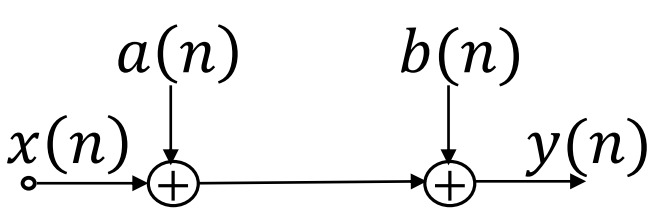
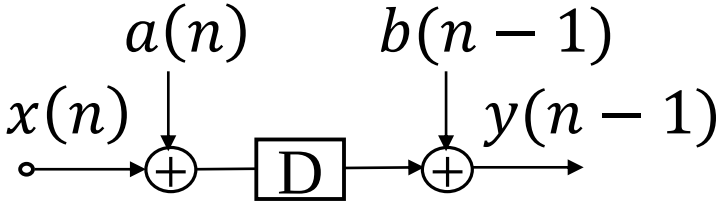
# Pipelining and Parallel Processing

- Pipelining ---reduction in the critical path
  - increase the clock speed, or
  - reduce power consumption at same speed
- Parallel Processing ---multiple outputs are computed in parallel in a clock period
  - effective sampling speed is increased
  - reduction of power consumption

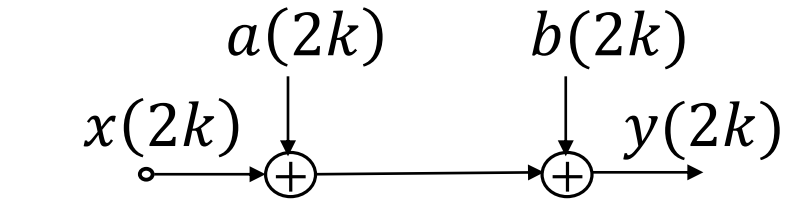
# Pipelining and Parallel Processing (cont'd)

