SNS COLLEGE OF ENGINEERING

Kurumbapalayam (PO), Coimbatore – 641 107
Accredited by NAAC-UGC with 'A' Grade
Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

DEPARTMENT OF INFORMATION TECHNOLOGY

COURSE NAME: 19IT301 COMPUTER ORGANIZATION AND ARCHITECTURE

II YEAR/ III SEM

Unit 3 : Processor and Pipelining

Topic 5: Pipelining – Basic Concepts

Overview - Pipelining

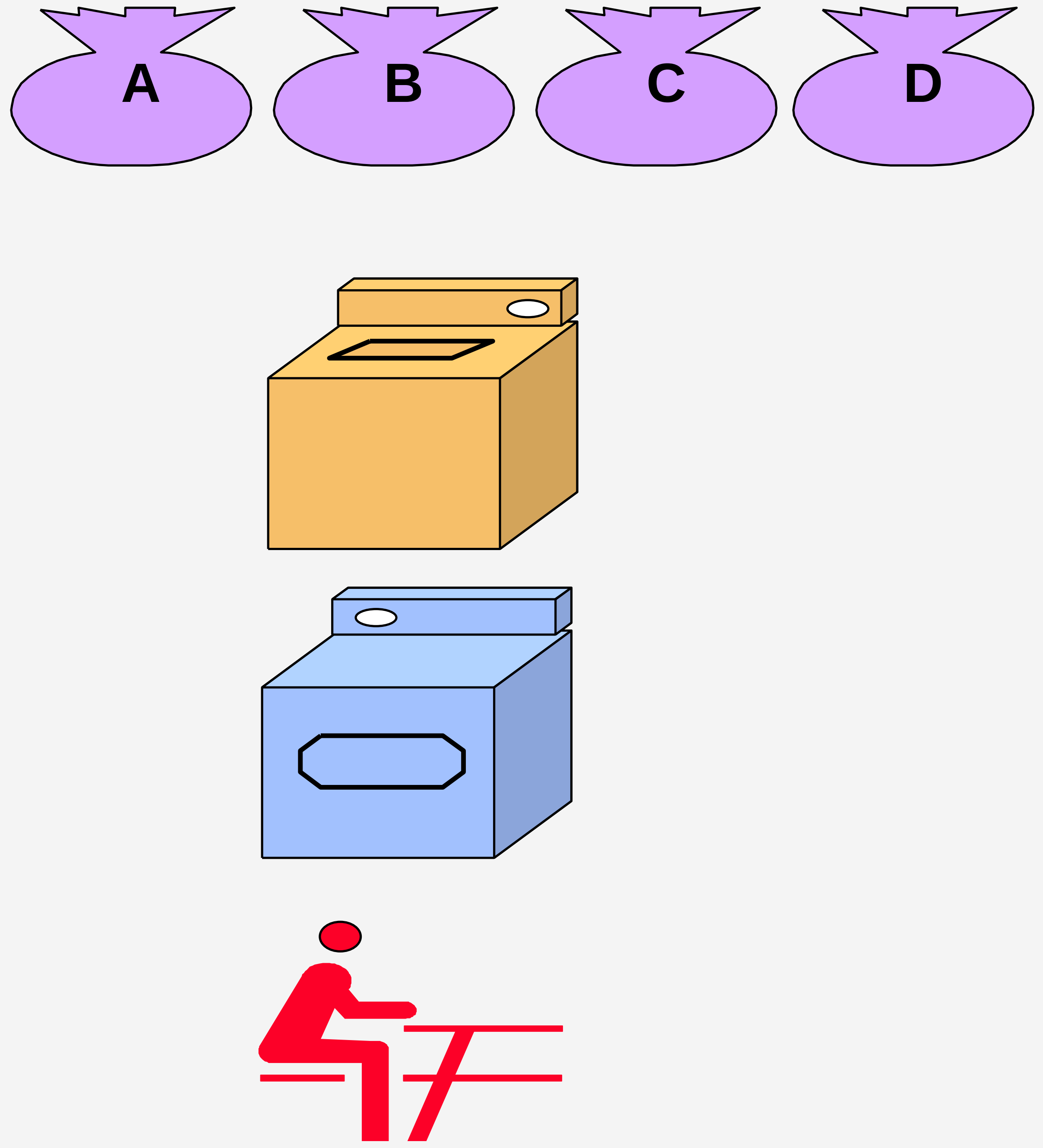
- Pipelining as a means for executing machine instructions concurrently
- It is widely used in modern processors.
- Pipelining improves system performance in terms of throughput.
- Pipelined organization requires sophisticated compilation techniques.

