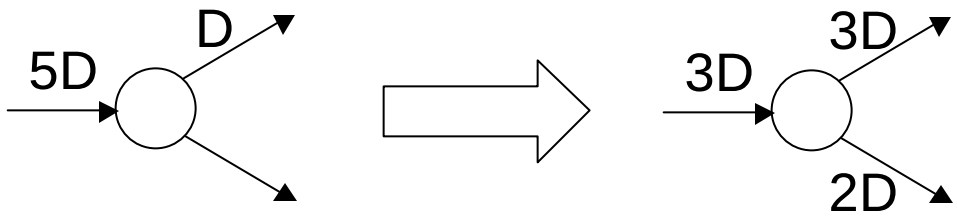
Chapter 4: Retiming

Keshab K. Parhi

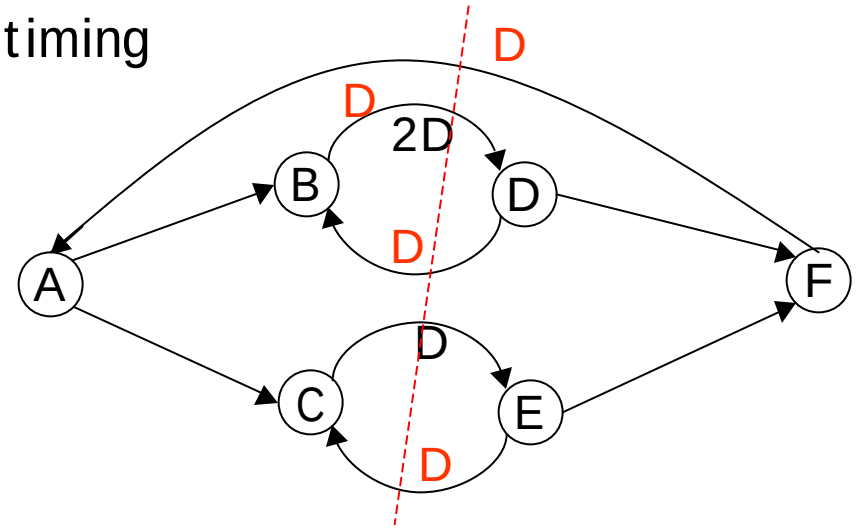
Retiming :

Moving around existing delays

- Does not alter the latency of the system
- Reduces the critical path of the system
- Node Retiming



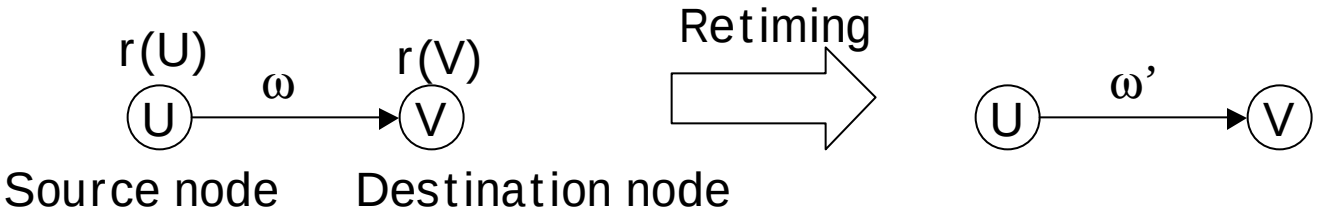
•Cutset Retiming



Retiming

- Generalization of Pipelining
- Pipelining is Equivalent to Introducing Many delays at the Input followed by Retiming

- Retiming Formulation



$$\omega' = \omega + r(V) - r(U)$$

- Properties of retiming
  - The weight of the retimed path $p = V_0 \rightarrow V_1 \rightarrow \ldots V_k$ is given by $\omega_r(p) = \omega(p) + r(V_k) - r(V_0)$
  - Retiming does not change the number of delays in a cycle.
  - Retiming does not alter the iteration bound in a DFG as the number of delays in a cycle does not change
  - Adding the constant value *j* to the retiming value of each node does not alter the number of delays in the edges of the retimed graph.
- Retiming is done to meet the following
  - Clock period minimization
  - Register minimization

