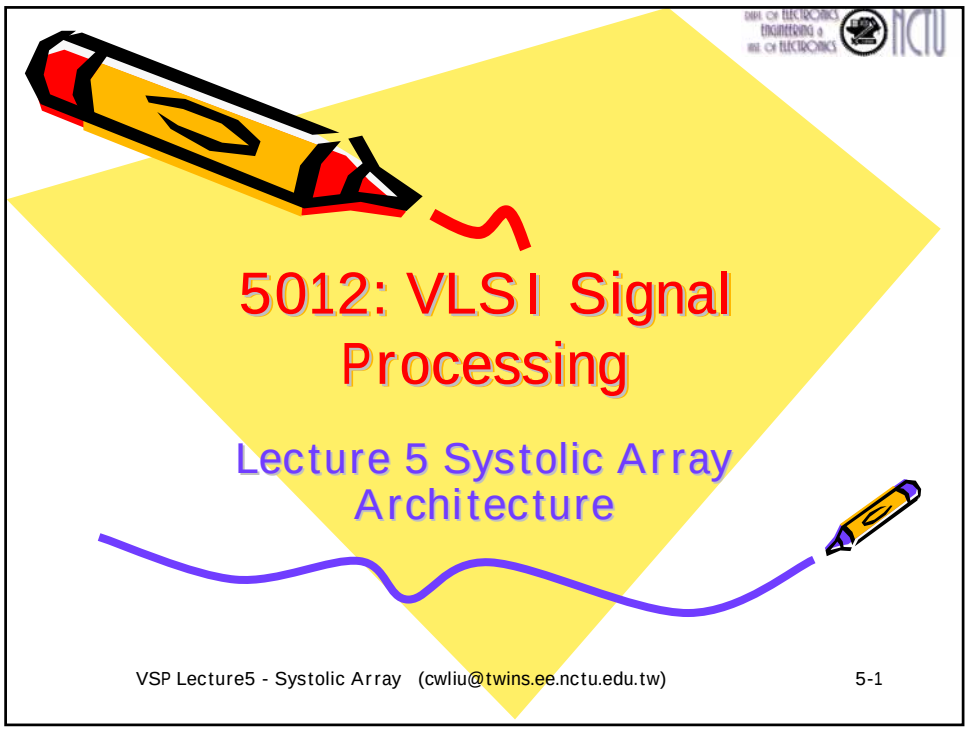


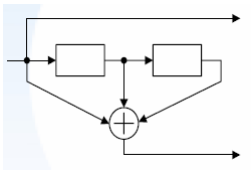
5012: VLSI Signal Processing

Lecture 5 Systolic Array Architecture

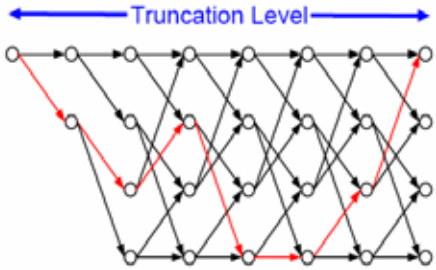
Techniques for VLSI Systems

- Algorithm Strength Reduction
  - Fast algorithm
    - Using polyphase filter bank to realize long-length taps linear phase filter
    - Using FFT instead of DFT
    - Fast convolution algorithm
  - Tradeoff between performance and complexity
    - Fast Trackback Viterbi algorithm (for convolutonal code, Turbo code)
    - Detect first and followed by BM algorithm for Reed-Solomon Code
    - Using CORDIC machine instead of complex multiplier
- Memory management
  - Memory bank, Register File
  - Local buffer (cache, or FIFO…)
  - Multiple-port memory is replaced by multiple single-port memory bank
- Power Management
  - Resource allocation
  - Using finite-state machine (FSM) to well timing and flow control, through enable/disable signals, in order to time-share the same PE
  - Clock gating, Data gating
  - Power-aware, Energy-aware design
- Low-power circuit design technology

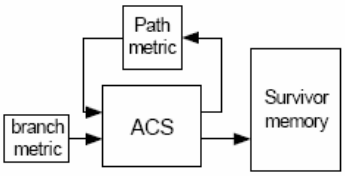
Convolutinal Code



Encoder



Trellis diagram
(or dependent graph)