Basic Concepts

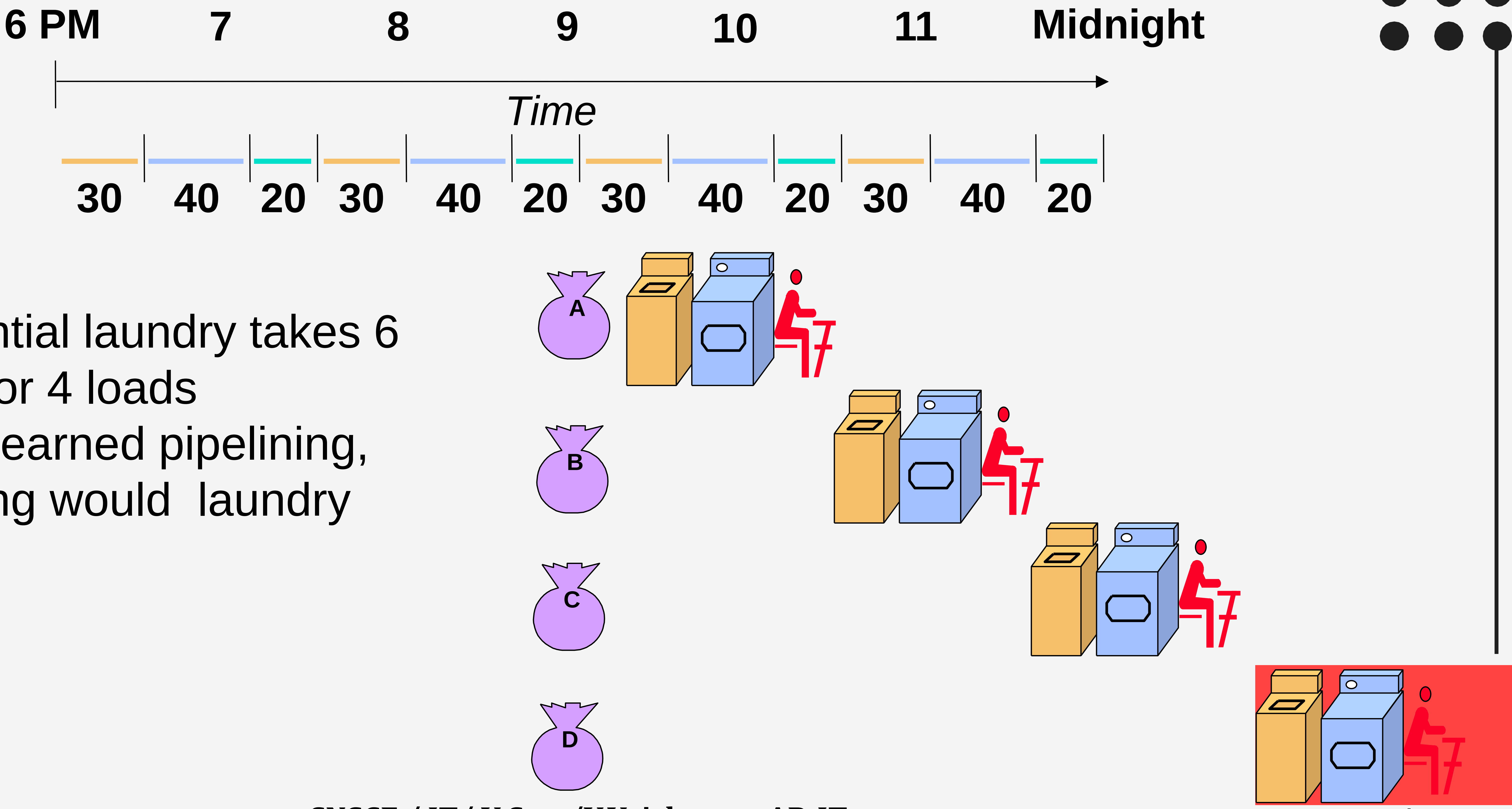
- Use faster circuit technology to build the processor and the main memory.
- Arrange the hardware so that more than one operation can be performed at the same time.
- In the latter way, the number of operations performed per second is increased even though the elapsed time needed to perform any one operation is not changed.

Traditional Pipeline Concept

- Laundry Example
- Anu, Bala, Cauvery, Dhanu each have one load of clothes to wash, dry, and fold
- Washer takes 30 minutes
- Dryer takes 40 minutes
- "Folder" takes 20 minutes



