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

Advanced Logic Design
Systolic Architecture

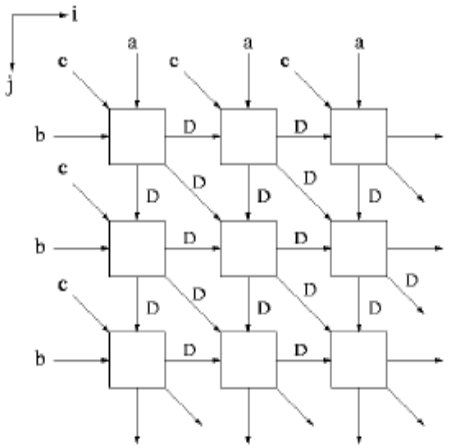
Mingoo Seok

Columbia University

Acknowledgement: Prof. Z. Zhang (UMich),
Prof. K. K. Parhi (UMinn)

Systolic Architecture (Systolic Array)

- **Two directions in parallel computing:** (1) coarse-grained multiprocessors and (2) fine-grained array processors (systolic array)
  - Systolic (heart contraction): regular pumping of blood by the heart
  - Systolic architecture: data-driven (data-flow), counterpart of the von Neumann paradigm (instruction-driven, control-flow)
  - Applications: matrix operations, image processing, computer graphics, cryptography, etc.



- A systolic array consists of matrix-like data processing units
  - Modular, regular and fully pipelined
  - System only contains local interconnections (This condition can be relaxed)
  - Each cell shares the information with its neighbors immediately after processing
  - Data flow across the array: different data flow in different directions

Dependence Graph

- Dependence graph (DG): a directed graph that shows the dependence of the computations
  - Nodes represent computations
  - Edges represent precedence constraints among nodes
  - A new node is created for a new computation
  - No node is ever reused on a single computation basis

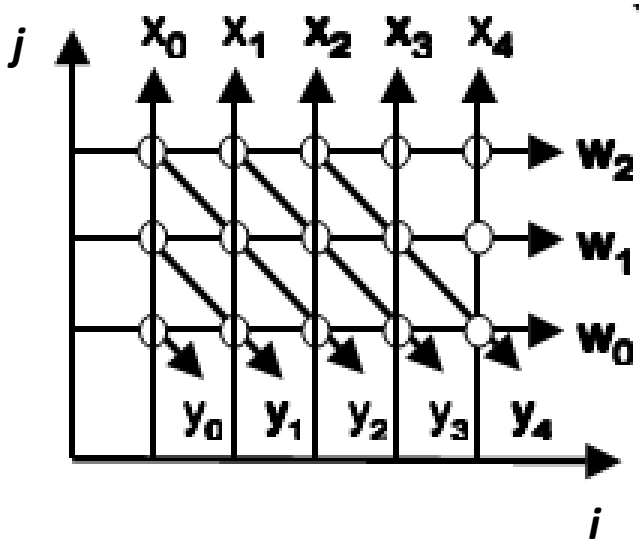
- DG is similar to DFG
  - Nodes in DFG cover the computations in one iteration and they are executed repetitively from iteration to iteration
  - DG contains computations for all iterations
  - DFG contains delay elements that store and pass data from current iteration to subsequent iterations
  - DG does not contain delay elements

FIR Filter Example

- 3-tap FIR filter

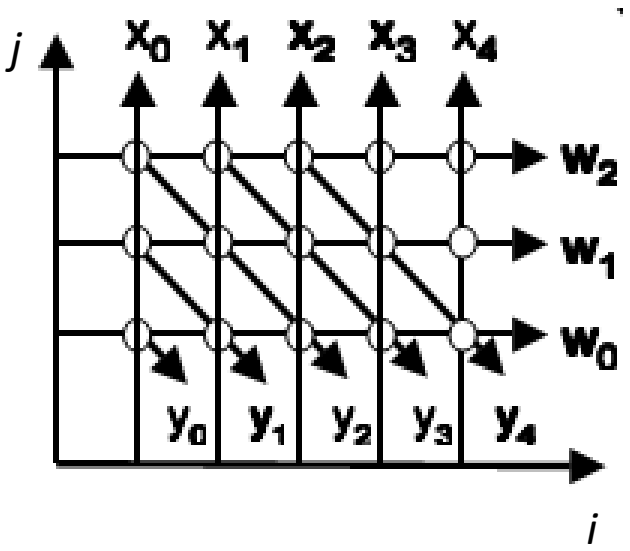
$$y(n) = w_0 x(n) + w_1 x(n-1) + w_2 x(n-2)$$

