
DO NOT SHARE
SLIDES AND CLASS MATERIALS
ON ONLINE SITES
Course Hero

Advanced Logic Design
DSP Hardware Design Methodologies

Mingoo Seok

Columbia University

Readings: Parhi Sec. 2 and 3

Acknowledgement: Prof. Z. Zhang (UMich),
VLSI Digital Signal Processing Systems: Design and Implementation by Prof. K. K. Parhi (UMinn

Representing DSP Algorithms

- Example: 3-tap FIR filter

  $$y(n) = ax(n) + bx(n-1) + cx(n-2)$$

  – non-terminating, i.e., $n = 1 \ldots \infty$

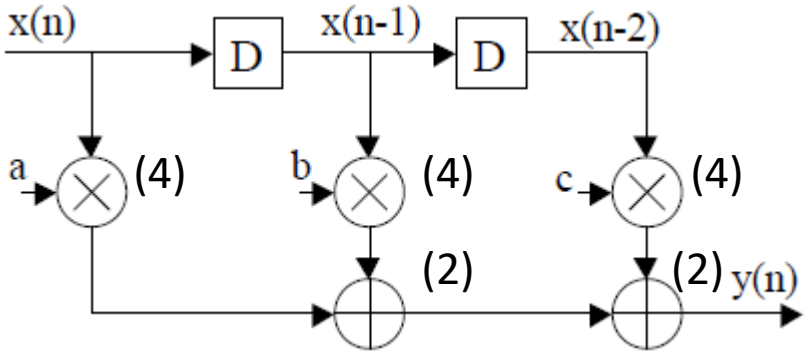
- Definitions:
  - **Iteration**: execution of all the computations in the algorithm once
  - **Iteration period**: the time required for one iteration
  - **Sampling rate** (or **throughput**): number of samples processed per second
  - **Critical path**: the longest path (computation time) between inputs and outputs (in combinational circuits), or the longest path between any two sequential elements (in sequential circuits)
  - **Minimum clock period**: determined by the critical path
  - **Latency**: the difference between the time an output is generated and the time at which its corresponding input was received by the system
    - Combinational circuit: latency is in terms of absolute time units
    - Sequential circuit: latency is in terms of number of clock cycles

(1) Block Diagram

- Example: 3-tap FIR filter

$$y(n) = ax(n) + bx(n-1) + cx(n-2)$$

  - non-terminating, i.e., $n = 1 \ldots \infty$



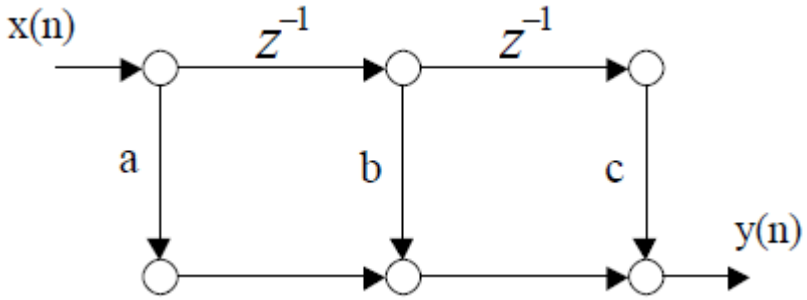
  - Consists of functional blocks connected with directed edges
  - Represents data flow from its input to output
  - Critical path is the longest path among paths that do not contain delay elements
  - 4 types of paths: input-delay, delay-output, input-output, delay-delay
  - Clock period is lower bounded by the critical path computation time

(2) Signal-Flow Graph (SFG)