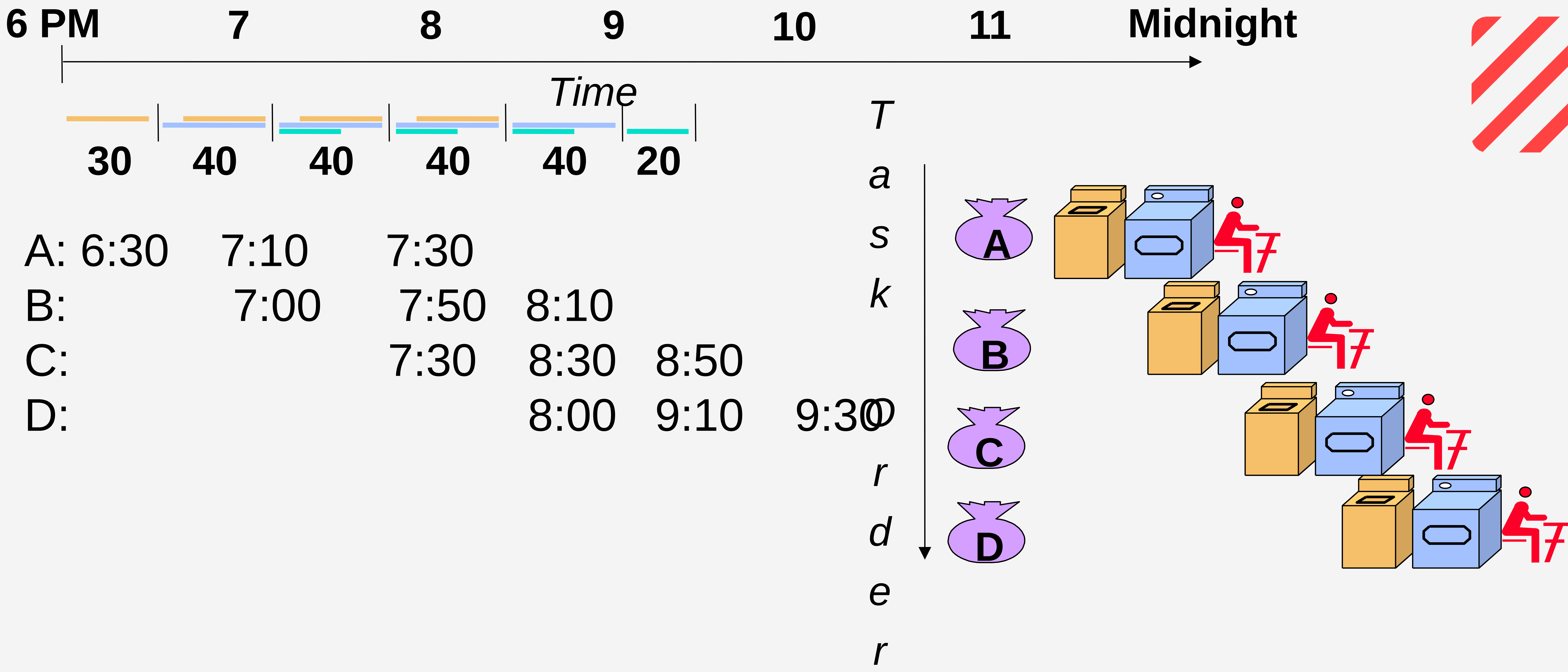
Traditional Pipeline Concept



- Sequential laundry takes 6 hours for 4 loads
- If they learned pipelining, how long would laundry take?

