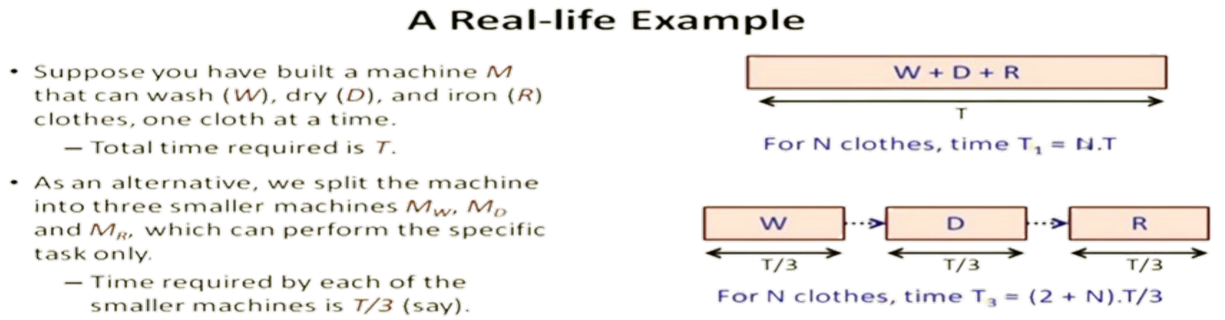
Experiment-7

Aim:- Design and implementation of a pipeline architecture in verilog.

Prerequisites:- A fair understanding of pipeline concept and verilog language.

Theory:- Pipelining is a technique used in modern processors to improve performance by executing multiple instructions simultaneously. It breaks down the execution of instructions into several stages, where each stage completes a part of the instruction. These stages can overlap, allowing the processor to work on different instructions at various stages of completion, similar to an assembly line in manufacturing.

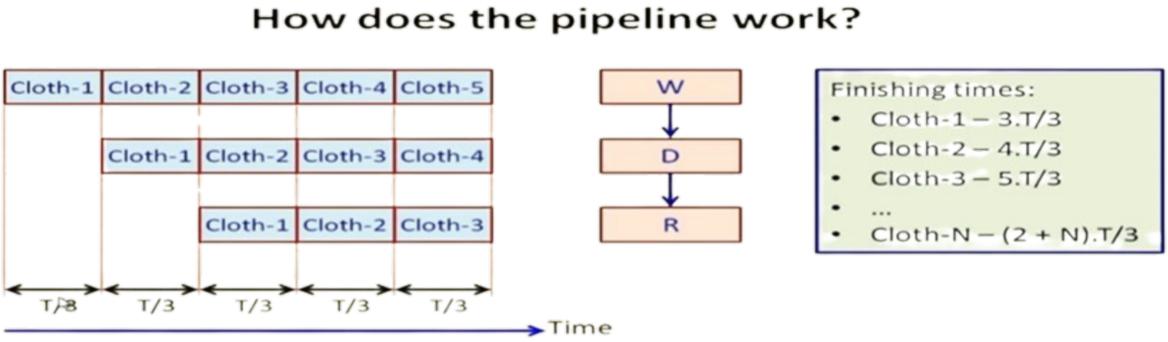


A Real-life Example

- Suppose you have built a machine *M* that can wash (*W*), dry (*D*), and iron (*R*) clothes, one cloth at a time.
  - Total time required is *T*.
- As an alternative, we split the machine into three smaller machines M_W, M_D and M_R, which can perform the specific task only.
  - Time required by each of the smaller machines is *T/3* (say).