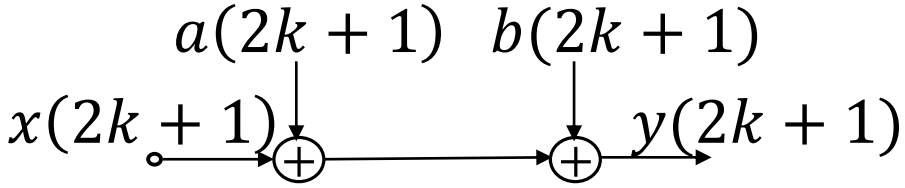



(a) A data path



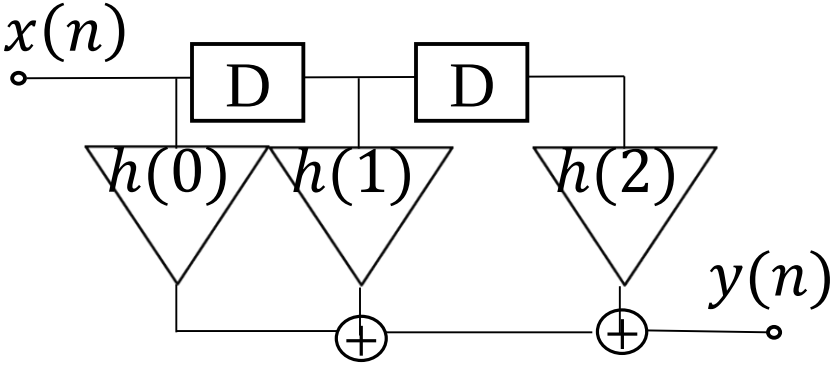

(b) 2-level pipelining




(c) 2-level parallel processing

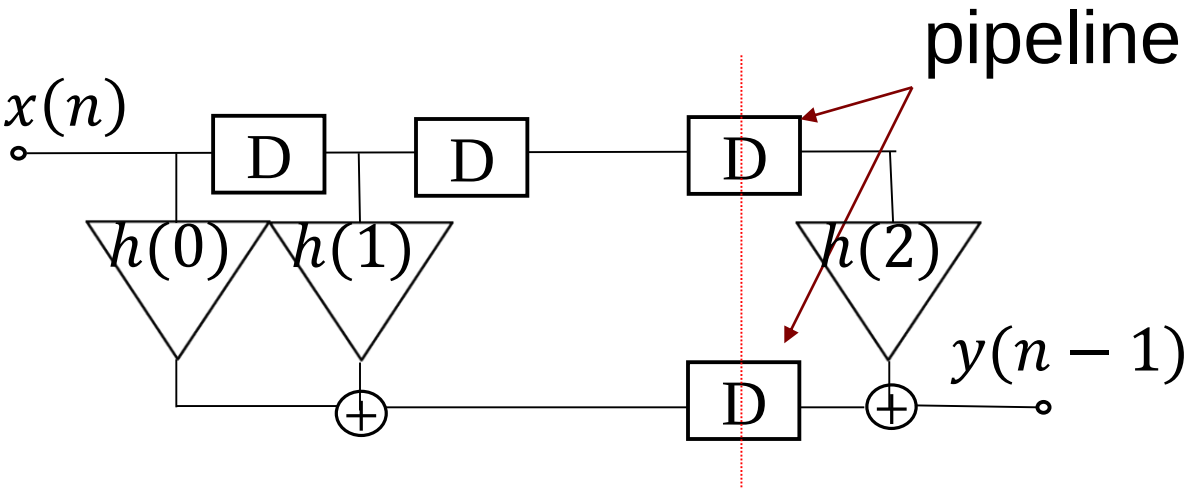
# 3-Tap FIR Filter




$$T_{sample} \geq T_M + 2T_A$$

$$f_{sample} \leq \frac{1}{T_M + 2T_A}$$

# Pipelined FIR Filter




critical path= $T_M + T_A$
increase in latency

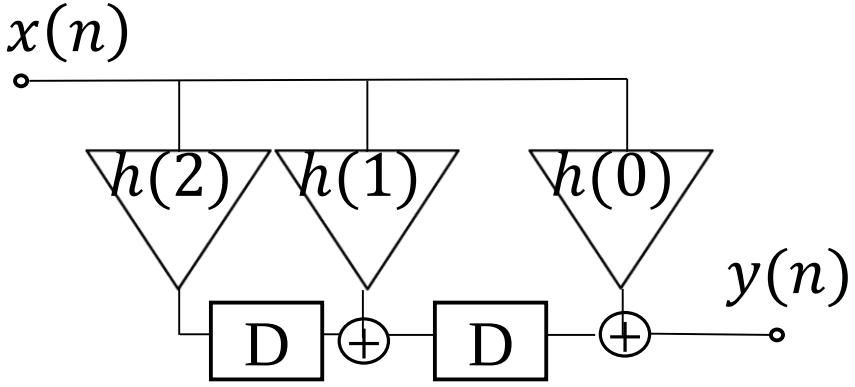
# Feed-forward Cutset

Pipeline latches can only be placed across any feed-forward cutset of the graph

- Cutset: a set of edges of a graph such that if these edges are removed, the graph becomes disjoint.
- Feed-forward Cutset: Data move in the forward direction on all edges of the cutset.