- Example: 3-tap FIR filter

$$y(n) = ax(n) + bx(n-1) + cx(n-2)$$

  - non-terminating, i.e., $n = 1 \ldots \infty$



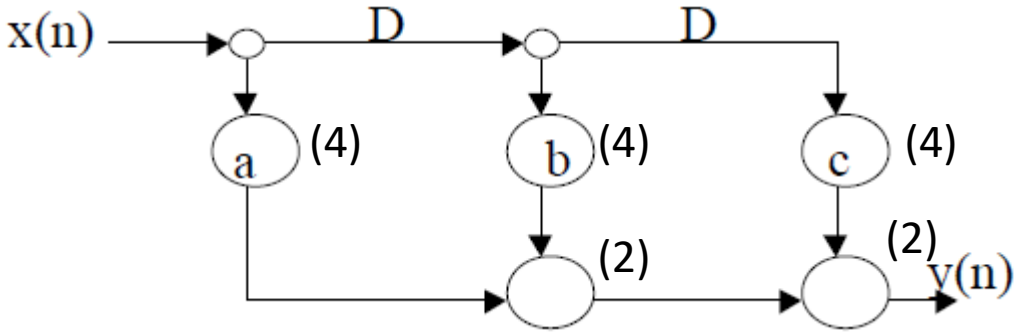
  - Represent additions and multiplications
  - Nodes represent variables for a computation
  - A directed edge denotes multiplication or delay
  - Multiple incoming edges and one outgoing edge denotes addition
  - Only used for linear systems; cannot be used to describe multirate DSP systems

(3) Data-Flow Graph (DFG)

- Example: 3-tap FIR filter

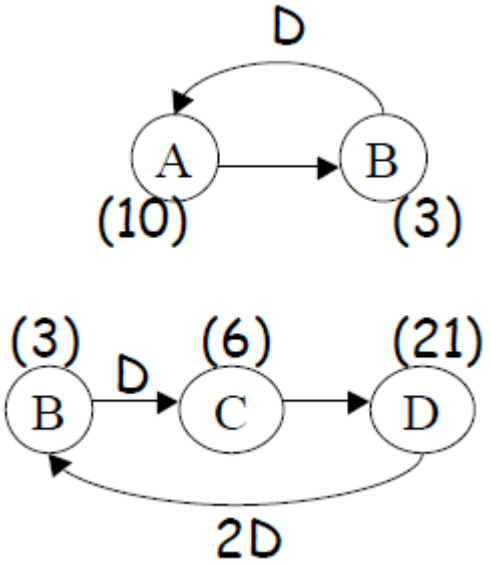
$$y(n) = ax(n) + bx(n-1) + cx(n-2)$$

  - non-terminating, i.e., $n = 1 \ldots \infty$



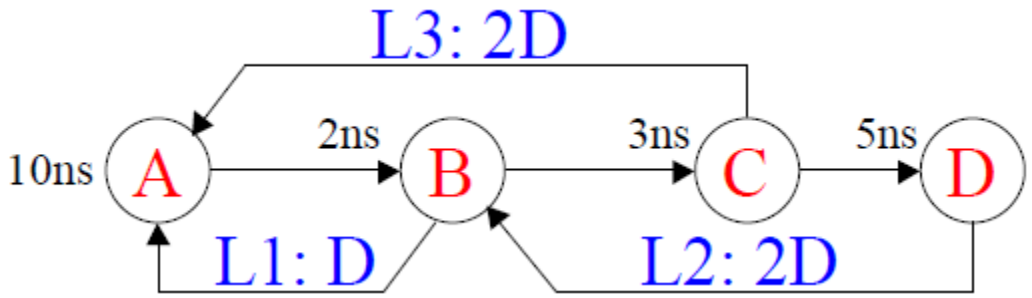
  - Nodes represent computations. Each has an execution time in absolute time units. i.e., numbers in the parenthesis
  - Directed edges represent data paths. It may have a delay in clock cycles.
  - Any node can fire when all inputs are available, and many nodes can fire concurrently
  - Intra-iteration precedence: every edge that has zero delay
  - Inter-iteration precedence: every edge that has non-zero delay
  - DFG is used to derive concurrent implementations onto parallel hardware