Mapping Algorithms onto Array Structures

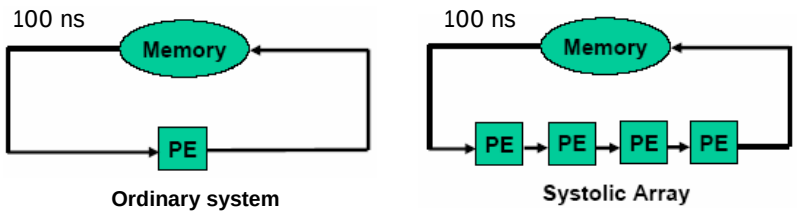
- Localized operations, intensive computations, and matrix operations are features of many DSP algorithms.
- Derive a maximal concurrency by using both pipelining and parallel processing
  - How is the inherent concurrency?
  - How is the array processor design dependent on the algorithm?
  - How is the algorithm best implemented in the array processor?
- Dependence graph (DG)
  - By tracing the associated space-time index space and using proper arcs to display the dependencies
  - It exhibits the full dependencies incurred in the execution of a specific algorithm
- Interconnection network
- Systolic Array
  - Modularity, regularity, local interconnection

History and Motivation

- Introduced by HT Kung and Leiserson, 1978
- Designs for matrix computations
- Illustrated by snapshots of operation
- Motivations
  - Improve performance of special-purpose systems (e.g. maximize processing per memory access)
  - Reduce design and implementation costs

What is a Systolic Architecture

- A network of processing elements (PEs) that computes and rhythmically passes data through it
- Multiple PEs to maximize processing per memory access

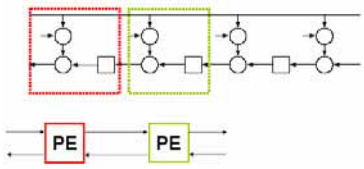


10 MOPS

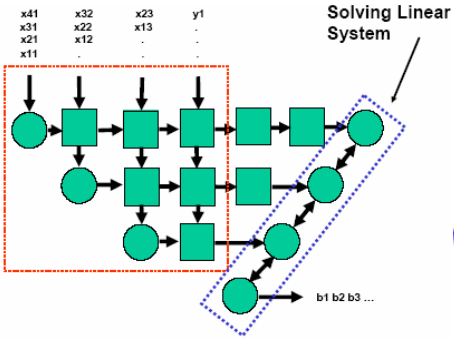
40 MOPS

Example

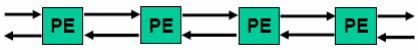
Systolic FIR filter



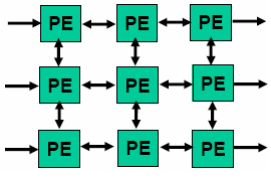
Least-Square Algorithm



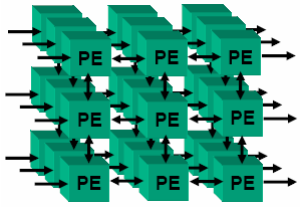
Array Structure Examples



1D array



2D array



3D array

Why Systolic Array

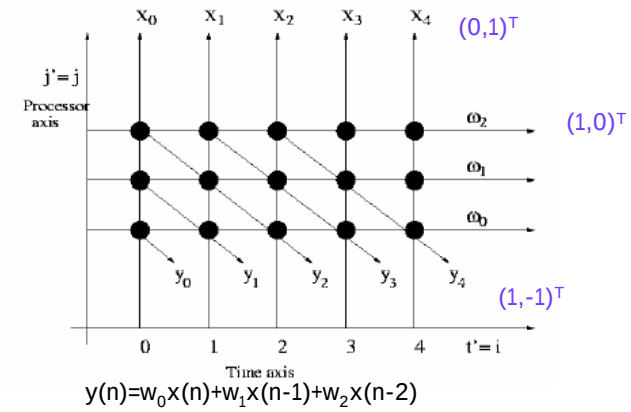
- A new class of pipelined array architectures
- Benefits
  - Simple and regular design (cost-effective)
  - Concurrency and communication
  - Modular and expandable
- Drawbacks
  - Not all algorithms can be implemented using a systolic architecture
  - Cost in hardware and area
  - Cost in latency

Systolic Fundamentals

- Systolic architecture are designed by using linear mapping techniques on regular dependence graph (DG)
- Regular Dependence Graph: the presence of an edge in a certain direction at any node in the DG represents presence of an edge in the same direction at all nodes in the DG
- DG corresponds to space representation → no time instance is assigned to any computation
- Systolic architectures have a space-time representation where each node is mapped to a certain processing element (PE) and is scheduled at a particular time instance.
- Systolic design methodology maps an N-dimensional DG to a lower dimensional systolic architecture

