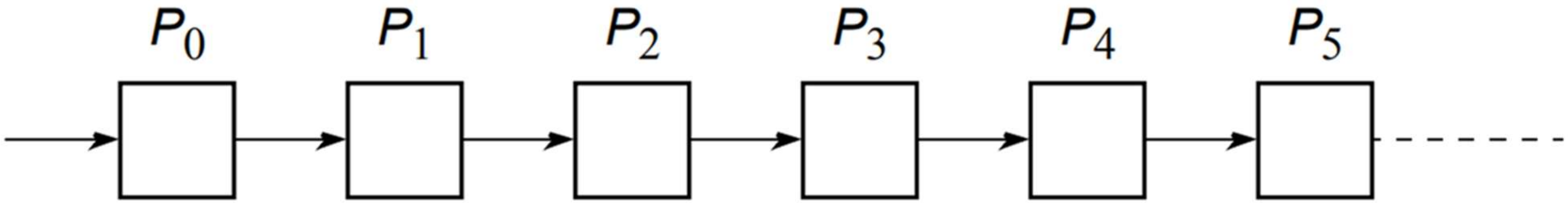
4 Pipelined Computations

[Weightage(15%): Approx. 10-11 Marks out of 70 Marks]

- Pipeline Technique [Chapter 5, Topic 5.1, Page 140]
- Computing Platform for Pipelined Applications [Chapter 5, Topic 5.2, Page 144]
- Pipeline Program Examples [Chapter 5, Topic 5.3, Page 145]

4.1 Pipeline Technique

Pipelined Computations: Problem divided into a series of tasks that have to be completed one after the other (the basis of sequential programming). Each task executed by a separate process or processor.

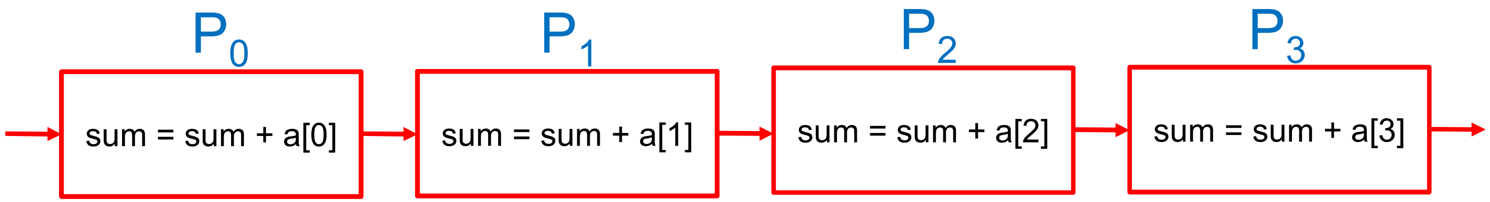


Example:

```
for (i=0; i<n; ++i)
         {
         sum = sum + a[i]
         }
```

sum = sum + a[0]
sum = sum + a[1]
sum = sum + a[2]
sum = sum + a[3]