Loops



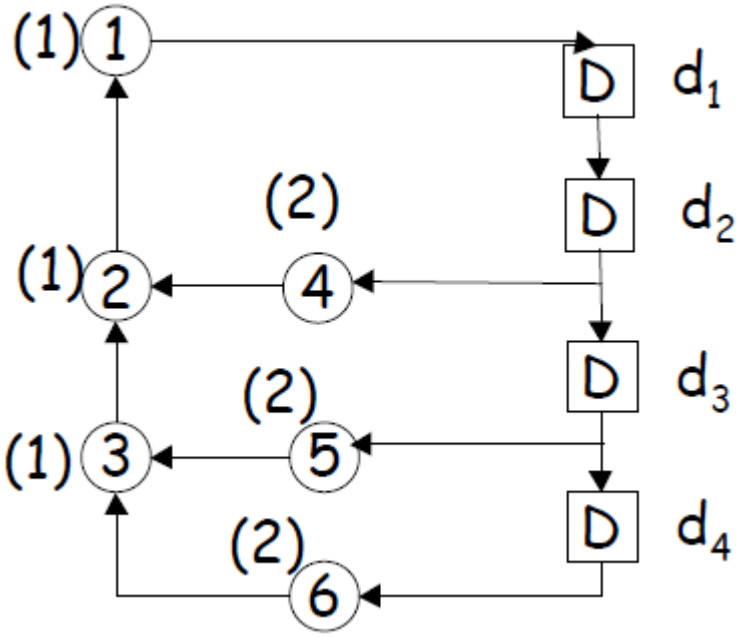
- **Loop**: directed path that begins and ends at the same node
- Loop bound: “loop computation time” divided by “loop delay”

Iteration Bound



- Three loops: L1, L2, L3.
- Loop bound of the l-th loop: t_l/w_l where t_l is the loop computation time, and w_l is the number of delays in the loop
  - Loop bound of L1: 12n/1D = 12
  - L2: 10n/2D = 5
  - L3: 15n/2D = 7.5
- **Critical loop**: loop with the maximum loop bound
  - L1
- **Iteration bound**: the loop bound of the critical loop
  - 12ns (= L1)
- Iteration bound is the *lower bound* on the iteration or sample period of the system (given speed of arithmetic operation and unlimited number of processing units). In other word, an implementation of the DSP program *can never achieve* an iteration period less than the iteration bound

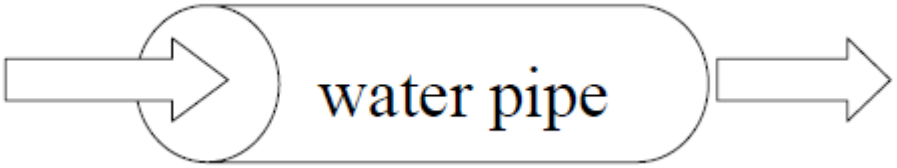
Compute Iteration Bound



- Enumerate loops and count loop bounds
  - Three loops: (1→4→2→1; 1→5→3→2→1; 1→6→3→2→1)
- Iteration bound is the maximum loop bound
  - Max(4/2, 5/3, 5/4) = 2

Pipelining and Parallel Processing

- Pipelining
  - Continue sending water without waiting for the water in the pipe to be drained

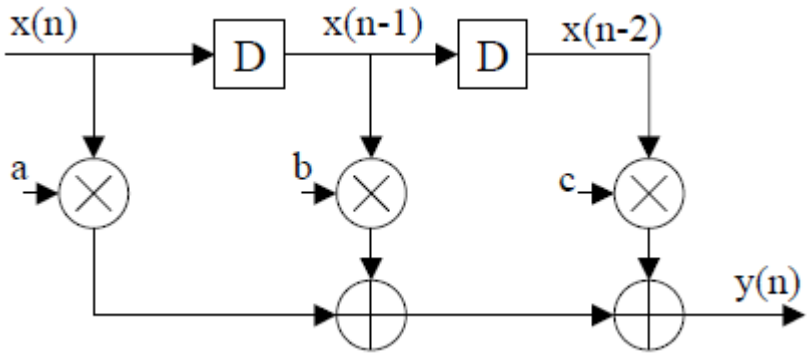


  - Shortens the critical path
  - Allows an increased clock speed, sampling rate (throughput) or reduced power consumption at the same throughput