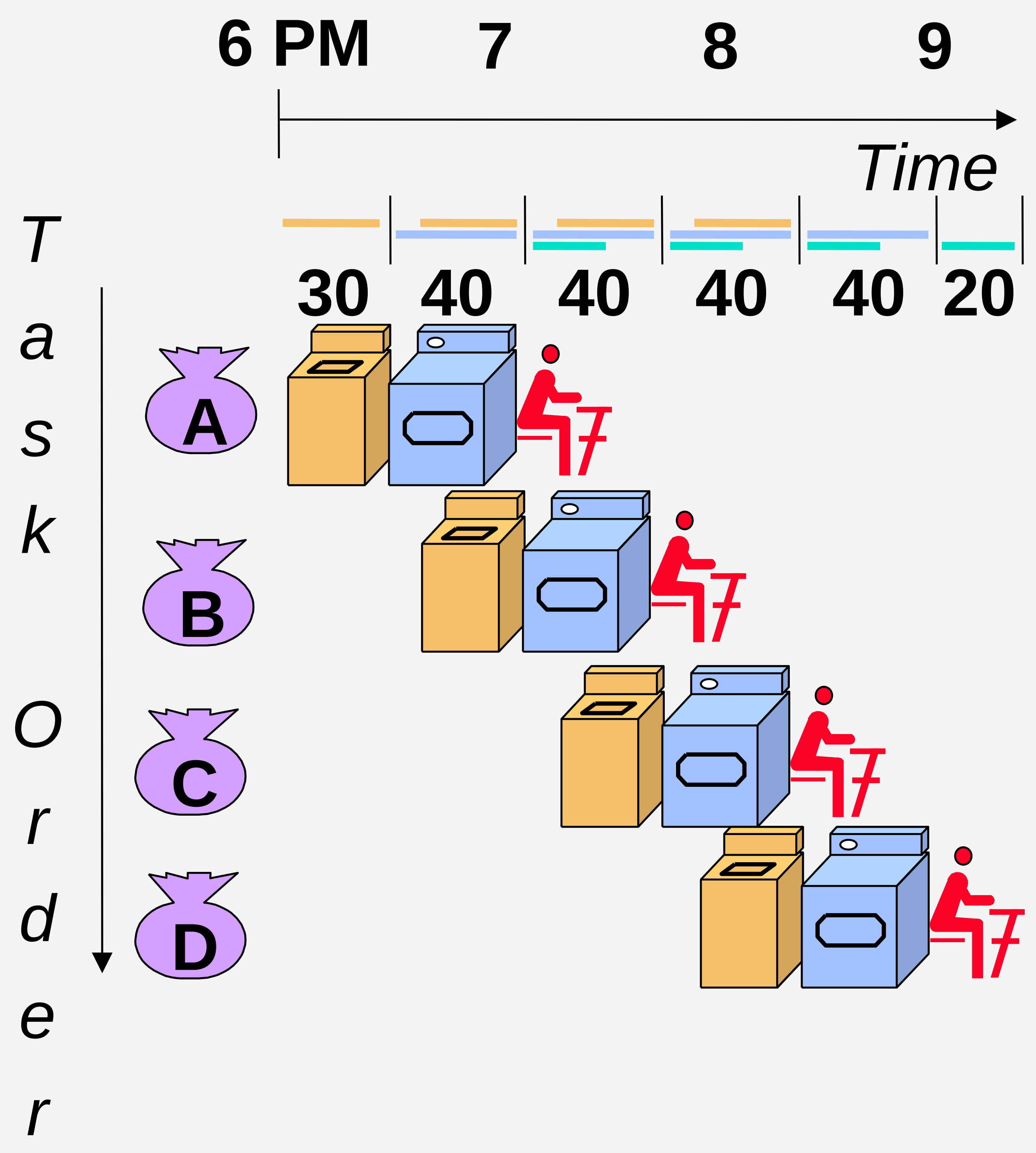
Traditional Pipeline Concept

- Pipelined laundry takes 3.5 hours for 4 loads



A:	6:30	7:10	7:30		
B:		7:00	7:50	8:10	
C:			7:30	8:30	8:50
D:				8:00	9:10	9:30

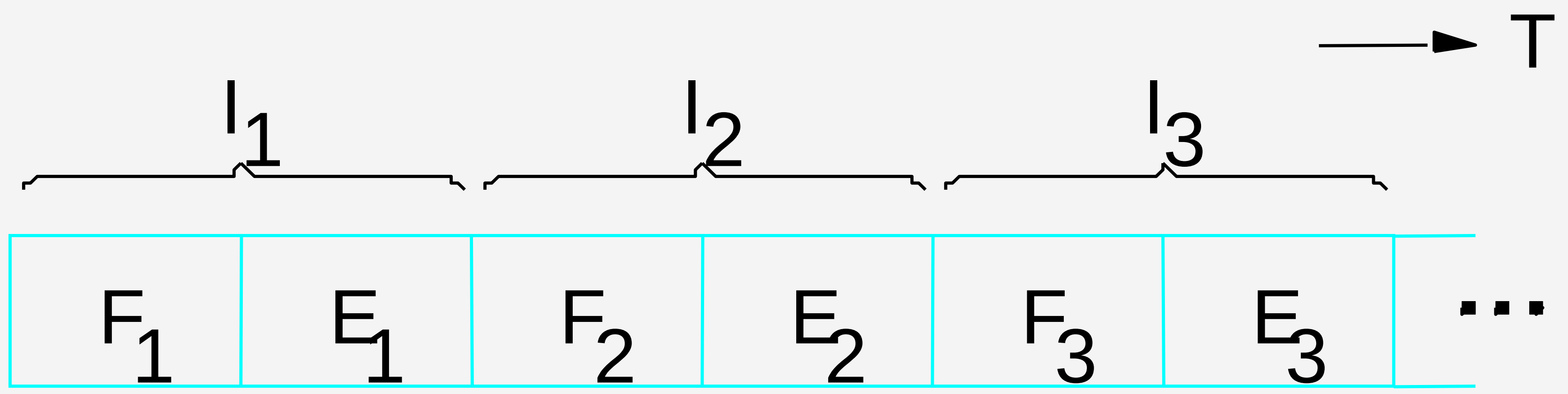
Traditional Pipeline Concept



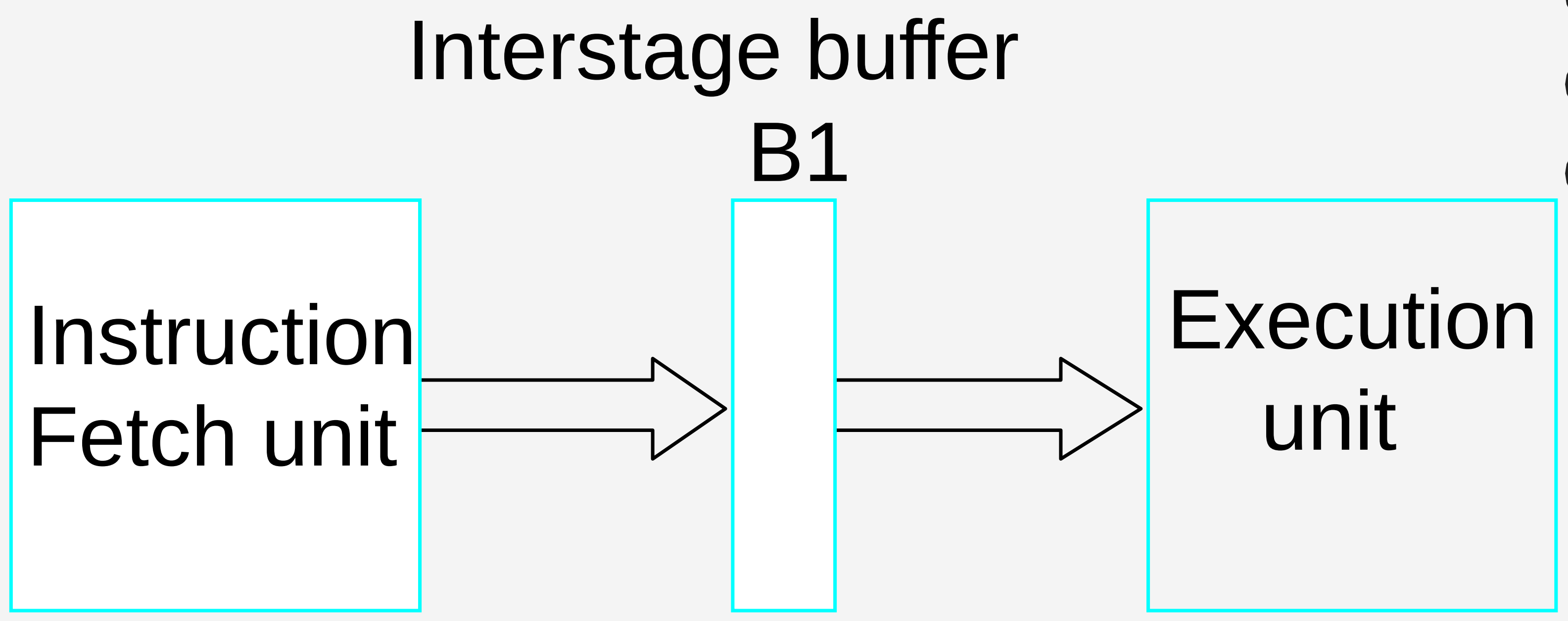
- Pipelining doesn’t help latency of single task, it helps throughput of entire workload
- Pipeline rate limited by slowest pipeline stage
- Multiple tasks operating simultaneously using different resources
- Potential speedup = Number pipe stages
- Unbalanced lengths of pipe stages reduces speedup
- Time to “fill” pipeline and time to “drain” it reduces speedup
- Stall for Dependences