For N clothes, time T_1 = N.T

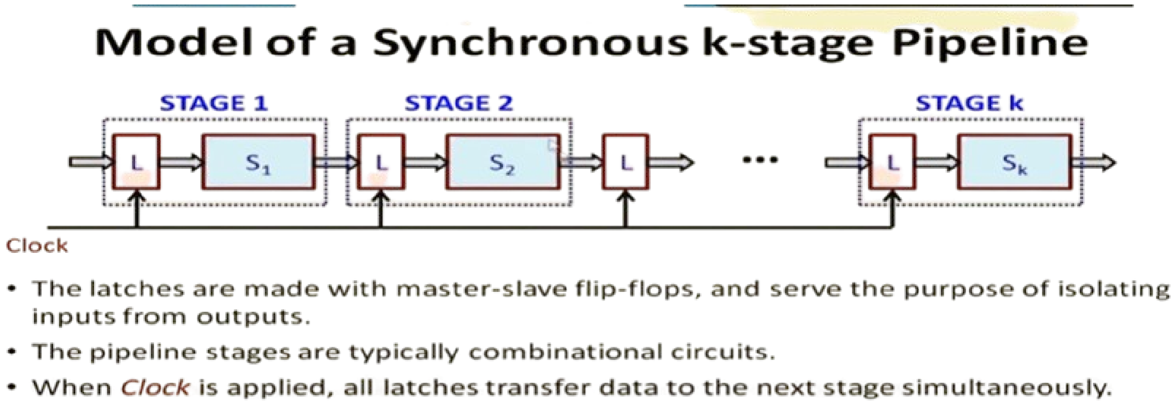
For N clothes, time T_3 = (2 + N).T/3



How does the pipeline work?

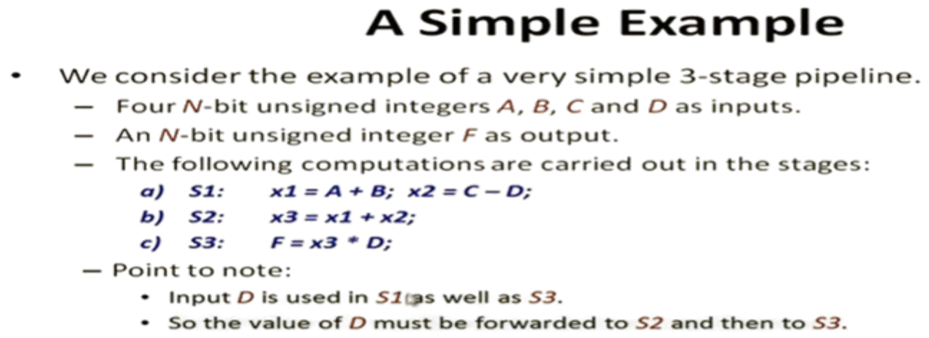
Cloth-1	Cloth-2	Cloth-3	Cloth-4	Cloth-5
	Cloth-1	Cloth-2	Cloth-3	Cloth-4
		Cloth-1	Cloth-2	Cloth-3
T/3	T/3	T/3	T/3	T/3

Finishing times:
- Cloth-1 – 3.T/3
- Cloth-2 – 4.T/3
- Cloth-3 – 5.T/3
- …
- Cloth-N – (2 + N).T/3