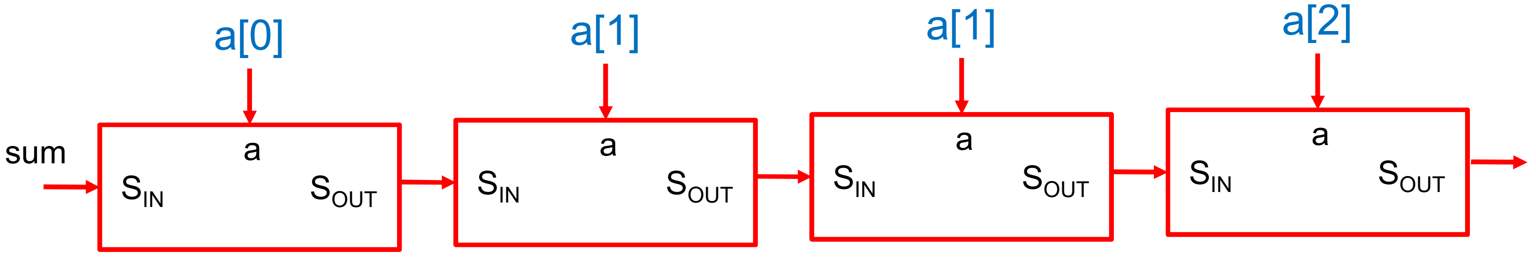
$S_{OUT} = S_{IN} + a[i]$



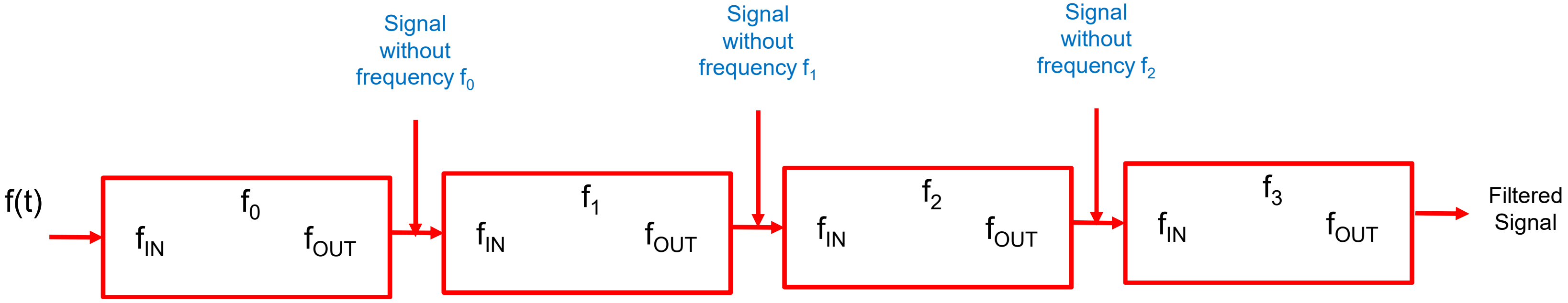
4.1 Pipeline Technique

Stage i

$S_{OUT} = S_{IN} + a[i]$