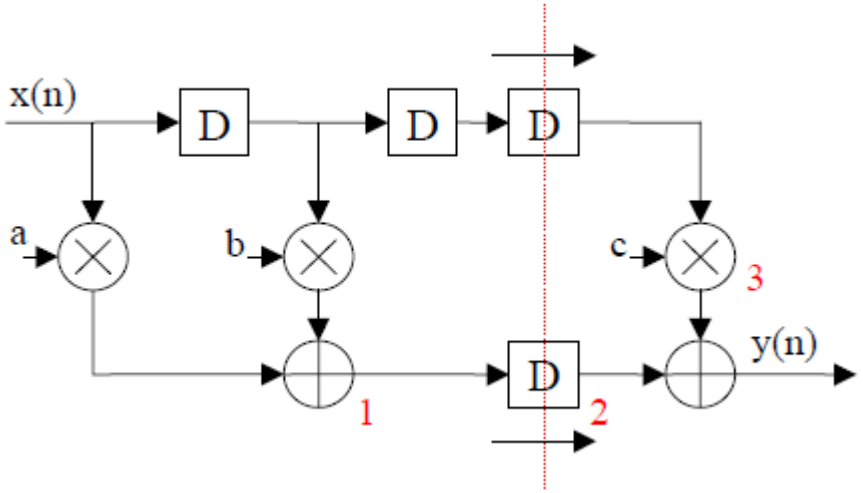
- Parallel processing
  - Multiple outputs are computed in parallel in a clock period
  - The effective sampling speed is increased by the level of parallelism
  - Allows an increased sampling rate (throughput) or reduced power consumption at the same throughput

Pipelining Digital Filters

– Direct-form 3-tap FIR filter

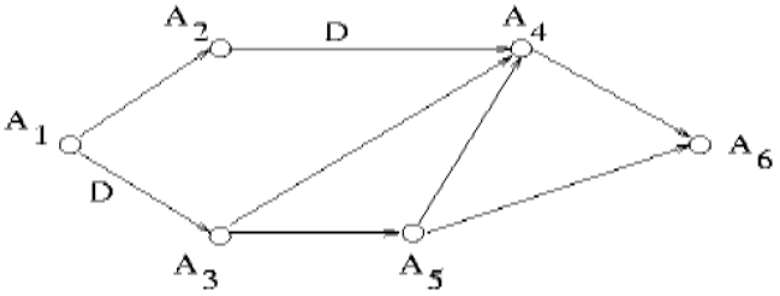


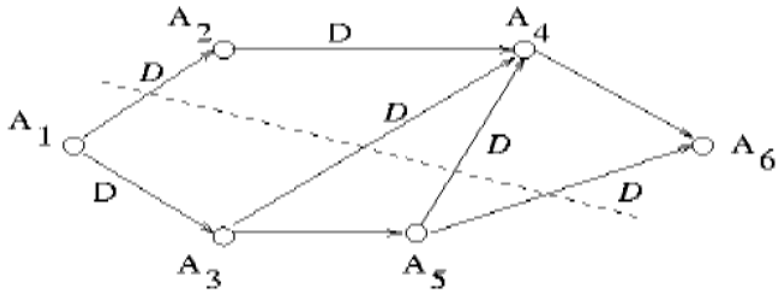
– Pipelined filter: reduces critical path, increases latency and register use



