Pipelining

Pipelining

- A technique of decomposing a sequential process into sub operations, with each sub process being executed in a special dedicated segment that operates concurrently with all other segments.
- Each segment performs partial processing in which the task is partitioned.
- The result obtained in each segment is transferred to the next segment in the pipeline.
- The final result is obtained after data have passed through all segments.

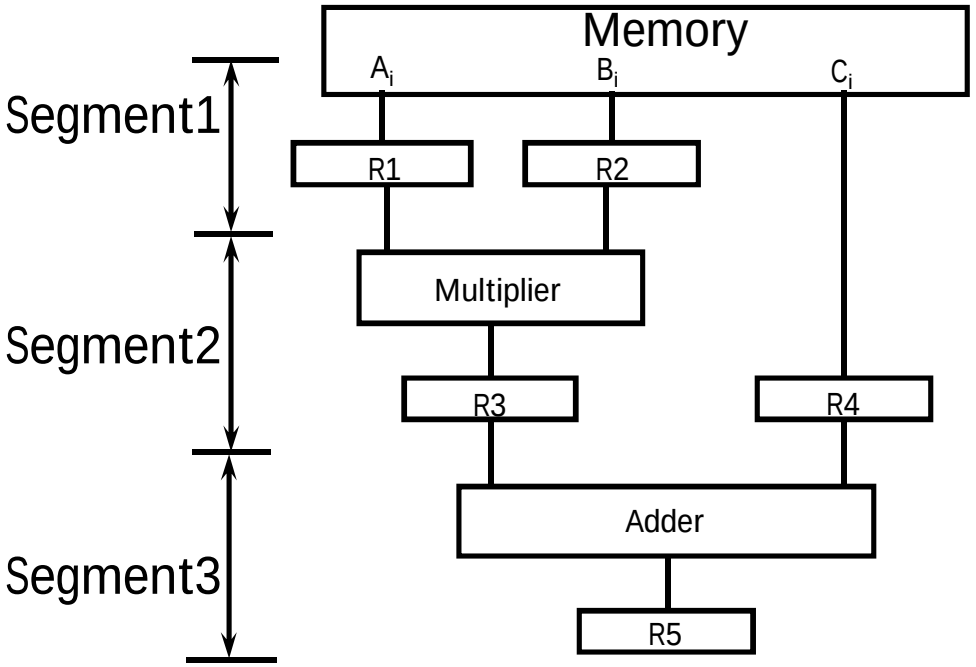
Pipelining

- The simplest way to implement pipeline is to imagine that each segment consist of input register followed by combinational circuit.
- The register holds the data and combinational circuit performs the sub operation in the particular segment.
- The output of the combinational circuit in a given segment is applied to the input register of the next segment.
- In this way the information flows through the pipeline one step at a time.

Pipelining

- Ex: Suppose that we want to perform the combined multiply and add operation with stream of numbers

$$A_i * B_i + C_i \quad \text{for } i = 1, 2, 3, \ldots, 7$$



Pipelining

- The register R1 to R5 that receive new data with every clock pulse
- The multiplier & adder are the combinational circuit
- The sub operation performed in each segment of the pipeline are as follows:

$R1 \leftarrow A_i$, $R2 \leftarrow B_i$	Load A_i and B_i
$R3 \leftarrow R1 * R2$, $R4 \leftarrow C_i$	Multiply and load C_i
$R5 \leftarrow R3 + R4$	Add

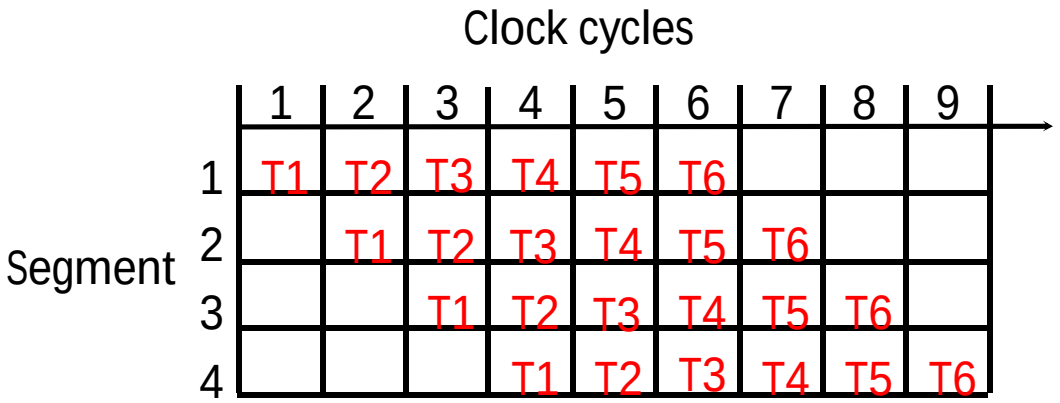
OPERATIONS IN EACH PIPELINE STAGE

Clock Pulse Number	Segment 1		Segment 2		Segment 3
	R1	R2	R3	R4	R5
1	A1	B1	---	---	-------
2	A2	B2	A1 * B1	C1	-------
3	A3	B3	A2 * B2	C2	A1 * B1 + C1
4	A4	B4	A3 * B3	C3	A2 * B2 + C2
5	A5	B5	A4 * B4	C4	A3 * B3 + C3
6	A6	B6	A5 * B5	C5	A4 * B4 + C4
7	A7	B7	A6 * B6	C6	A5 * B5 + C5
8			A7 * B7	C7	A6 * B6 + C6
9					A7 * B7 + C7

General Consideration

- The segments are separated by register Ri that holds the intermediate results between the stages.
- The task is defined as the total operation performed by going through all the segments in the pipeline.
- The behavior of pipeline can be illustrated with a space time diagram that shows the segment utilization as a function of time.

General Consideration



- The above diagram shows 6 tasks T1 through T6 executed in 4 segments.
- Initially Task T1 is handled by segment 1.After the first clock cycle segment 2 is busy with T1 while the segment 1 is busy with T2, Continuing this manner the first task T1 is completed after fourth clock cycle.

PIPELINE SPEEDUP

– Consider the case where a k-segment pipeline used to execute n tasks.
  - n = 6 in previous example
  - k = 4 in previous example

- Pipelined Machine (k stages, n tasks)
  - The first task t1 requires k clock cycles to complete its operation since there are k segments
  - The remaining n-1 tasks require n-1 clock cycles
  - The n tasks clock cycles = k+(n-1) (9 in previous example)
- Conventional Machine (Non-Pipelined)
  - Cycles to complete each task in nonpipeline = k
  - For n tasks, nk cycles is required.
- Speedup (S)
  - S = Nonpipeline time /Pipeline time
    - For n tasks: S = nk/(k+n-1)

Example

- 4-stage pipeline
    - 100 tasks to be executed
    - 1 task in non-pipelined system; 4 clock cycles

Pipelined System : k + n - 1 = 4 + 99 = 103 clock cycles

Non-Pipelined System : n*k = 100 * 4 = 400 clock cycles

Speedup : S = 400 / 103 = 3.88