Pipeline for an unrolled loop



Pipeline for a Frequency Filter

Pipelining in Microprocessor Instruction Execution

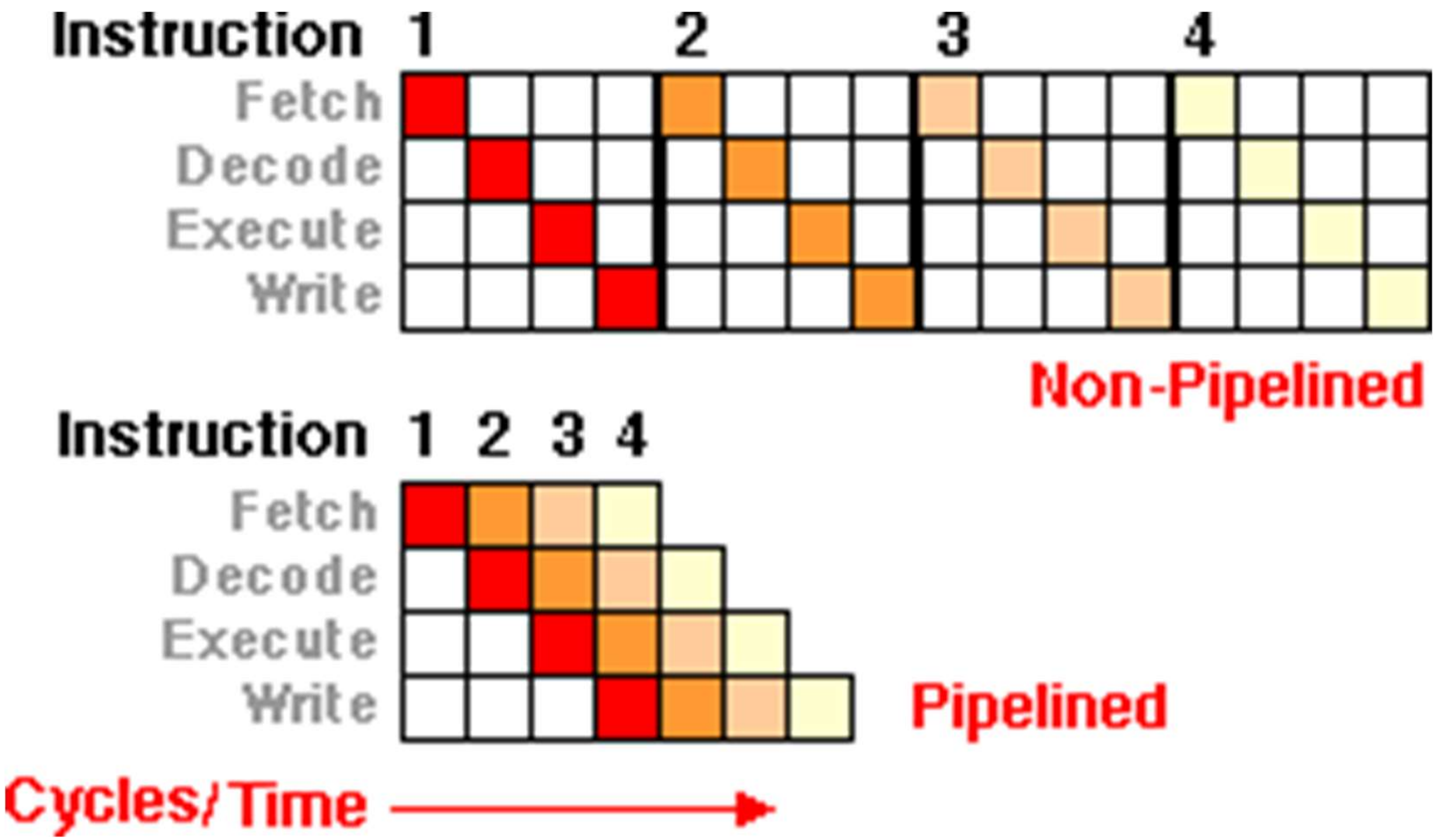
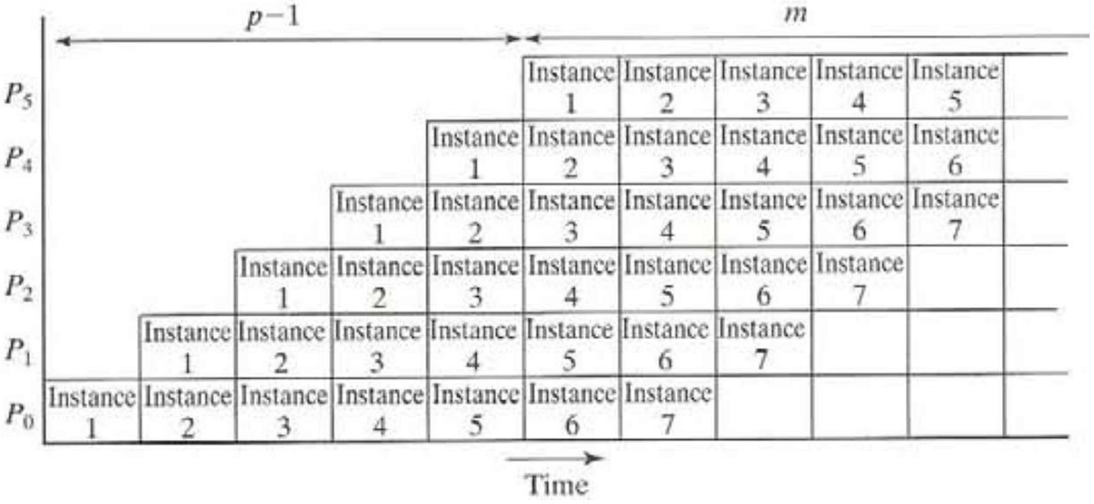