- 2-dimensional DG (space representation)
  - Each node has a coordinate (*i*, *j*) and can be mapped to a processor
  - Three data flows:
    Inputs broadcast/moves in the (0,1) direction
    Weights (coefficients) broadcast/moves in the (1,0) direction
    Outputs move in the (1,-1) direction
  - Non-terminating program: an infinite number of processors (nodes)
  - **Systolic design methodology:**
    - Map a 2-D DG to 1-D systolic array + time schedule, or
    - Map an N-D DG to a (N-1)-D systolic array + time schedule



FIR Systolic Array

- Iteration vector: $d = (1,0)$
  - Nodes spaced by d are assigned to the same processor
- Processor space vector: $p = (0,1)$
  - Node $I = (i, j)$ is mapped to processor $p\, I^T = (0,1)(i,j)^T = \boldsymbol{j}$
- Scheduling vector: $s = (1,0)$
  - Node $I = (i, j)$ is scheduled to run at time $s\, I^T = (1,0)(i,j)^T = \boldsymbol{i}$



- Hardware utilization efficiency (HUE)
  - HUE = $1/|s\, d^T|$
  - Two tasks executed by the same processor are spaced $|s\, d^T|$ time units apart

Feasibility Constraints

- $p\ d^T = 0$, i.e., processor space vector must be orthogonal to the iteration vector
  - If points A and B differ by the iteration vector (d), i.e., I_A-I_B is same as d, then they must be executed by the same processor. In other words, $p^T I_A = p^T I_B$. This leads to:

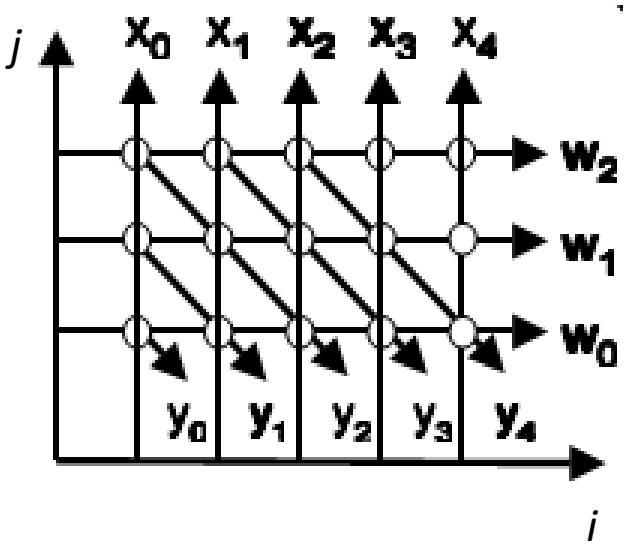
$$p^T(I_A - I_B) = 0 \rightarrow p^T d = 0$$

- $s\ d^T \neq 0$, i.e., schedule vector must not be orthogonal to the iteration vector
  - If A and B are mapped to the same processor, then they cannot be executed at the same time, i.e.,

$$s^T I_A \neq s^T I_B, \text{ i.e., } s^T d \neq 0$$

FIR Systolic Array

- Edge mapping:
  - If an edge e exists in the DG, then an edge $\mathbf{p}\cdot\mathbf{e}^{\mathbf{T}}$ is introduced in the systolic array with $\mathbf{s}\cdot\mathbf{e}^{\mathbf{T}}$ delays
- Example: the 3-tap FIR filter
  - $d = (1,0)$, $p = (0,1)$, $s = (1,0)$
  - Edge mapping table
  - HUE = $1/|s\ d^T| = 1$