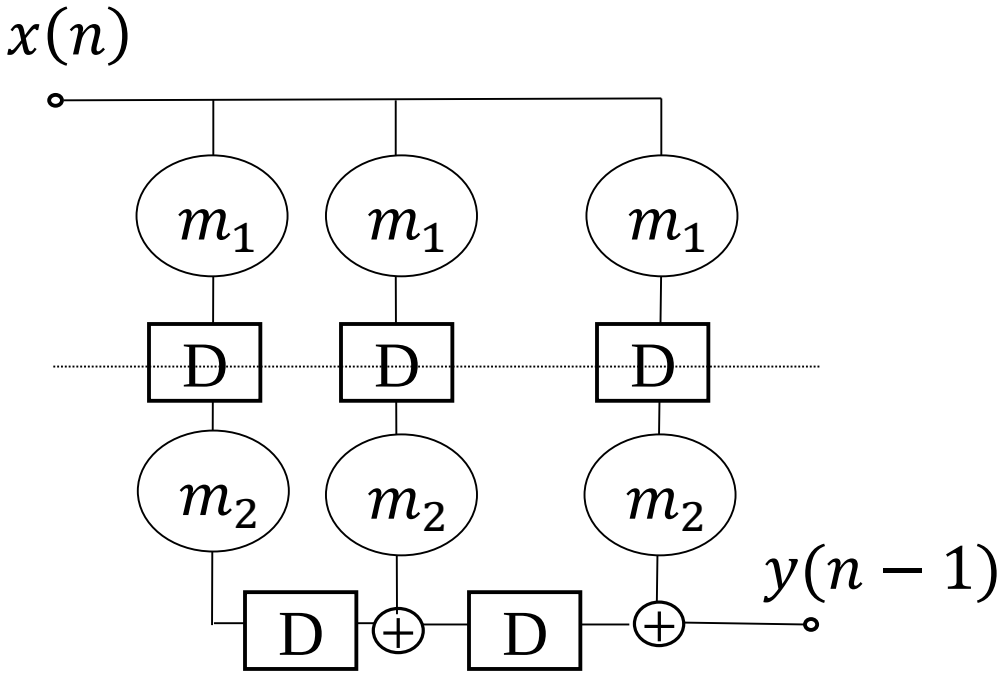
# Data-Broadcast Structure

- Transposition theorem : flow reversal preserves the functionality of the system.




critical path $= T_M + T_A$

# Fine-Grain Pipelining



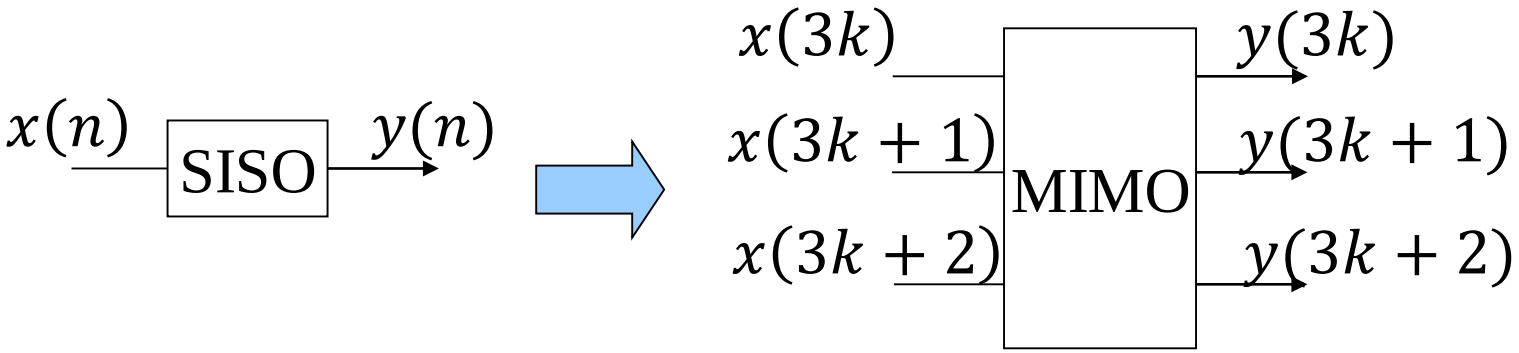
# Parallel Processing

pipelining
Parallel Processing
} duals of each other

Both exploit concurrency available in the computation in different ways

- pipelining : independent sets of computations are computed in an interleaved manner
- parallel processing : use duplicate hardware