Image Source: https://www.mackido.com/Hardware/Pipelines.html

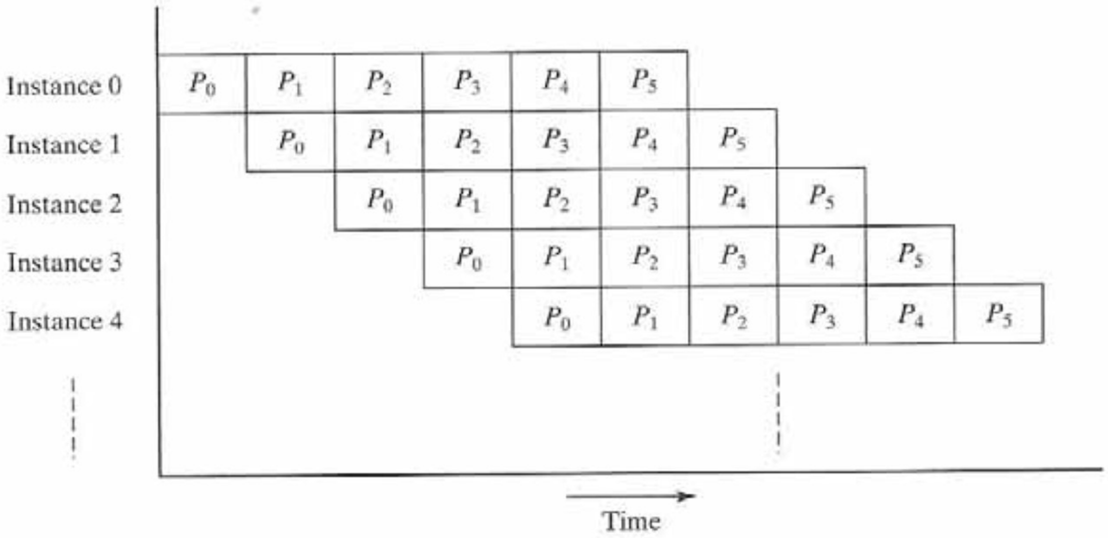
Space-time diagram for a pipeline



P_5						Instance 1	Instance 2	Instance 3	Instance 4	Instance 5	
P_4					Instance 1	Instance 2	Instance 3	Instance 4	Instance 5	Instance 6	
P_3				Instance 1	Instance 2	Instance 3	Instance 4	Instance 5	Instance 6	Instance 7	
P_2			Instance 1	Instance 2	Instance 3	Instance 4	Instance 5	Instance 6	Instance 7		
P_1		Instance 1	Instance 2	Instance 3	Instance 4	Instance 5	Instance 6	Instance 7			
P_0	Instance 1	Instance 2	Instance 3	Instance 4	Instance 5	Instance 6	Instance 7				

Figure 5.4 Space-time diagram of a pipeline.