Pipeline Register Placement

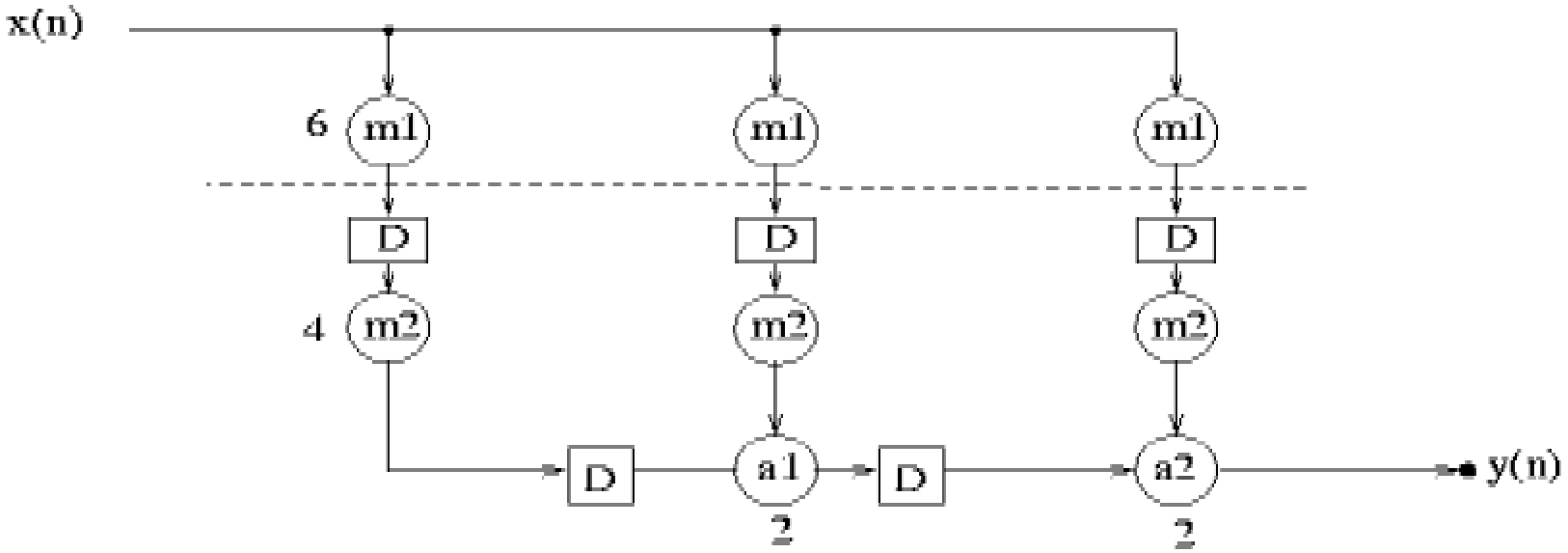
- Start with DFG
- Cut: separate the vertices of a graph into two disjoint subsets
- Cutset: the *set of edges* whose end points are in different subsets
- Feed-forward cutset: data move in the forward direction on all the edges of the cutset
- Place registers across any feed-forward cutset of the graph





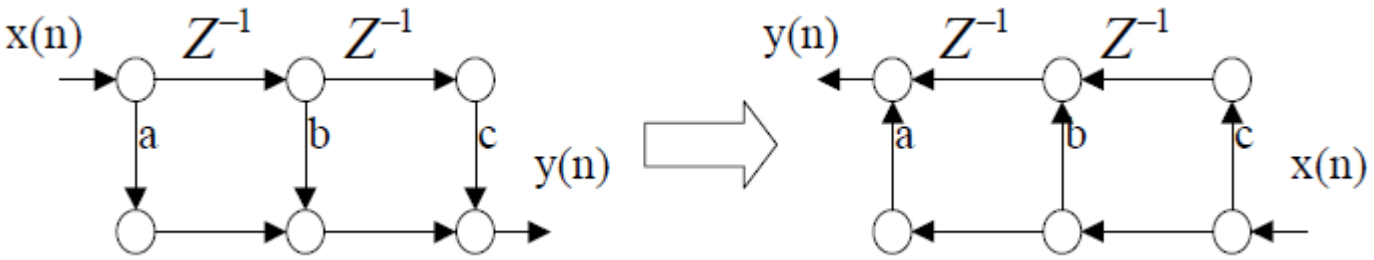
Fine-Grain Pipelining

- Break functional units for a better balance and shorten critical path

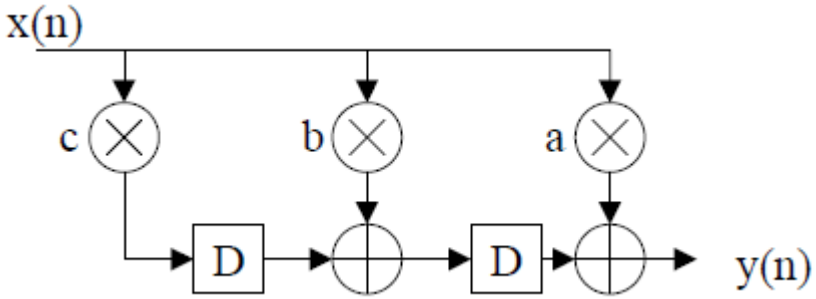


Data-Broadcast Structure

- Transposed SFG
  - Reverse the direction of all the edges in the SFG and interchange the input and output preserves the functionality of the system