e	pe^T	se^T
$wt(1,0)$	0	1
$input(0,1)$	1	0
$result(1,-1)$	−1	1

direction

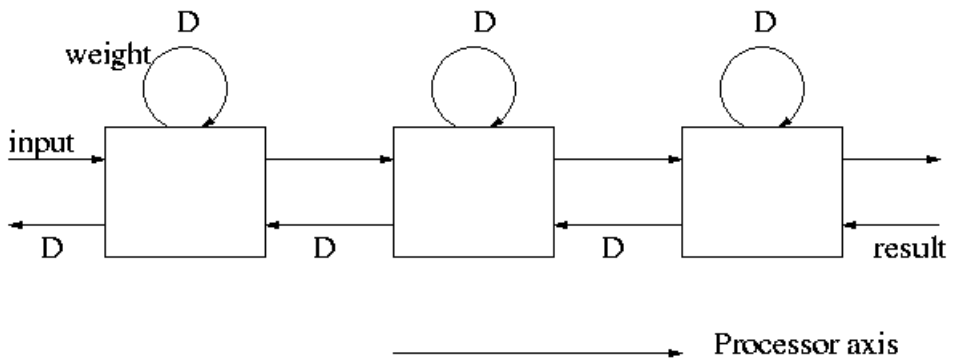
weight

FIR Systolic Array (1)

Broadcast Inputs, Move Results, Weights Stay

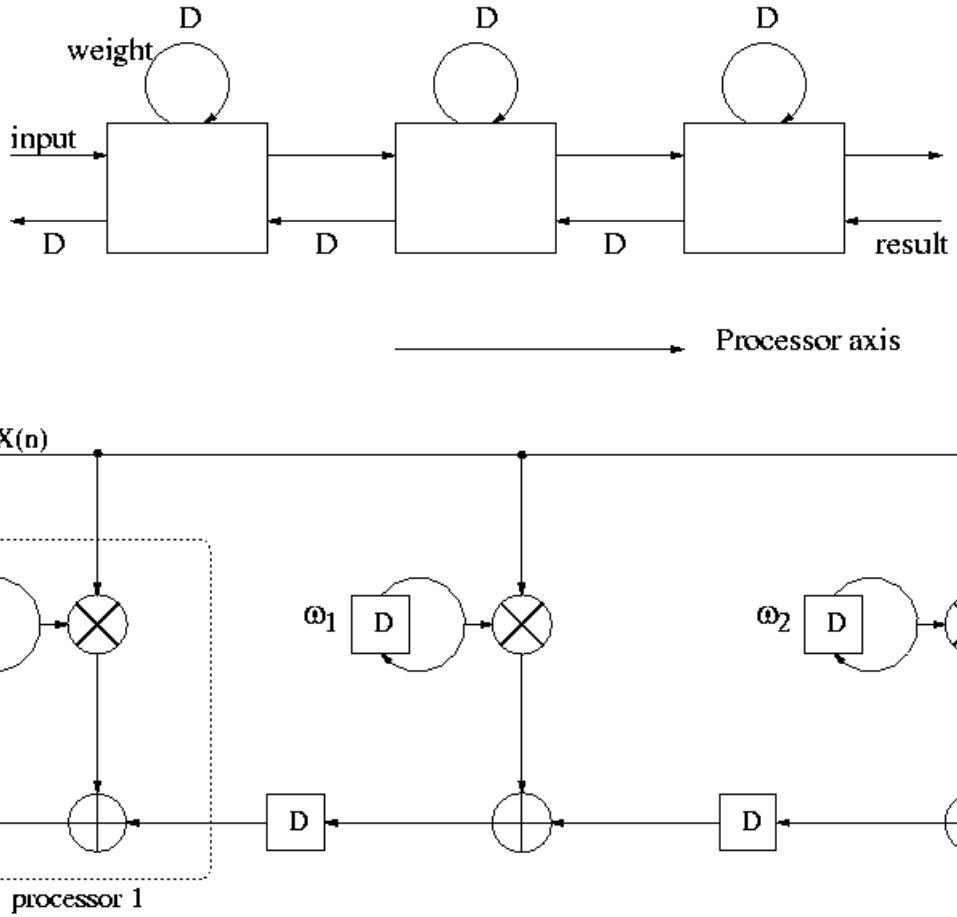
- Iteration vector: $d = (1,0)$
- Processor space vector: $p = (0,1)$
- Scheduling vector: $s = (1,0)$
- HUE = $1/|s\, d^T| = 1$