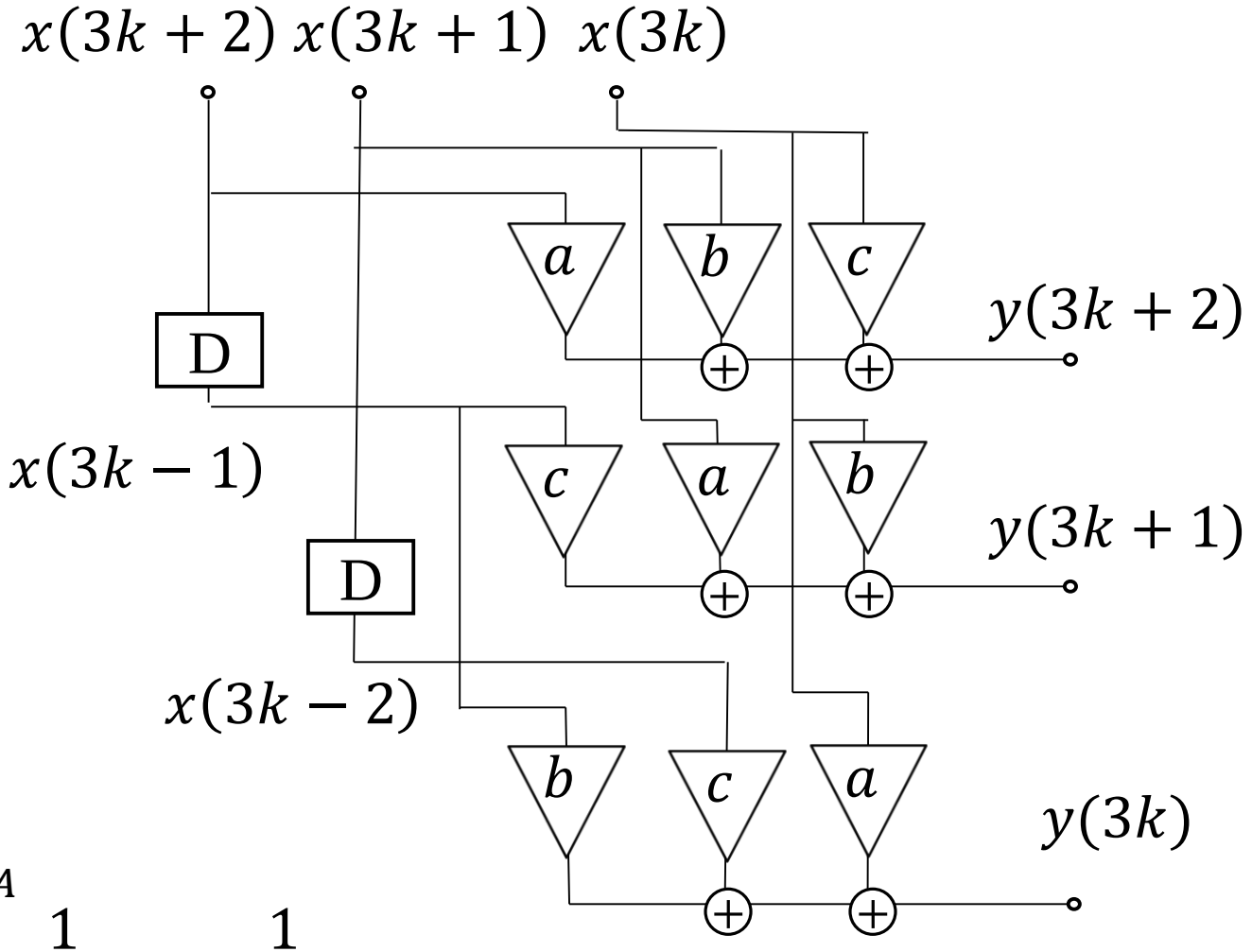
# Block Processing




Sequential System

3-Parallel System

# 3-Parallel 3-tap FIR Filter

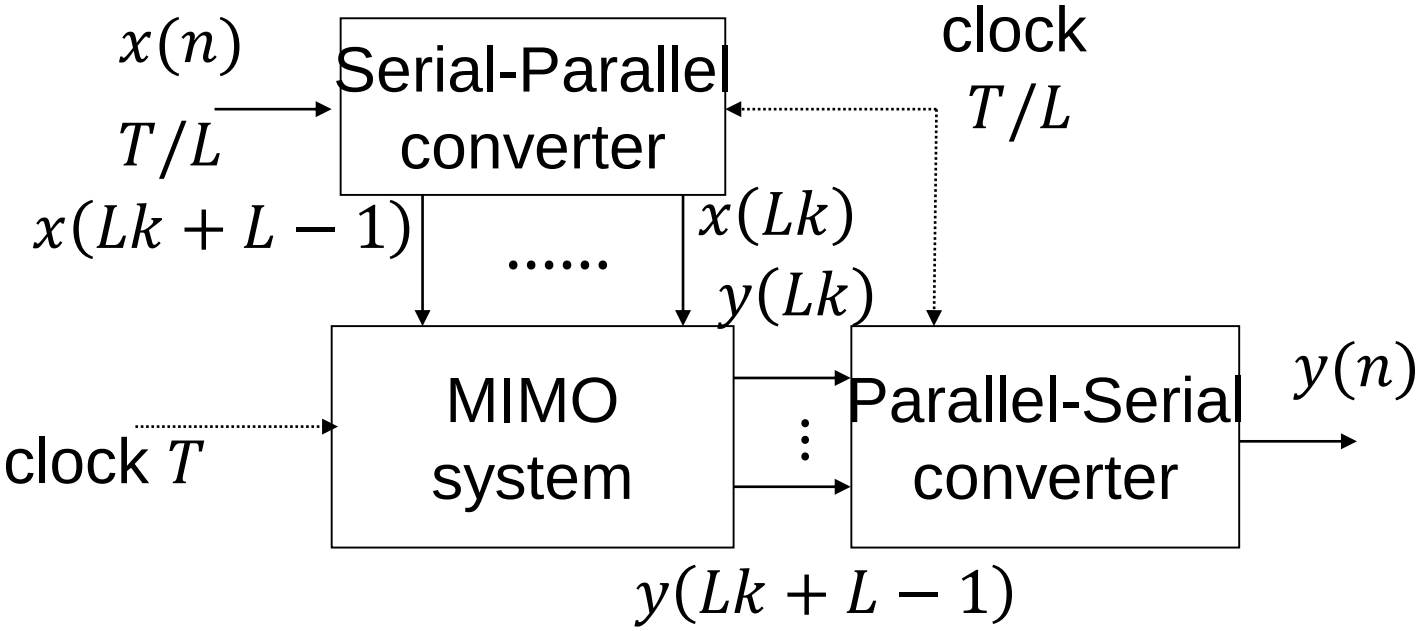



$D$: block delay

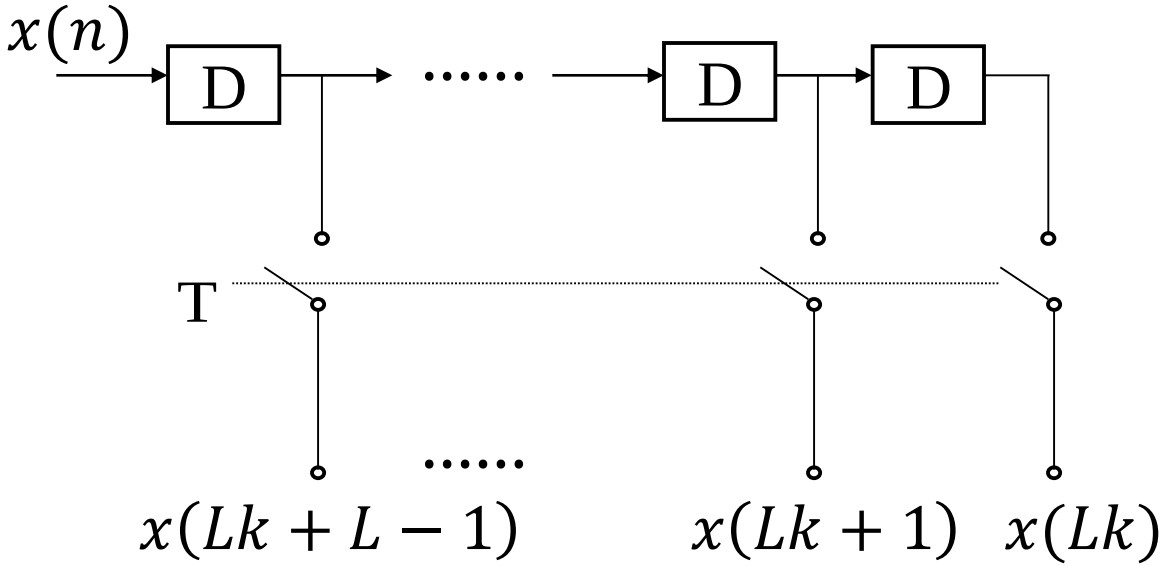
$T_{clk} \geq T_M + 2T_A$

$$T_{itr} = T_{sample} = \frac{1}{L} T_{clk} \geq \frac{1}{3}(T_M + 2T_A)$$

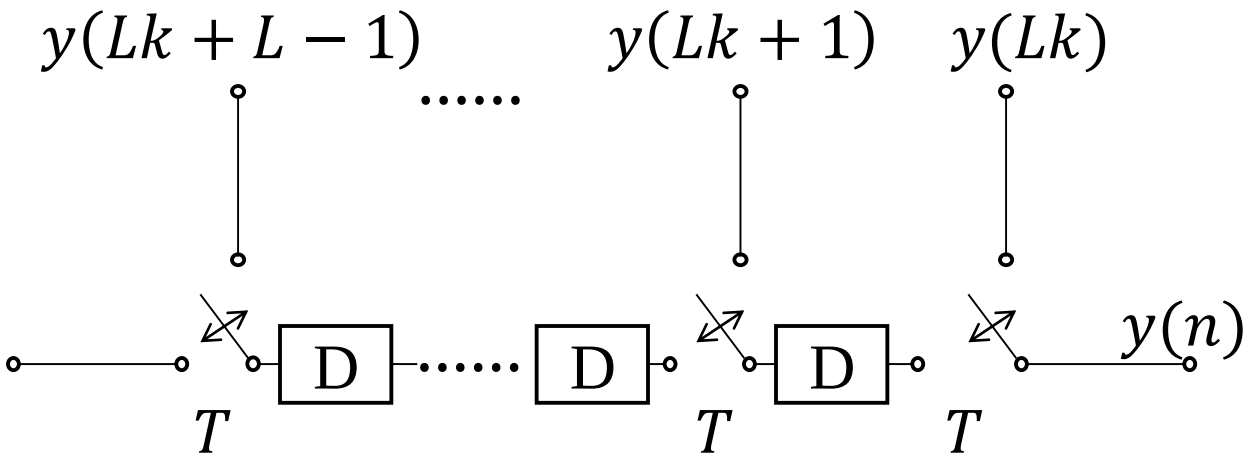
# Complete Parallel Processing System



# Serial-Parallel Converter



# Parallel-Serial Converter



# Pipelining vs. Parallel Processing


- If critical path < I/O delay
  - I/O bottleneck
  - communication bound
- pipelining can no longer increase the speed.

# Power Dissipation

- Propagation delay $T_{pd} = \frac{C_{charge} V_0}{k(V_0 - V_T)^2}$

  $C_{charge}$: capacitance to be charged/discharged, i.e. capacitance along the critical path