- Retiming for clock period minimization
  - Feasibility constraint

$$\omega'(U,V) \geq 0 \quad \Rightarrow \quad \text{causality of the system}$$

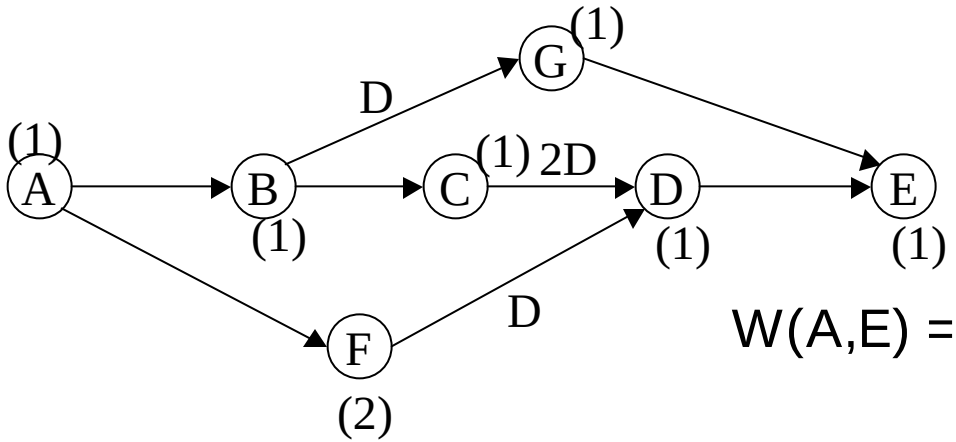
$$\Rightarrow \omega(U,V) \geq r(U) - r(V) \quad \text{(one inequality per edge)}$$

  - Critical Path constraint

$r(U) - r(V) \leq W(U,V) - 1$ for all vertices U and V in the graph such that $D(U,V) > c$ where c = target clock period. The two quantities W(U,V) and D(U,V) are given as:

$$W(U,V) = \min\{w(p) : U \rightarrow V\}$$

$$D(U,V) = \max\{t(p) : U \rightarrow V \text{ and } w(p) = W(U,V)\}$$

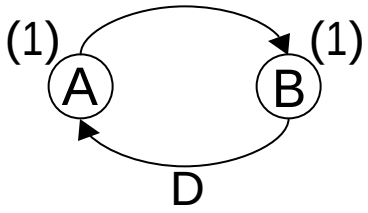


W(A,E) = 1 & D(A,E) = 5

- Algorithm to compute W(U,V) and D(U,V):
  - Let $M = t_{max}n$, where t_{max} is the maximum computation time of the nodes in G and n is the # of nodes in G.
  - Form a new graph G' which is the same as G except the edge weights are replaced by $w'(e) = Mw(e) - t(u)$ for all edges $U \rightarrow V$.
  - Solve for all pair shortest path problem on G' by using Floyd Warshall algorithm. Let S'_{UV} be the shortest path form $U \rightarrow V$.
  - If $U \neq V$, then $W(U,V) = \lceil S'_{UV}/M \rceil$ and $D(U,V) = MW(U,V) - S'_{UV} + t(V)$. If $U = V$, then $W(U,V) = 0$ and $D(U,V) = t(U)$.
- Using W(U,V) and D(U,V) the feasibility and critical path constraints are formulated to give certain inequalities. The inequalities are solved using *constraint graphs* and if a feasible solution is obtained then the circuit can be clocked with a period 'c'.

- Solving a system of inequalities : Given M inequalities in N variables where each inequality is of the form $r_i - r_j \leq k$ for integer values of k.
  - Draw a constraint graph
    - Draw the node i for each of the N variables r_i, I= 1, 2, …, N.
    - Draw the node N+1.
    - For each inequality $r_i - r_j \leq k$, draw the edge j→i of length k.
    - For each node i, i = 1, 2, …, n, draw the edge N+1 →i from the node N+1 to node I with length 0.
  - Solve using a shortest path algorithm.
    - The system of inequalities have a solution iff the constraint graph contains no negative cycles.
    - If a solution exists, one solution is where r_i is the minimum length path from the node N+1 to node i.