Use the Idea of Pipelining in a Computer

Basic idea of instruction pipelining

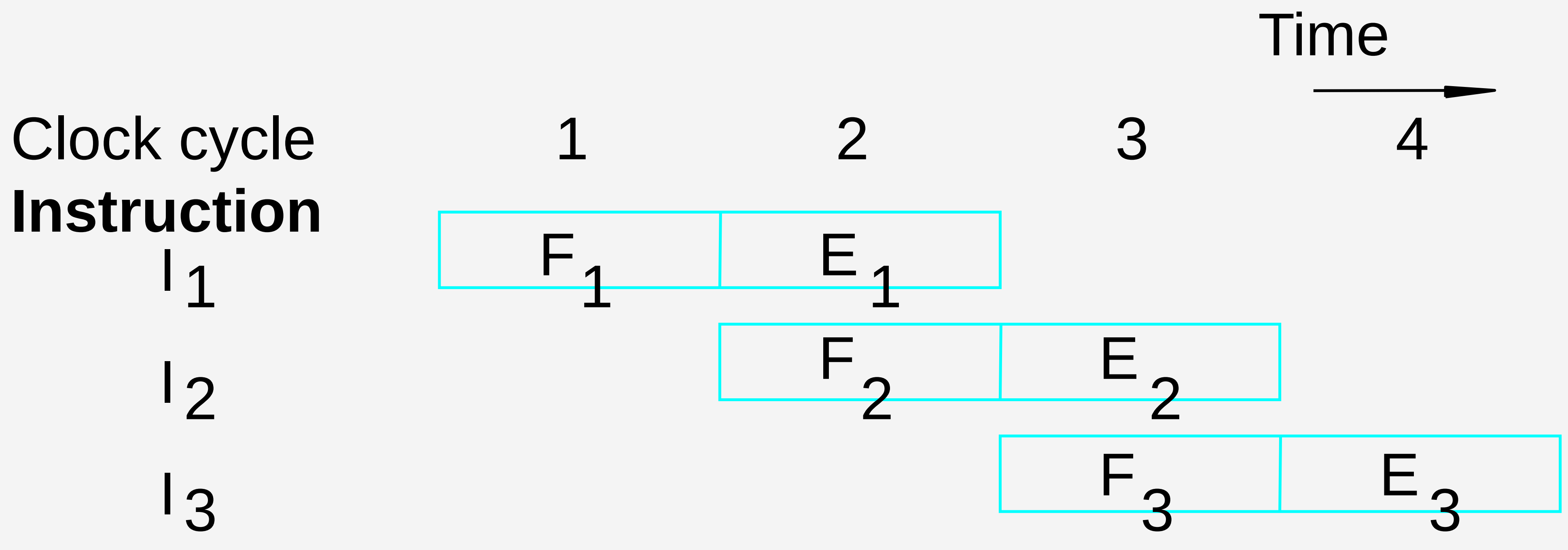


(a) Sequential execution



(b) Hardware organization

Fetch + Execution



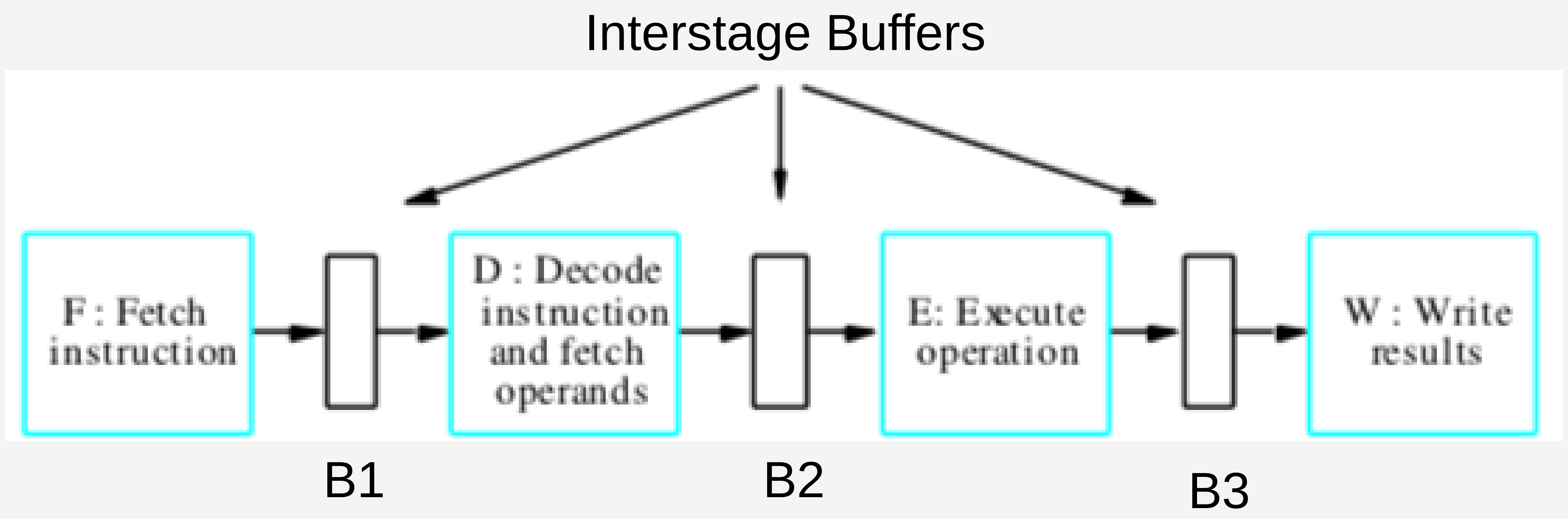
(c) Pipelined execution

4-stage Pipeline

Clock Cycle :	1	2	3	4	5	6	7
	F1	D1	E1	W1			
		F2	D2	E2	W2		
			F3	D3	E3	W3	
				F4	D4	E4	W4