Regular Dependence Graph

- Space representation for FIR filter



$y(n)=w_0x(n)+w_1x(n-1)+w_2x(n-2)$

Definitions

- Projection vector $\mathbf{d}=\begin{pmatrix} d_1 \\ d_2 \end{pmatrix}$ (also called iteration vector)
  - Two nodes that are displaced by **d** or multiples of **d** are executed by the same processor
- Scheduling vector $\mathbf{s}^T=(s_1,s_2)$
  - Any node with index **I** would be executed at time $\mathbf{s}^T\mathbf{I}$
- Processor space vector $\mathbf{p}^T=(p_1,p_2)$
  - Any node with index $\mathbf{I}^T=(i,j)$ would be executed by processor

$$\mathbf{p}^T\mathbf{I}=(p_1,p_2)\begin{bmatrix} i \\ j \end{bmatrix}$$

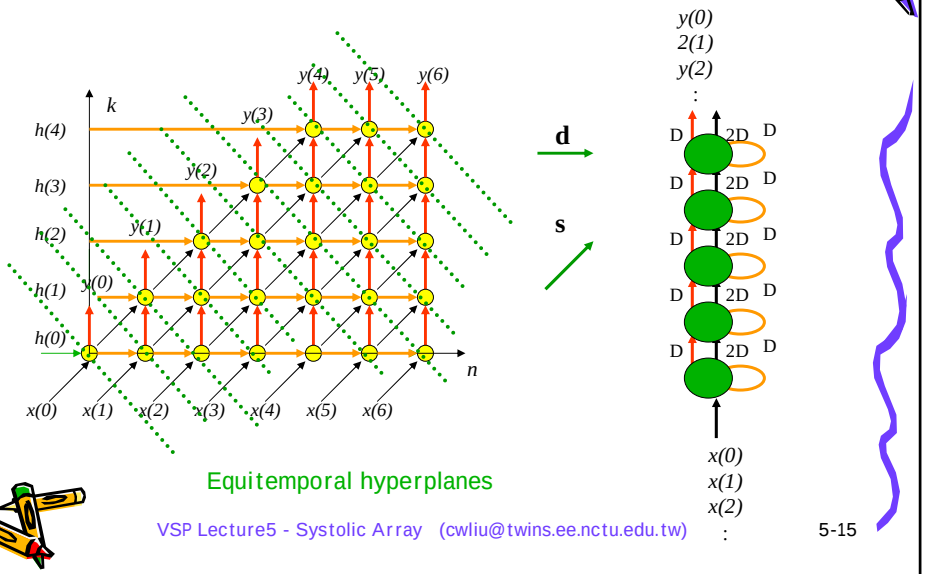
Systolic Design Methodology

- Many systolic architectures can be designed for a given algorithm by selecting different projection, processor space, and scheduling vectors.
- Feasibility constraints
  - If point $\mathbf{I}_A$ and point $\mathbf{I}_B$ differ by $\mathbf{d}$, $\mathbf{d}=\mathbf{I}_A-\mathbf{I}_B$, i.e. they are lying on the same direction along projection vector, they must be executed by the same processor. That is, $\mathbf{p}^T\mathbf{I}_A=\mathbf{p}^T\mathbf{I}_B$ or $\mathbf{p}^T\mathbf{d}=0$
  - If point $\mathbf{I}_A$ and point $\mathbf{I}_B$ are mapped to the same processor, i.e. $\mathbf{I}_A-\mathbf{I}_B=\mathbf{d}$, they cannot be executed at the same time. That is, $\mathbf{s}^T\mathbf{I}_A \neq \mathbf{s}^T\mathbf{I}_B$ or $\mathbf{s}^T\mathbf{d}\neq 0$
  - If an edge $\mathbf{e}$ exists in DG, then an edge $\mathbf{p}^T\mathbf{e}$ is introduced in the systolic array with $\mathbf{s}^T\mathbf{e}$ delay

Array Architecture Design

- Step 1: mapping algorithm to DG
  - Based on the space-time indices in the recursive algorithm
  - Shift-Invariance (Homogeneity) of DG
  - Localization of DG: broadcast vs. transmitted data
- Step 2: mapping DG to SFG
  - Processor assignment: a projection method may be applied (projection vector **d**)
  - Scheduling: a permissible linear schedule may be applied (schedule vector **s**)
    - Preserve the inter-dependence
    - Nodes on an equitemporal hyperplane should not be projected to the same PE
- Step 3: mapping an SFG onto an array processor