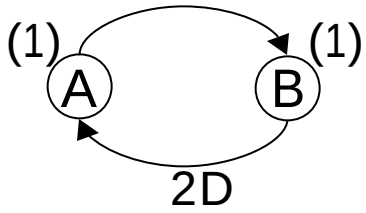
- K-slow transformation
  - Replace each D by kD



Clock	
0	$A0 \rightarrow B0$
1	$A1 \rightarrow B1$
2	$A2 \rightarrow B2$

T_{iter}= 2ut

After 2-slow transformation

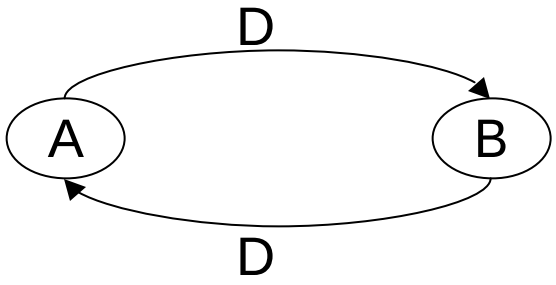


Clock	
0	$A0 \rightarrow B0$
1	
2	$A1 \rightarrow B1$
3	
4	$A2 \rightarrow B2$

T_{clk}= 2ut
T_{iter}= 2×2ut=4ut

*Input new samples every alternate cycles.
*null operations account for odd clock cycles.
*Hardware utilized only 50% time