Fetch + Decode
+ Execution + Write

(a) Instruction execution divided into four steps



(b) Hardware organization

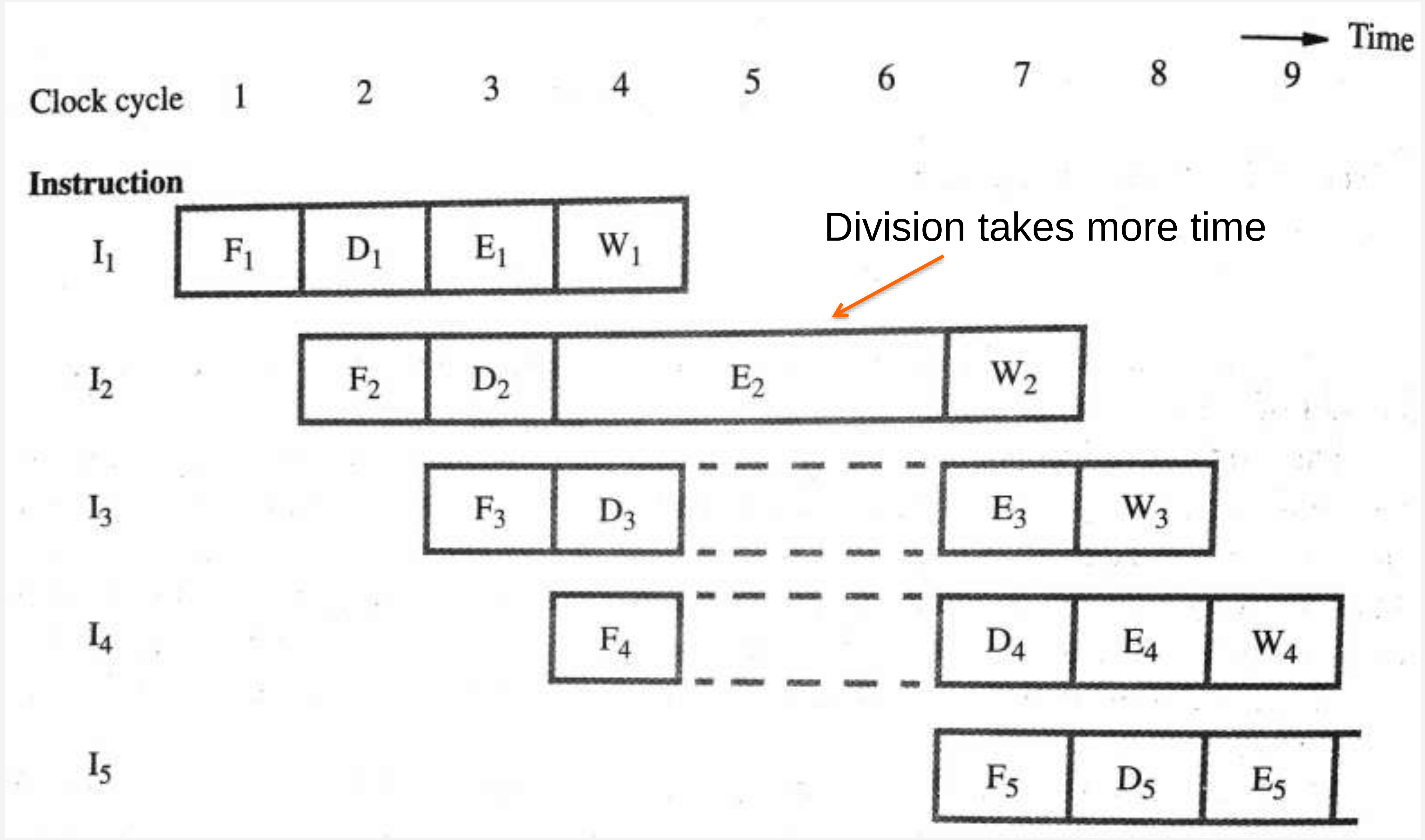
Role of Cache Memory

- Each pipeline stage is expected to complete in one clock cycle.
- The clock period should be long enough to let the slowest pipeline stage to complete.
- Faster stages can only wait for the slowest one to complete.
- Since main memory is very slow compared to the execution, if each instruction needs to be fetched from main memory, pipeline is almost useless.
- Fortunately, we have cache.

Pipeline Performance

- The potential increase in performance resulting from pipelining is proportional to the number of pipeline stages.
- However, this increase would be achieved only if all pipeline stages require the same time to complete, and there is no interruption throughout program execution.
- Unfortunately, this is not true.