  $V_0$: supply voltage

  $V_T$: threshold

- Power Consumption $P = C_{total} V_0^2 f$

  $C_{total}$: total capacitance of the circuit

  $V_0$: supply voltage

  $f$: clock frequency

# Pipelining for Low Power

Pipelining reduces the critical path to $\frac{1}{M}$

capacitance $\rightarrow \frac{C_{charge}}{M}$

If $f$ is maintained, supply voltage can be reduced to $\beta V_0$ $(0 < \beta < 1)$

$$P_{pip} = C_{total}\beta^2 V_0^2 f = \beta^2 P_{seq}$$

$$\begin{cases} T_{seq} = \dfrac{C_{charge} V_0}{k(V_0 - V_T)^2} \\ T_{pip} = \dfrac{\frac{C_{charge}}{M}\beta V_0}{k(\beta V_0 - V_T)^2} \end{cases}$$

$$M(\beta V_0 - V_T)^2 = \beta(V_0 - V_T)^2$$

# Parallel Processing for Low Power

- $L$-parallel system increases the throughput $L$-times
- The clock period is increased to $LT_{seq}$
- There is more time to charge the same capacitance
- The supply voltage can be reduced to $\beta V_0$

$$LT_{seq} = \frac{C_{charge}\,\beta V_0}{k(\beta V_0 - V_T)^2}$$

$$\mathrm{L}(\beta V_0 - V_T)^2 = \beta(\mathrm{V}_0 - V_T)^2$$

$$P_{par} = LC_{charge}(\beta V_0)^2 \frac{f}{L} = \beta^2 P_{seq}$$

# Combining Pipelining and Parallel Processing

- Pipelining reduces the capacitance to be charged/discharged in 1 clock period
- Parallel processing increases the clock period for charging/discharging the original capacitance.
- Propagation delay of the parallel-pipelined system

$$LT_{pd} = \frac{(C_{charge}/M)\beta V_0}{k(\beta V_0 - V_T)^2} = \frac{LC_{charge}V_0}{k(V_0 - V_T)^2}$$

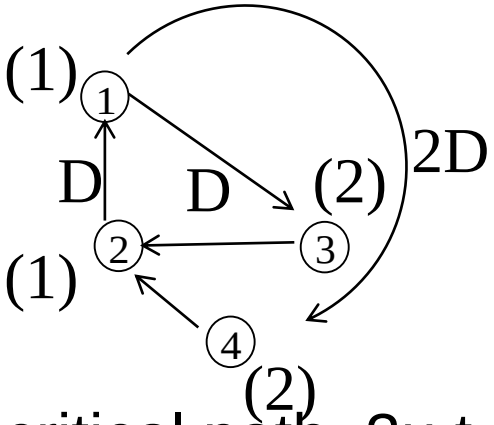
$$LM(\beta V_0 - V_T)^2 = \beta(V_0 - V_T)^2$$