- Data-broadcast structure
  - Reduced critical path delay



Parallel Processing

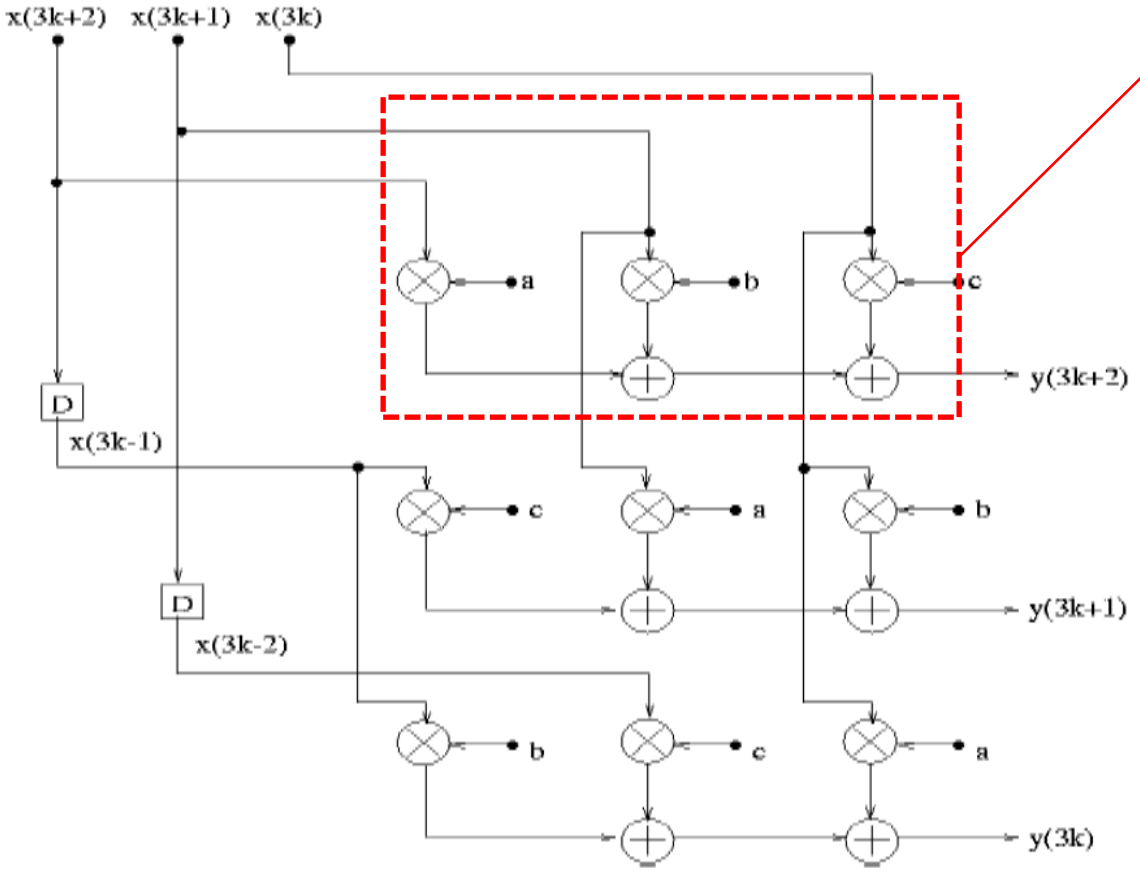
- Parallel processing is the dual of pipelining
  - Exploit concurrency in different ways:
    - Pipelining: break computation hardware into sets to improve utilization by concurrently operating on different samples
    - Parallel processing: duplicate computation hardware to operate on different samples
  - Example: $y(n) = ax(n) + bx(n-1) + cx(n-2)$
  - 3× parallel processing: *k* denotes clock cycle

$$y(3k) = ax(3k) + bx(3k-1) + cx(3k-2)$$
$$y(3k+1) = ax(3k+1) + bx(3k) + cx(3k-1)$$
$$y(3k+2) = ax(3k+2) + bx(3k+1) + cx(3k)$$

