX	MEM	WB			
		IF	ID	EX	MEM	WB		
			IF	ID	EX	MEM	WB	
				IF	ID	EX	MEM	WB

   The five stages are:
   - Instruction fetch: A new instruction is fetched from the memory location pointed to by the intrusion register.
   - Instruction decode: The fetched instruction is decoded to determine the operation and operands.
   - Execute: The decoded instruction is executed, performing the required operations.
   - Memory access: If the instruction involves memory operations, such as loading or storing data, this stage handles those.
   - Write back: The results of the execution are written back to the appropriate registers or memory locations.
4. List the advantages and disadvantages of processor pipelining.
   Advantages
   - Increased Throughput: More instructions are completed in a given period because different stages of multiple instructions are processed simultaneously.
   - Improved CPU Utilization: Each stage of the pipeline is utilized effectively, leading to a better overall use of CPU resources.
   - Faster Instruction Execution: The time required to complete each instruction is reduced because different parts of multiple instructions are processed in parallel.

   Disadvantages
   - Pipeline Hazards: Issues such as data hazards (dependencies between instructions), control hazards (branch instructions), and structural hazards (resource conflicts) can reduce efficiency.
   - Complexity: Implementing and managing pipelining requires more complex hardware and control mechanisms.
   - Stall Cycles: Pipeline stalls or delays may occur due to hazards, which can negate some of the performance benefits.
5. Consider a pipeline having 5 phases with duration as below:
   a. Fetch - 60ns
   b. Decode - 50ns

c. Execute - 90ns
d. Memory access - 100ns
e. Write back - 150ns
f. Latch delay - 25ns

Find the Pipeline and Non-pipeline execution time.

Non-pipeline execution time = Total Time=60ns+50ns+90ns+100ns+150ns+25ns=475ns

Pipeline execution time = Pipeline Latency=Time of Slowest Stage+Latch Delay=150ns+25ns=175ns