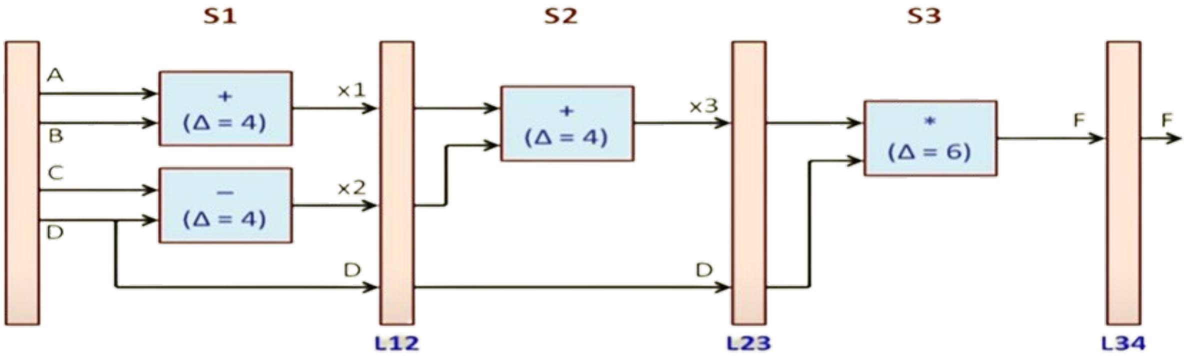
Model of a Synchronous k-stage Pipeline

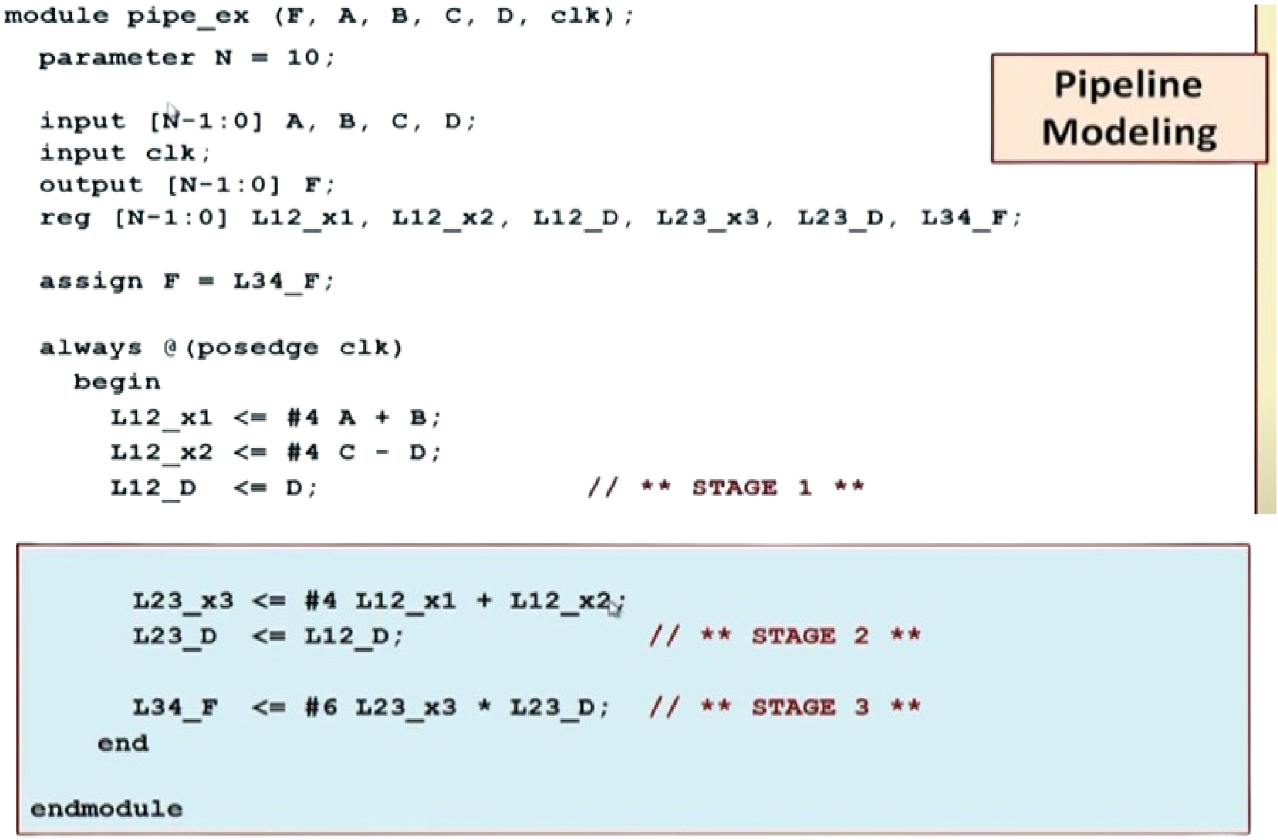
- The latches are made with master-slave flip-flops, and serve the purpose of isolating inputs from outputs.
- The pipeline stages are typically combinational circuits.
- When *Clock* is applied, all latches transfer data to the next stage simultaneously.



A Simple Example

- We consider the example of a very simple 3-stage pipeline.
  - Four *N*-bit unsigned integers *A*, *B*, *C* and *D* as inputs.
  - An *N*-bit unsigned integer *F* as output.
  - The following computations are carried out in the stages:
    - ***a) S1: x1 = A + B; x2 = C – D;***
    - ***b) S2: x3 = x1 + x2;***
    - ***c) S3: F = x3 * D;***
  - Point to note:
    - Input *D* is used in *S1* as well as *S3*.
    - So the value of *D* must be forwarded to *S2* and then to *S3*.





Pipeline Modeling

```
module pipe_ex (F, A, B, C, D, clk);
  parameter N = 10;

  input [N-1:0] A, B, C, D;
  input clk;
  output [N-1:0] F;
  reg [N-1:0] L12_x1, L12_x2, L12_D, L23_x3, L23_D, L34_F;

  assign F = L34_F;

  always @(posedge clk)
    begin
      L12_x1 <= #4 A + B;
      L12_x2 <= #4 C - D;
      L12_D  <= D;                  // ** STAGE 1 **

      L23_x3 <= #4 L12_x1 + L12_x2;
      L23_D  <= L12_D;              // ** STAGE 2 **

      L34_F  <= #6 L23_x3 * L23_D;  // ** STAGE 3 **
    end

endmodule
```