  - 3× sampling rate and 3× throughput at the same clock frequency
  - To maintain the same sampling rate, clock frequency can be reduced by 3×
  - Clock rate ≠ sampling rate

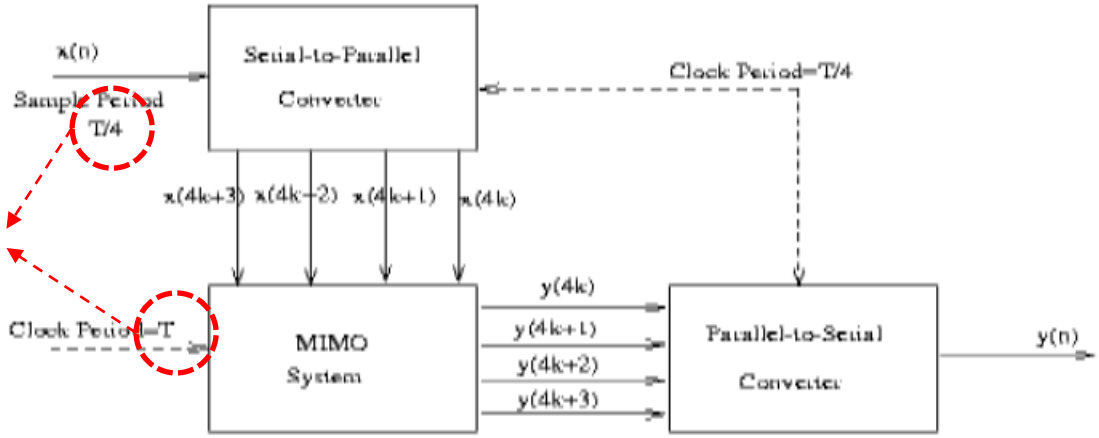
Parallel Processing

- 3× parallel direct-form FIR



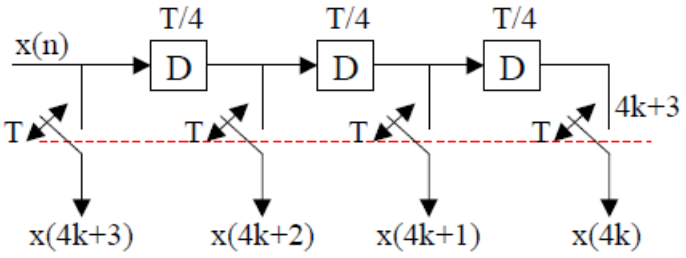
The computing core design is unchanged

Complete Parallel System

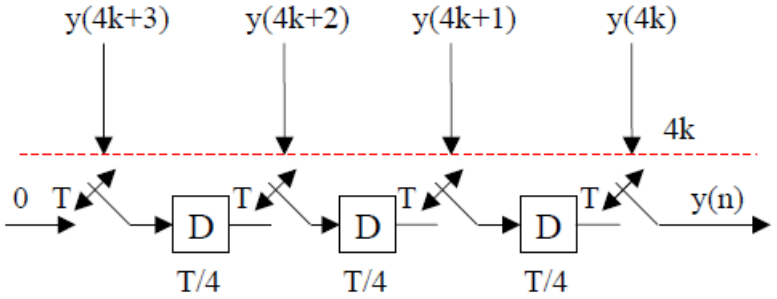


Computing is 4X slower

– Serial to parallel



Parallel to serial



Communication Bound

- Why only pipelining is used?
- Pipelining can be used only to the extent, until the critical path computation time is limited by the communication or I/O
  - Pipelining can no longer increase the speed once the bound is reached
  - Use parallel processing to further increase the speed