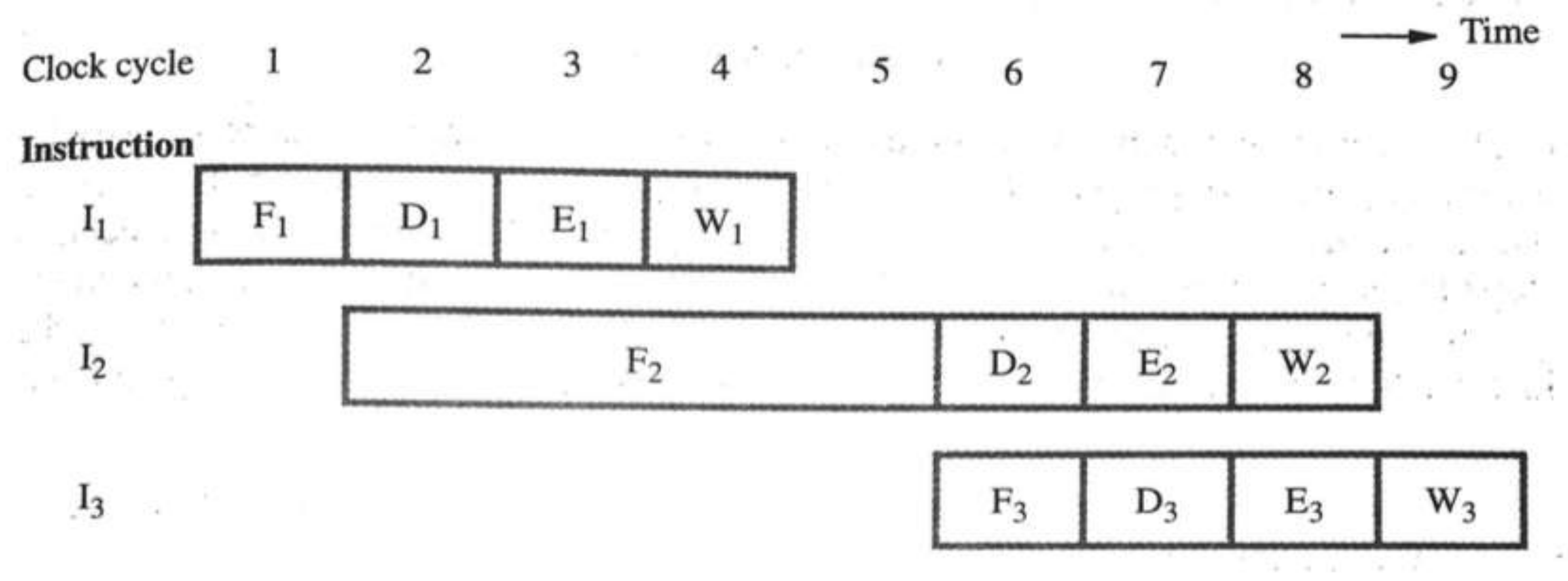
Effect of an execution operation taking more than one clock cycle



Pipeline Performance

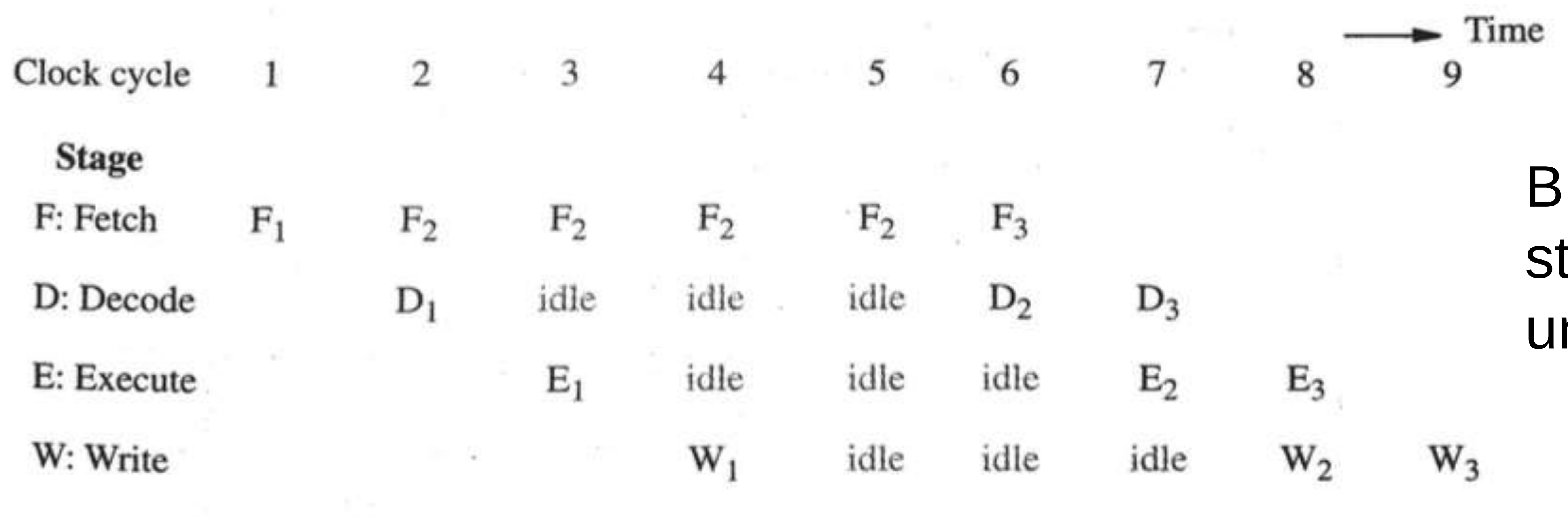
- The previous pipeline is said to have been **stalled** for two clock cycles.
- Any condition that causes a pipeline to stall is called a **hazard.**
- **Data hazard** – any condition in which either the source or the destination operands of an instruction are not available at the time expected in the pipeline.
- So some operation has to be delayed, and the pipeline stalls.
- **Instruction (control) hazard** – a delay in the availability of an instruction causes the pipeline to stall.
- Example: Cache miss on pipeline operation
    - **Structural hazard** – the situation when two instructions require the use of a given hardware resource at the same time.