# Retiming

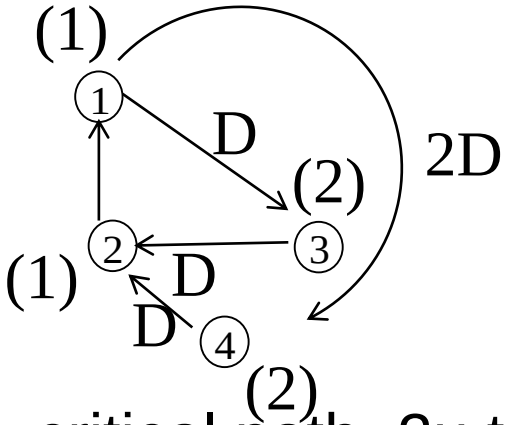
- Change the location of delay elements in a circuit without affecting the input-output characteristics.
- applications
  - reduce the clock period
  - reduce the number of registers
  - reduce the power consumption

# Retiming

- does not change the number of delays in a cycle.
- does not alter the iteration bound in a DFG

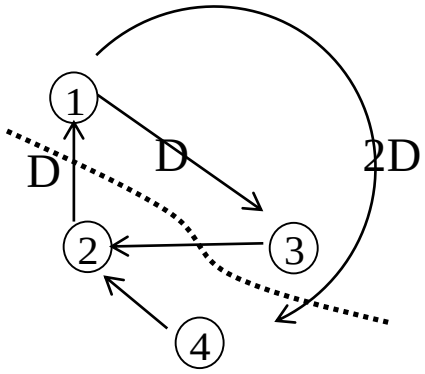


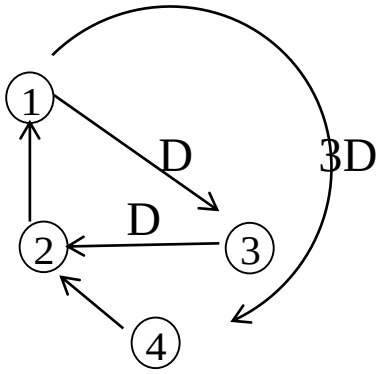

critical path=3u.t.




critical path=2u.t.

# Cutset Retiming




- If cutset is removed, 2 disconnected subgraphs are created, which are labeled $G_1$, $G_2$
- Add $k$ delays to each edge from $G_1$ to $G_2$
- Remove $k$ delays from each edge from $G_2$ to $G_1$

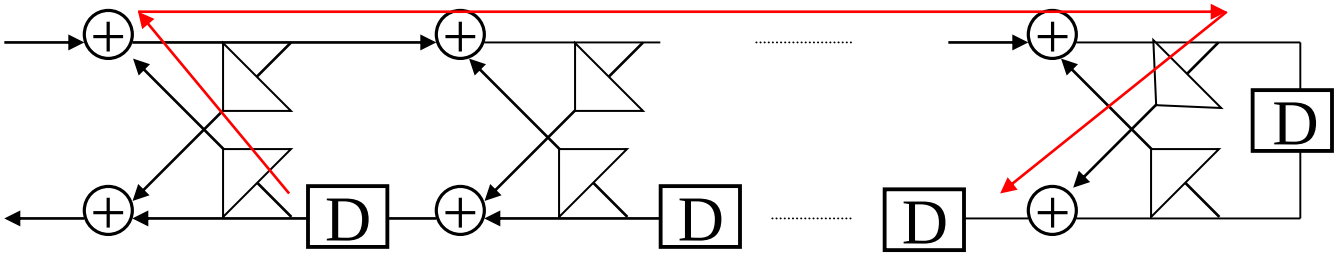
# Pipelining

- Special case of cutset retiming where there are no edges in the cutset from $G_2$ to $G_1$
- Pipelining applies to graphs without loops
- These cutsets are referred to as feed-forward cutsets.