	Direction	Weight
e	pe^T	se^T
$wt(1,0)$	0	1
$input(0,1)$	1	0
$result(1,-1)$	-1	1



FIR Systolic Array (1)

Broadcast Inputs, Move Results, Weights Stay

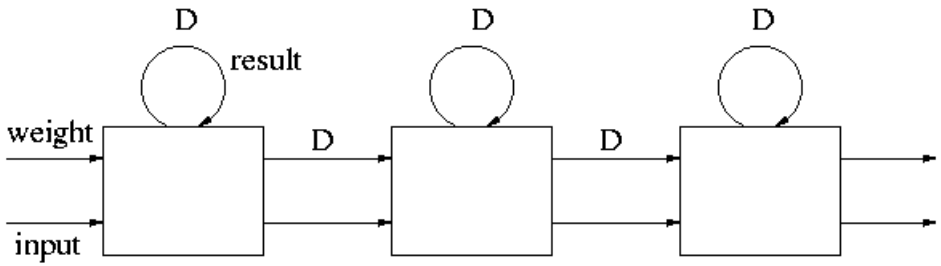


FIR Systolic Array (2)

Broadcast Inputs, Move Weights, Results Stay

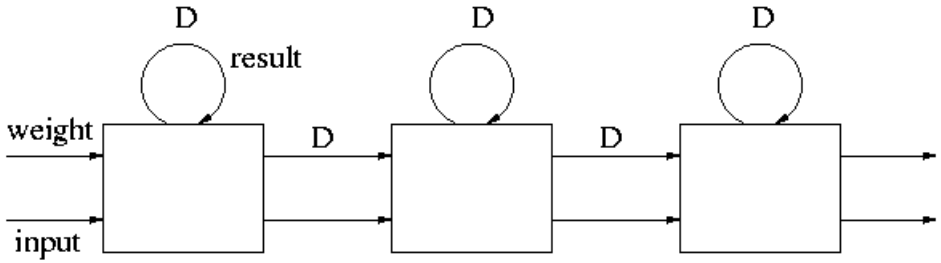
- Iteration vector: $d = (1,-1)$
- Processor space vector: $p = (1,1)$
- Scheduling vector: $s = (1,0)$
- HUE = $1/|s\, d^T| = 1$