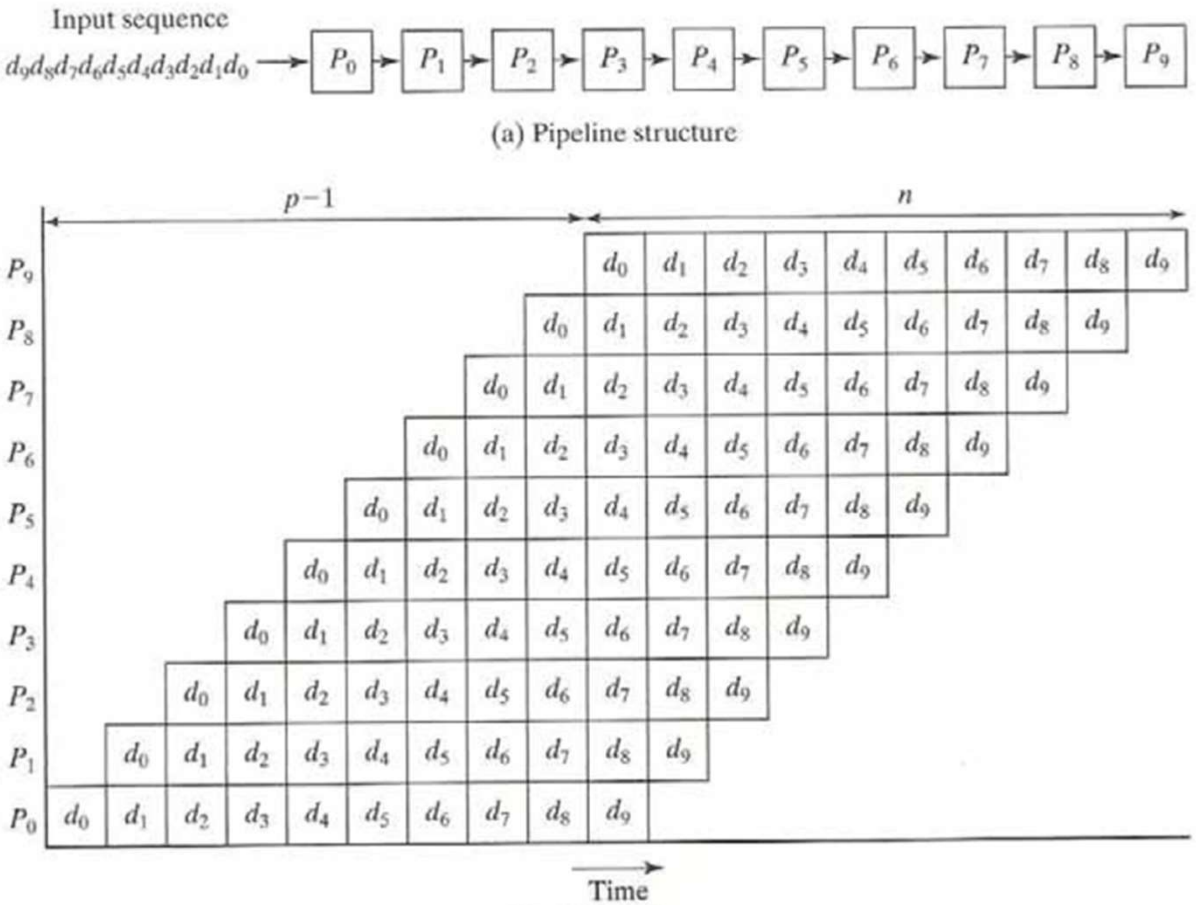
Alternative Space-time diagram



Instance 0	P_0	P_1	P_2	P_3	P_4	P_5				
Instance 1		P_0	P_1	P_2	P_3	P_4	P_5			
Instance 2			P_0	P_1	P_2	P_3	P_4	P_5		
Instance 3				P_0	P_1	P_2	P_3	P_4	P_5	
Instance 4					P_0	P_1	P_2	P_3	P_4	P_5

Figure 5.5 Alternative space-time diagram.

Pipeline Processing 10 data elements



(a) Pipeline structure

P_9										d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9
P_8									d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	
P_7								d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9		
P_6							d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9			
P_5						d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9				
P_4					d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9					
P_3				d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9						
P_2			d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9							
P_1		d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9								
P_0	d_0	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9									

(b) Timing diagram

Figure 5.6 Pipeline processing ten data elements.

E.g. Multiplying elements of an array where individual elements enter the pipeline as sequential series of numbers.

4.2 Computing Platform for Pipelined Applications

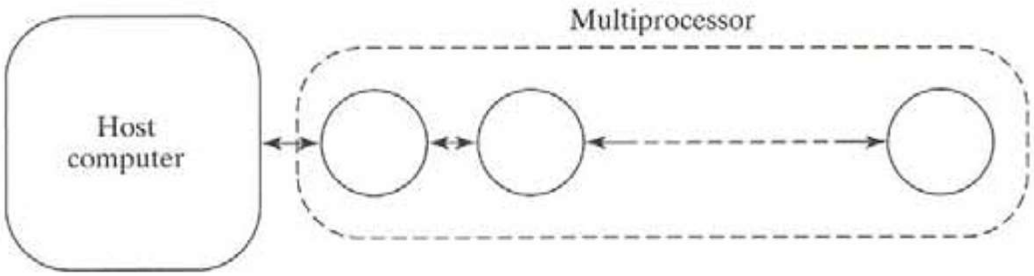


Figure 5.9 Multiprocessor system with a line configuration.

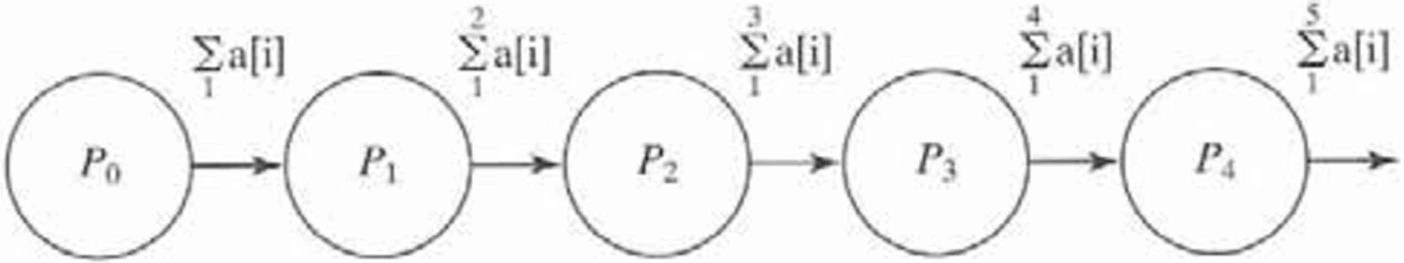


Figure 5.10 Pipelined addition.

recv(&accumulation, P_{i-1});
accumulation = accumulation + number;
send(&accumulation, P_{i+1})

Except for first process, P_0, which is
send (&number, P_1)