Pipeline stall by cache miss in F2



Time
Clock cycle 1 2 3 4 5 6 7 8 9
Instruction
I_1: F_1 D_1 E_1 W_1
I_2: F_2 D_2 E_2 W_2
I_3: F_3 D_3 E_3 W_3

(a) Instruction execution steps in successive clock cycles

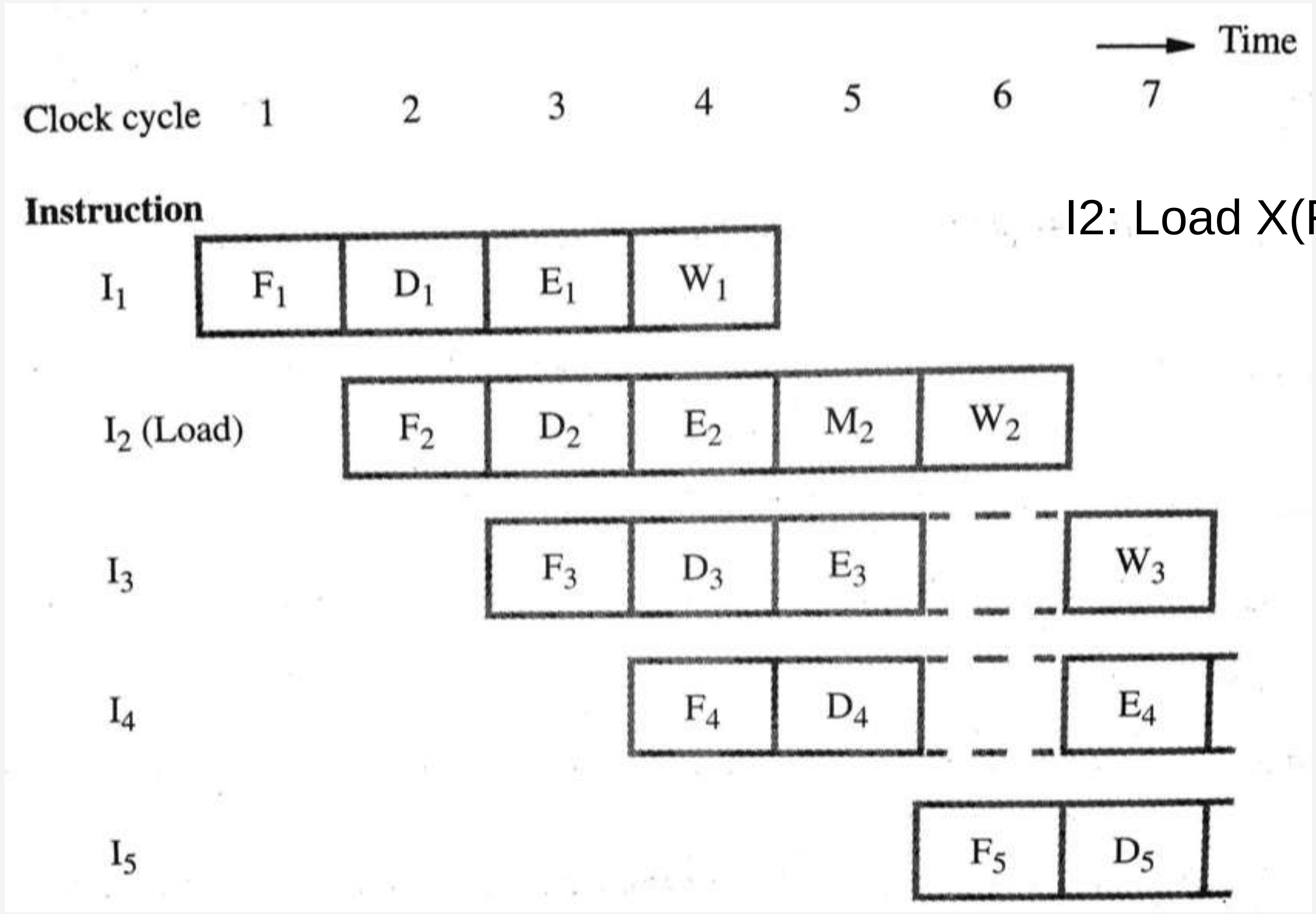


Clock cycle	1	2	3	4	5	6	7	8	9
Stage									
F: Fetch	F_1	F_2	F_2	F_2	F_2	F_3			
D: Decode		D_1	idle	idle	idle	D_2	D_3		
E: Execute			E_1	idle	idle	idle	E_2	E_3	
W: Write				W_1	idle	idle	idle	W_2	W_3

(b) Function performed by each processor stage in successive clock cycles

Bubble(idle): A delay in one stages bubbles down until last unit

Structural Hazard



I2: Load X(R1), R2

Pipeline performance

- Again, pipelining does not result in individual instructions being executed faster; rather, it is the throughput that increases.
- Throughput is measured by the rate at which instruction execution is completed.
- Pipeline stall causes degradation in pipeline performance.
- We need to identify all hazards that may cause the pipeline to stall and to find ways to minimize their impact.

Quiz

Four instructions, the I3 takes two clock cycles for execution. Draw the figure for 4-stage pipeline, and figure out the total cycles needed for the four instructions to complete.

Thank You