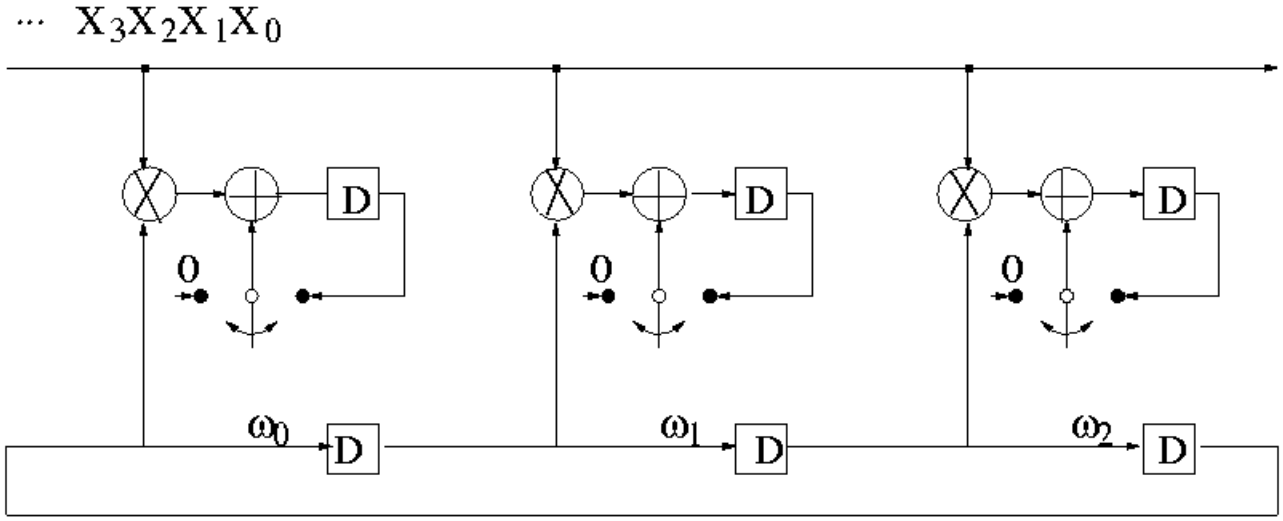
	Direction	Weight
e	pe^T	se^T
$wt(1,0)$	1	1
$input(0,1)$	1	0
$result(1,-1)$	0	1



FIR Systolic Array (2)

Broadcast Inputs, Move Weights, Results Stay



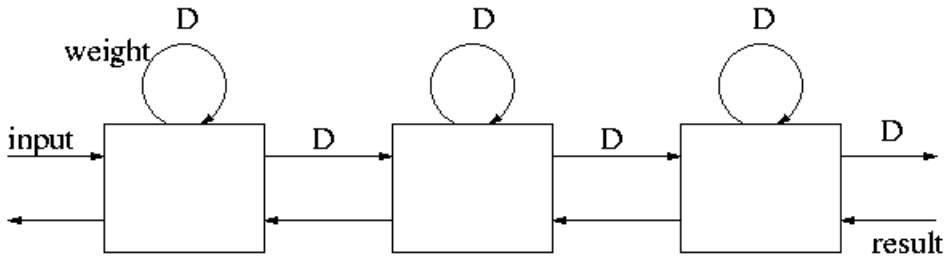


FIR Systolic Array (3)