4.2 Computing Platform for Pipelined Applications

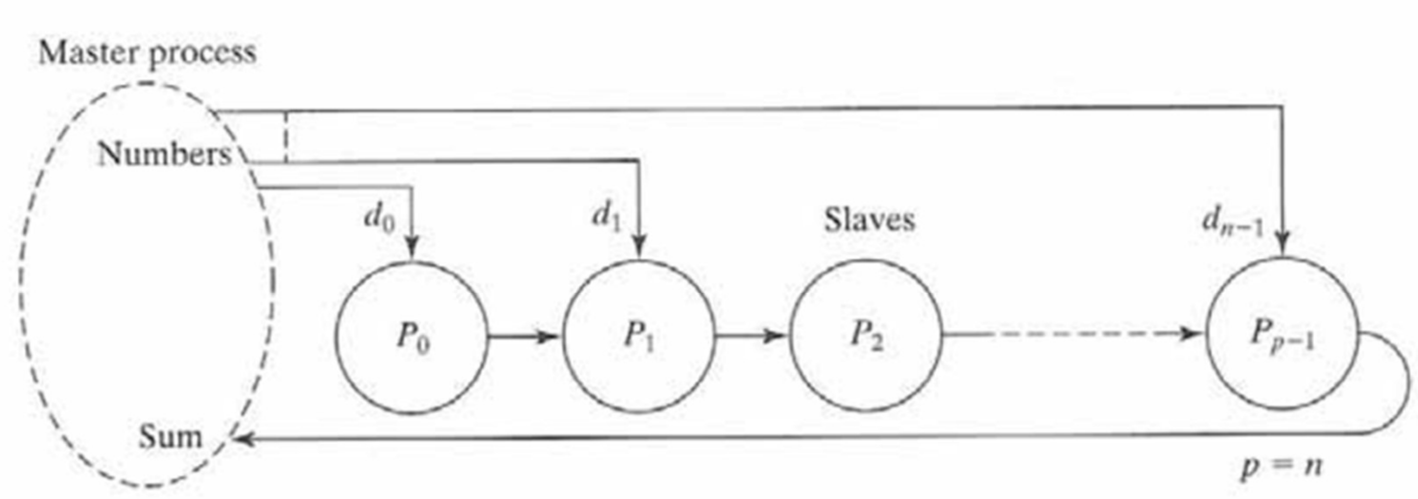


Figure 5.12 Pipelined addition of numbers with direct access to slave processes.

Master/Slave Processes

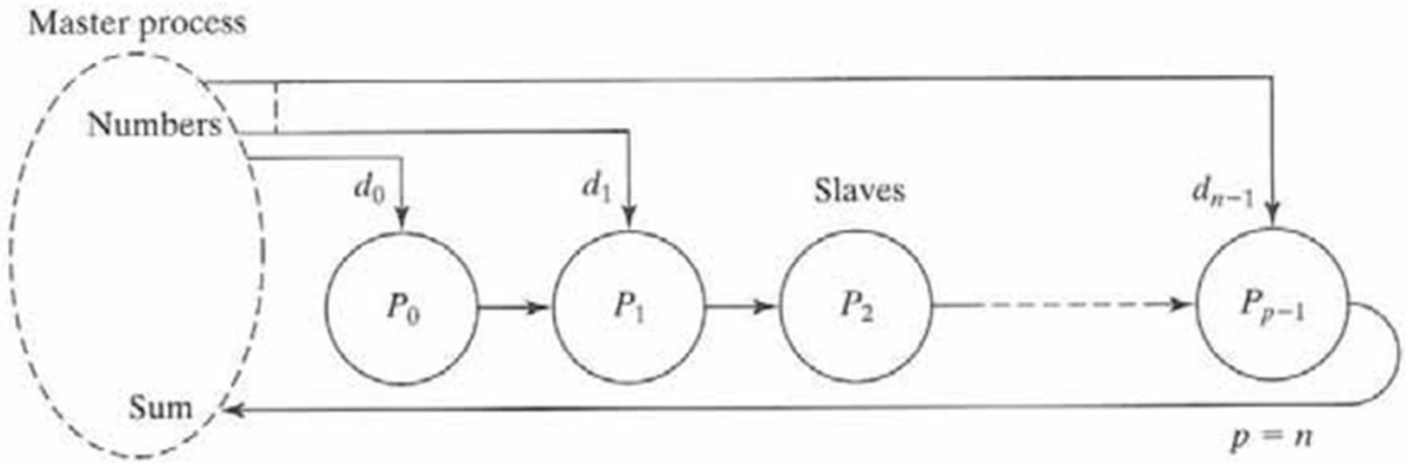


Figure 5.12 Pipelined addition of numbers with direct access to slave processes.