- Retiming 2-slow graph

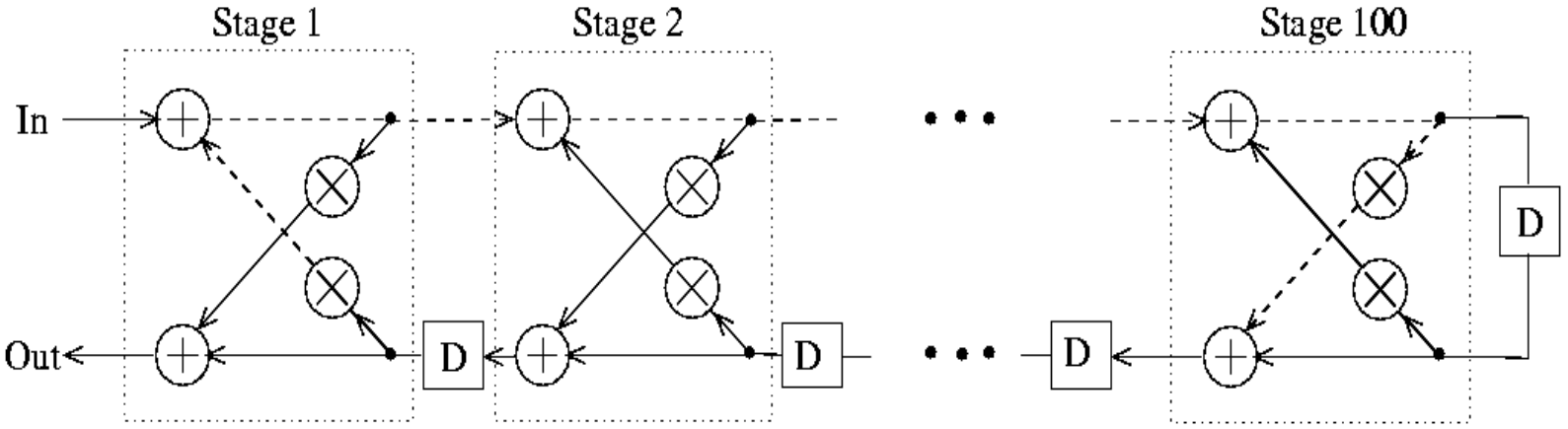


T_{clk} = 1ut
T_{iter} = 2×1=2ut

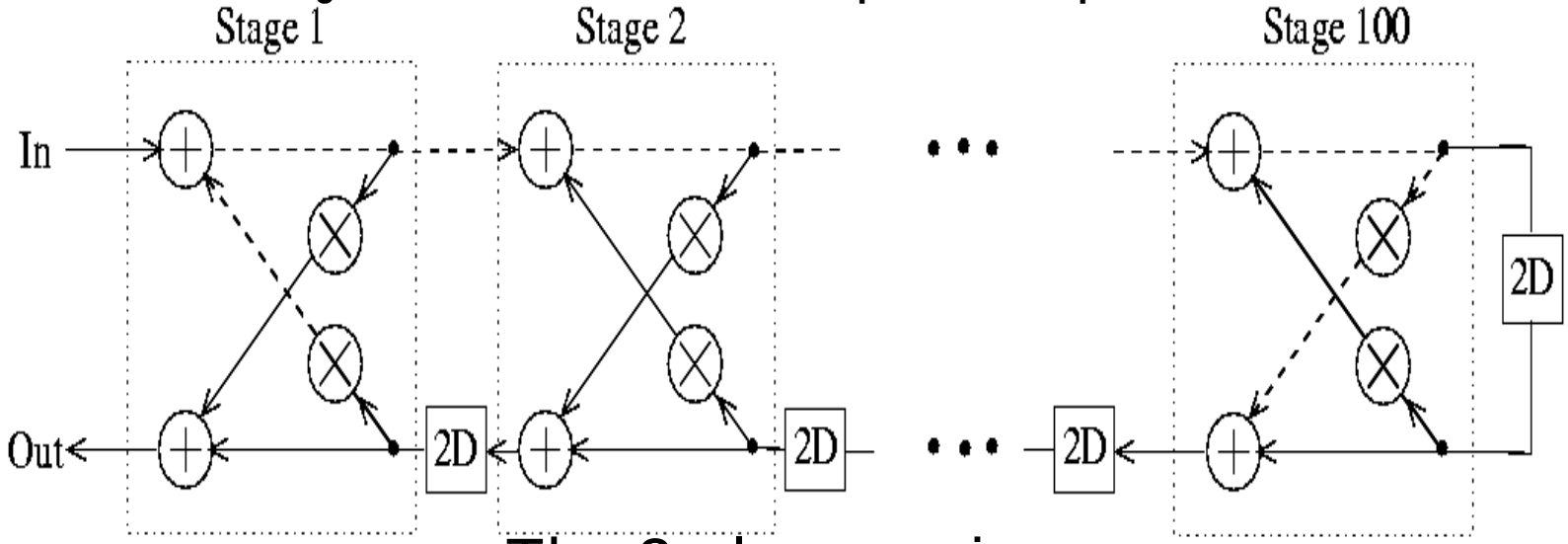
*Hardware Utilization = 50 %

*Hardware can be fully utilized if two independent operations are available.

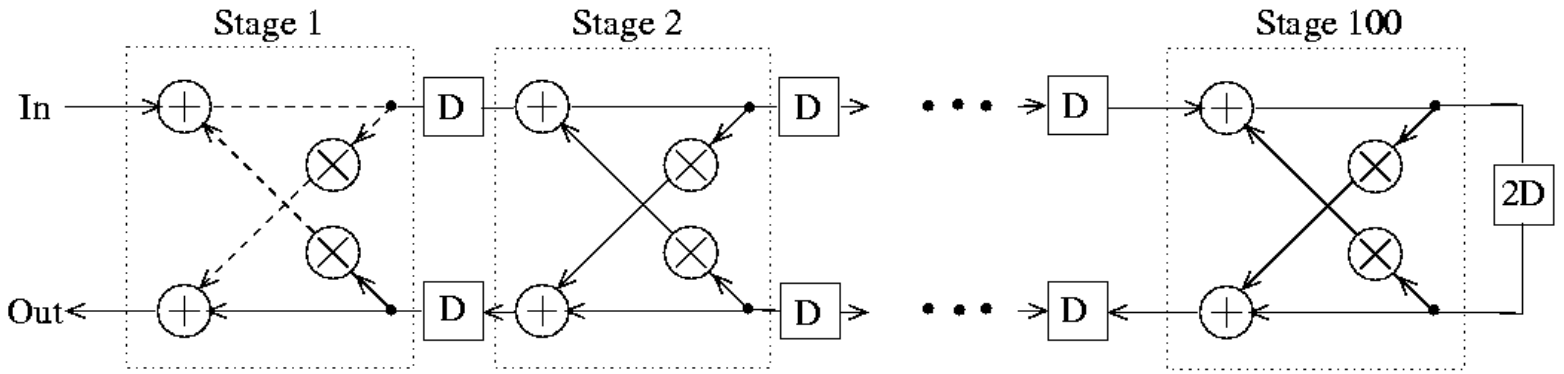
2-Slow Lattice Filter (Fig. 4.7)



A 100 stage Lattice Filter with critical path 2 multiplications and 101 additions



The 2-slow version



A retimed version of the 2 slow circuit with critical path of 2 multiplications and 2 additions

If T_m = 2 u.t. and T_a = 1 u.t., then
T_{clk} = 6 u.t., T_{iter} = 2X6 = 12 u.t.

In Original Lattice Filter, T_{iter} = 105 u.t.
Iteration Period Bound = 7 u.t.

Other Applications of Retiming

- Retiming for Register Minimization (Section 4.4.3)
- Retiming for Folding (Chapter 6)
- Retiming for Power Reduction (Chap. 17)
- Retiming for Logic Synthesis (Beyond Scope of This Class)
- Multi-Rate/Multi-Dimensional Retiming (Denk/Parhi, Trans. VLSI, Dec. 98, Jun.99)