Fan-in Results, Move Inputs, Weights Stay

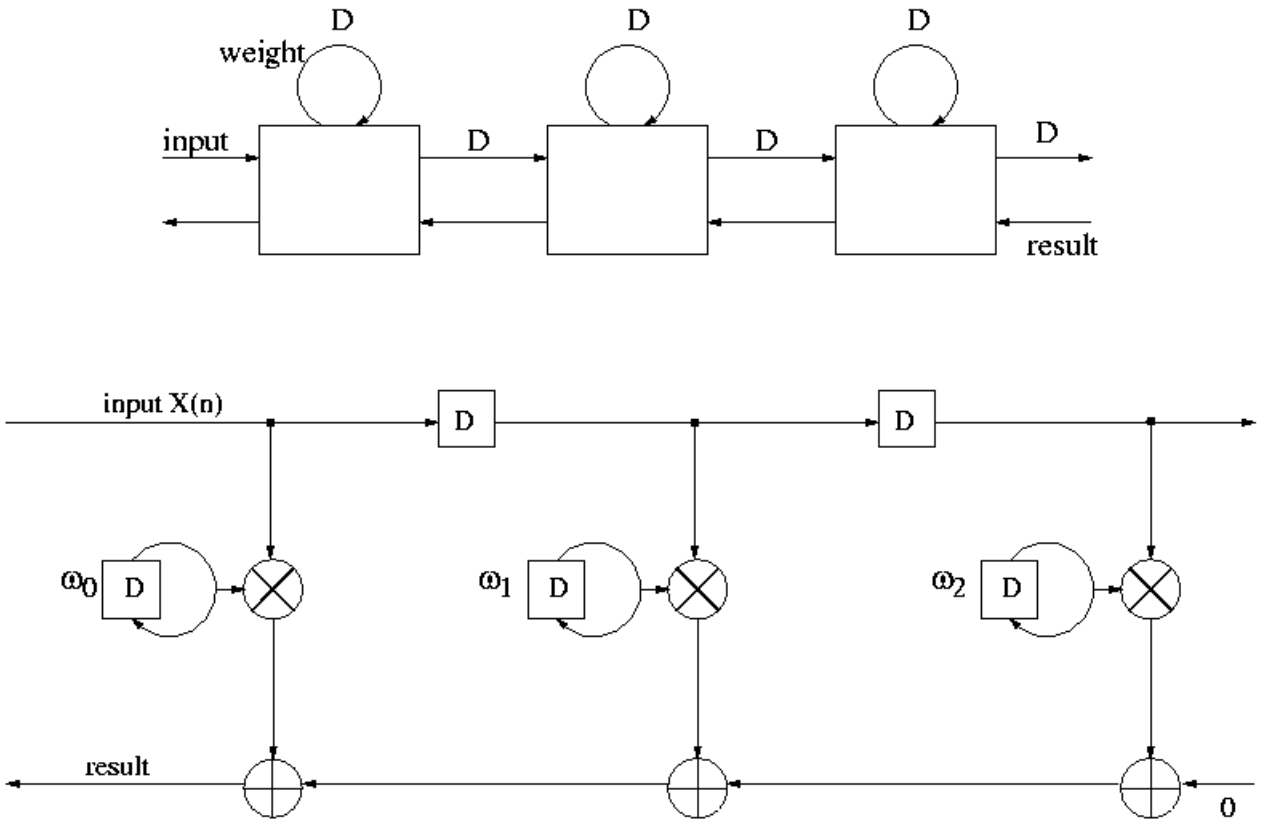
- Iteration vector: $d = (1,0)$
- Processor space vector: $p = (0,1)$
- Scheduling vector: $s = (1,1)$
- HUE = $1/|s\, d^T| = 1$

	Direction	Weight
e	pe^T	se^T
$wt(1,0)$	0	1
$input(0,1)$	1	1
$result(1,-1)$	-1	0



FIR Systolic Array (3)

Fan-in Results, Move Inputs, Weight Stay

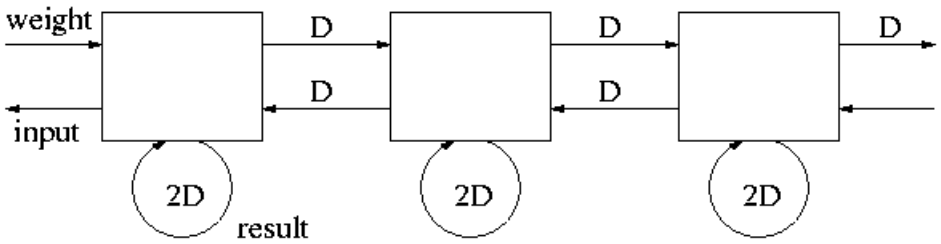


FIR Systolic Array (4)

Results Stay, Inputs and Weights Move in Opposite Directions

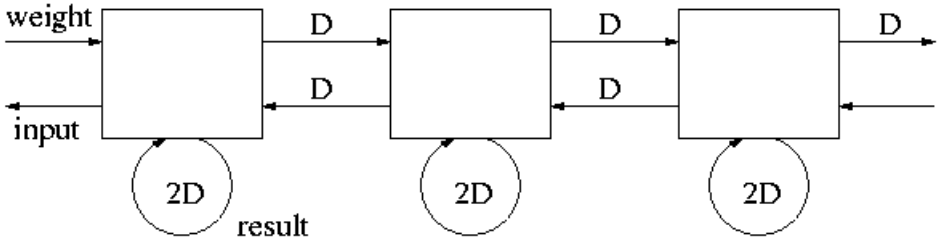
- Iteration vector: $d = (1,-1)$
- Processor space vector: $p = (1,1)$
- Scheduling vector: $s = (1,-1)$
- HUE = $1/|s\, d^T| = 1/2$