Sorting using Pipelining

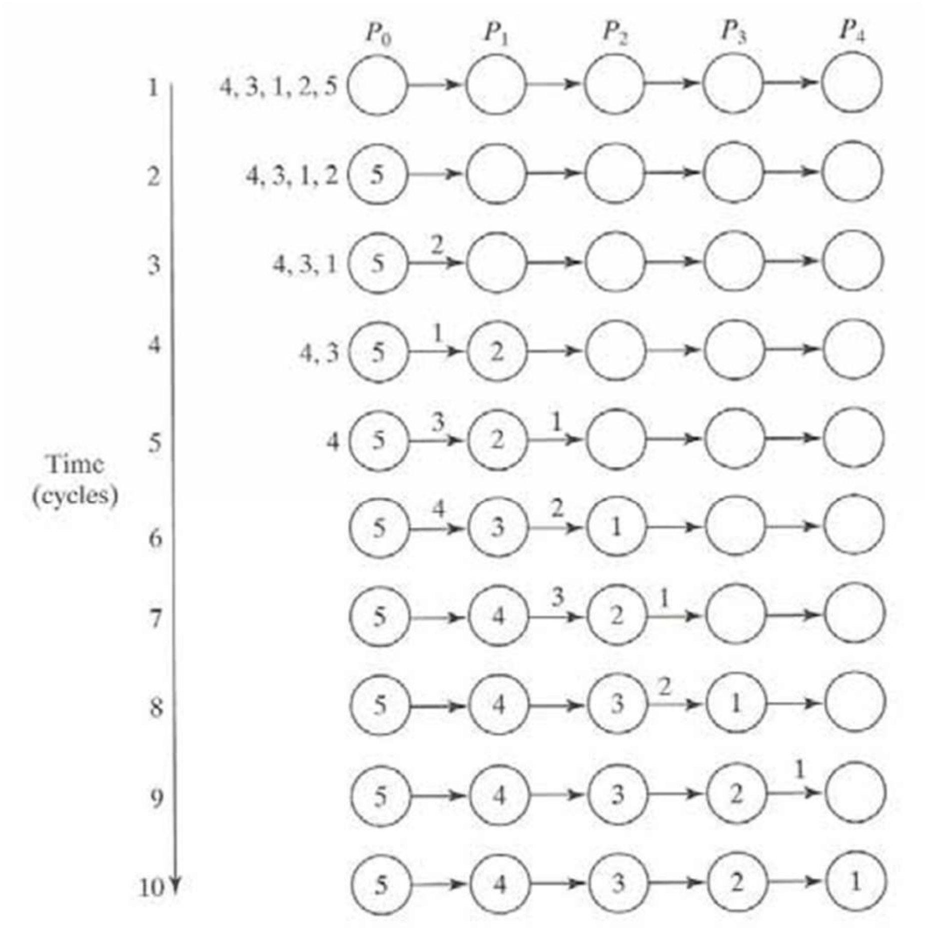


Figure 5.13 Steps in insertion sort with five numbers.

Basic Algorithm for process P_i

```
recv (&number, P_{i-1});
if(number > x)
        {
        send(&x, P_{i-1});
        x = number;
        }
else {send(&number, P_{i+1});}
```

with 'n' numbers:

```
right_procNum = n - i - 1;
recv(&x, P_{i-1});
for(j=0; j<right_procNum; j++)
        {
        recv(&number, P_{i-1});
        if (&number > x)
                {
                send(&x, P_{i+1});
                x = number;
                }
        else
                {send(&number, P_{i+1})}
        }
```

4 Pipelined Computations

[Weightage(15%): Approx. 10-11 Marks out of 70 Marks]

1. Explain pipeline processing with time-space diagram [5, April 2022]
2. Explain (a) Pipeline for an unrolled loop OR (b) Pipeline for a Frequency Filter with neat diagram.
3. Explain the usage of Pipeline with space-time diagram.
4. Explain with neat diagram, how 10 elements of an array are multiplied using pipelining where individual elements enter the pipeline as sequential series of numbers.
5. Explain with diagram, how multiprocessor system can be used to add ‘n’ numbers.
6. Explain sorting with pipelining. (Take any 5 sample numbers of your choice to demonstrate the process)