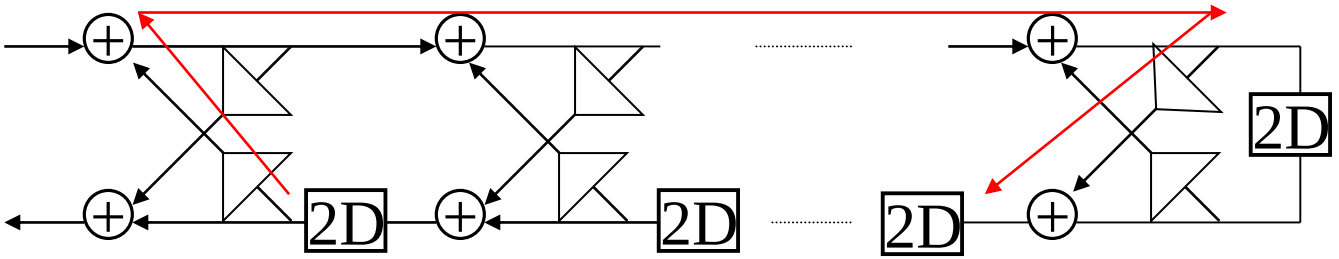
# Slow-Down

- Replace each delay in the DFG with $N$ delays to create $N$-slow version of the DFG.
- $N-1$ null operations (or $0$ samples) must be interleaved after each useful signal sample to preserve the functionality of the algorithm.

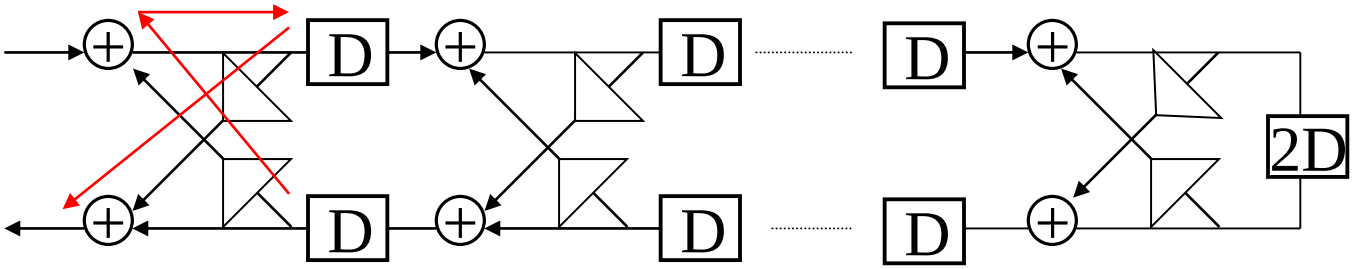
# Retiming to Reduce the Clock Period




100-stage lattice $101T_A+2T_M$



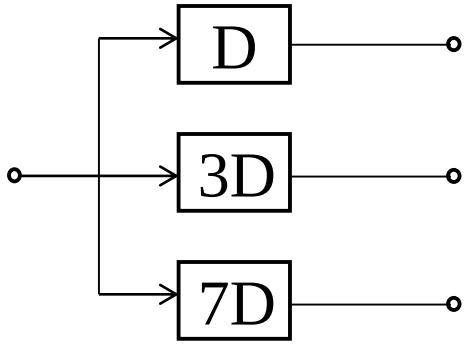

2-slow version



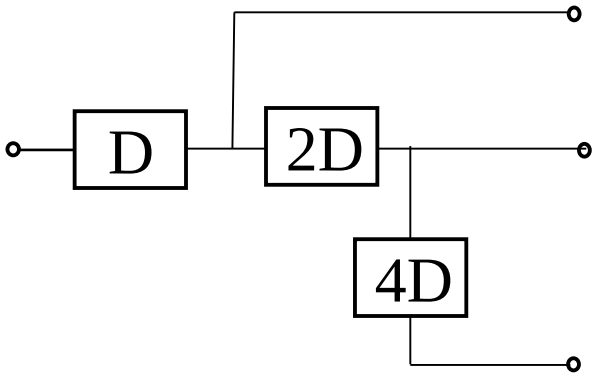

retimed version $2(2T_A+2T_M)$

critical path

# Retiming for Register Minimization




11 registers




7 registers

# Exercise 12

1. Consider the 4th-order FIR filter

   $y(n) = h(0)x(n) + h(2)x(n-2) + h(4)x(n-4).$

   Draw a topology for this filter such that the clock period is limited by 1 multiply-add time. Neither a broadcast structure nor new latches should be used.

2. An IIR filter

   $y(n) = ay(n-1) + x(n)$

   can be rewritten as

   $$y(n) = a\{ay(n-2) + x(n-1)\} + x(n)$$
   $$= a^2y(n-2) + ax(n-1) + x(n).$$

   Draw DFGs for the two filters, and compare their iteration bounds. Retime the latter DFG so that its iteration bound becomes minimum.

# Exercise 12 (cont.)

3. For the filter below,
   a. What is the loop bound and the iteration bound?
   b. What is the critical path?
   c. Pipeline or retime the filter to achieve a critical path equal to the iteration bound.