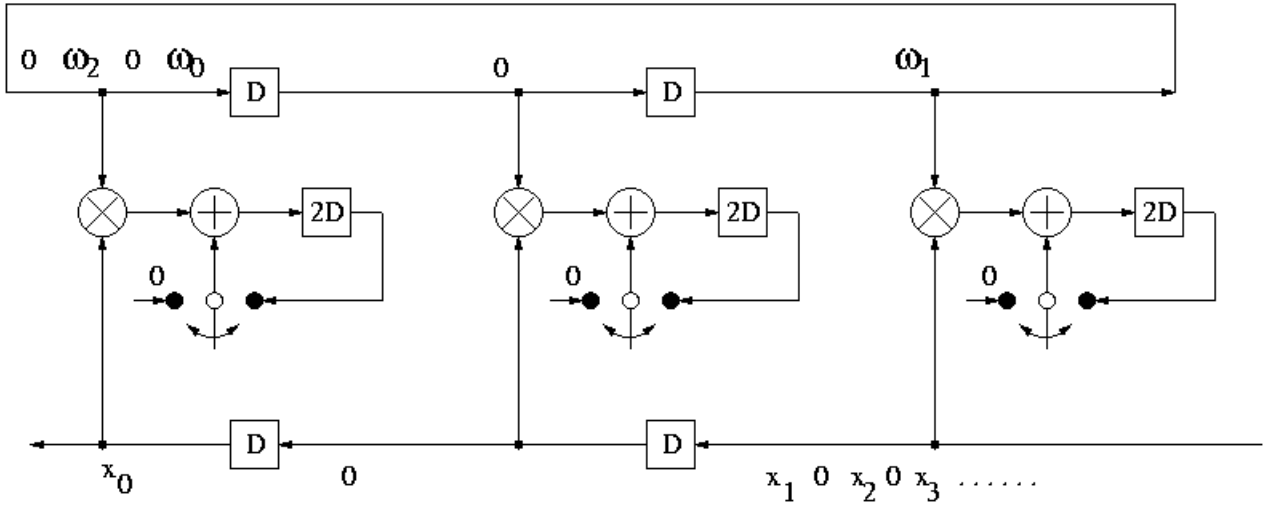
	Direction	Weight
e	pe^T	se^T
$wt(1,0)$	1	1
$input(0,-1)$	-1	1
$result(1,-1)$	0	2



FIR Systolic Array (4)

Results Stay, Inputs and Weights Move in Opposite Directions

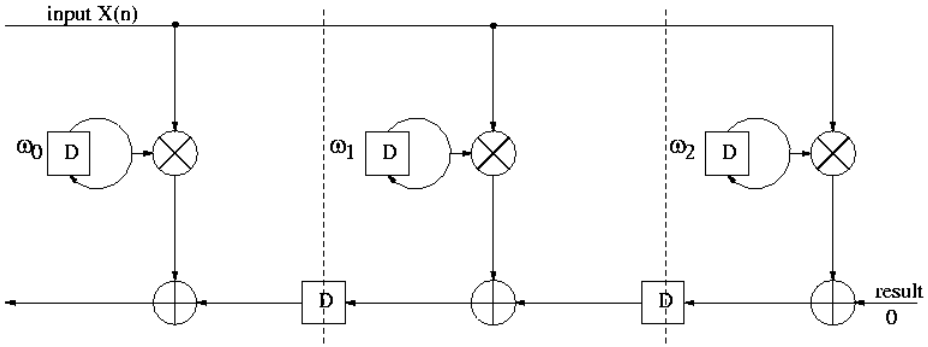




Systolic Design and Transforms

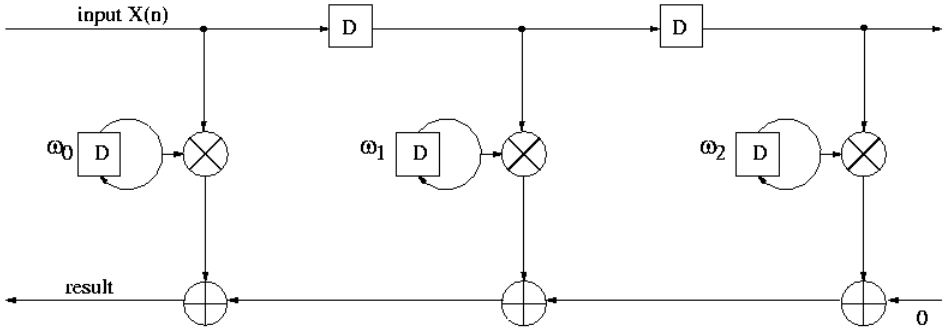
- FIR systolic architectures obtained using **the same iteration vector and processor vector**, but **different scheduling vectors**, can be derived from each other by transforms
  - Edge reversal: reversing the direction of an edge in the DG
  - Associativity: $(a + b) + c = a + (b + c)$
  - Slow-down
  - Retiming
  - Pipelining

Example



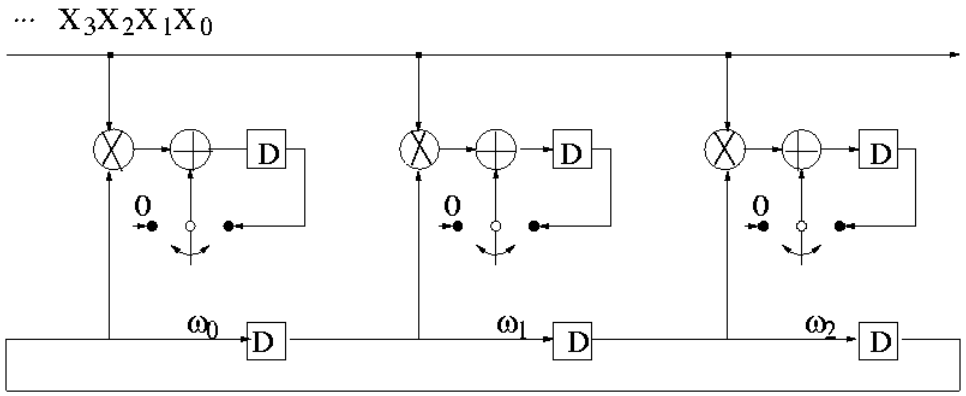
d=(1,0)
p=(0,1)
s=(1,0)

Cutset retiming



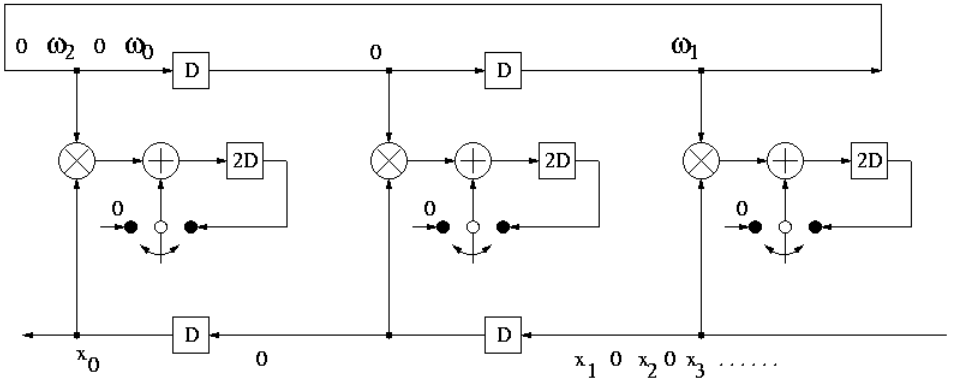
d=(1,0)
p=(0,1)
s=(1,1)

Example



d=(1,-1)
p=(1,1)
s=(1,0)

Slow down, edge reversal and retiming



d=(1,-1)
p=(0,1)
s=(1,-1)