Example: FIR Filter



Equitemporal hyperplanes

Space-Time Representation

- The space representation or DG can be transformed to a space-time representation by interpreting one of the spatial dimensions as temporal dimension
- 2D DG:

$$\begin{bmatrix} i' \\ j' \\ t' \end{bmatrix} = T \begin{pmatrix} i \\ j \\ t \end{pmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ & \mathbf{p}^T & 0 \\ & \mathbf{s}^T & 0 \end{bmatrix} \begin{bmatrix} i \\ j \\ t \end{bmatrix}$$

Scheduling time instance

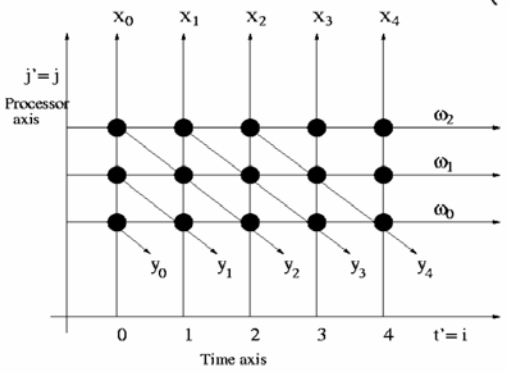
$$i' = t, \quad j' = \mathbf{p}^T I, \quad t' = \mathbf{s}^T I$$

Processor axis

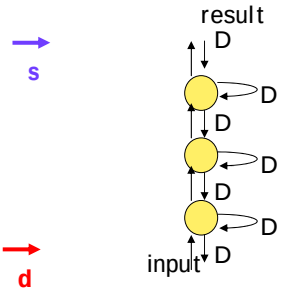
(2D-DG is mapped to a 1D systolic array)

Example

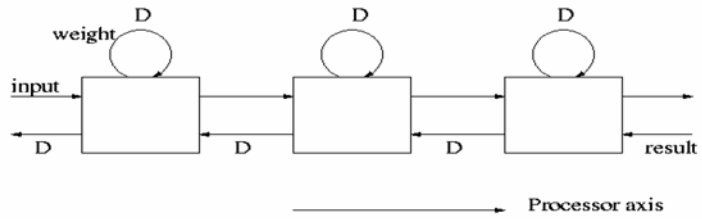
$d^T = (1\ 0),\ p^T = (0\ 1),\ s^T = (1\ 0)$



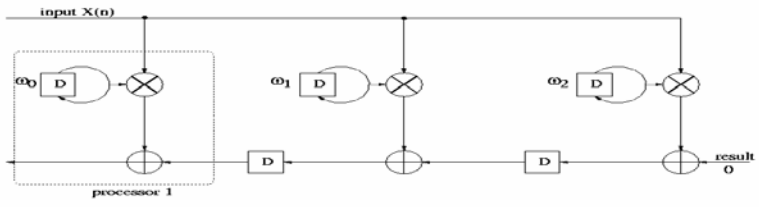
Space-time representation of B_1 design



e	$p^T e$	$s^T e$
wt(1 0)	0	1
i/p(0 1)	1	0
result(1 -1)	-1	1



Block diagram of B_1 design



Low-level implementation of B_1 design

Selection of Scheduling Vector

- Linear scheduling

$$S_X=\mathbf{s}^T I_X=(s_1\ s_2)\ (i_x,j_x)^T$$
$$S_Y=\mathbf{s}^T I_Y=(s_1\ s_2)\ (i_Y,j_Y)^T$$

- Affine scheduling (A transformation followed by a translation)

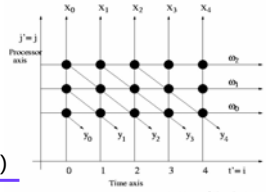
$$S_X=\mathbf{s}^T I_X+\gamma_x=(s_1\ s_2)\ (i_x,j_x)^T+\gamma_x$$
$$S_Y=\mathbf{s}^T I_Y+\gamma_Y=(s_1\ s_2)\ (i_Y,j_Y)^T+\gamma_Y$$

- For a dependence relation $X \rightarrow Y$, where $I_X^T=(i_x,j_x)^T$ and $I_y^T=(i_y,j_y)^T$. Then we have $S_Y \geq S_X+T_x$, where S_X and S_Y are scheduling times for node X and Y, respectively, and T_X is the computation time for node X.
- Each edge of a DG leads to an inequality for selection of scheduling vectors

Regular Iteration Algorithm (RIA)

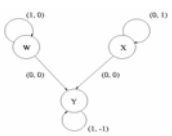
- Standard input RIA form
  - If the index of the inputs are the same for all equations
- Standard output RIA form
  - If all the output indices are the same
- For FIR filtering, we have O/I- relationship

$$w(i+1,j) = w(i,j)$$
$$x(i,j+1) = x(i,j)$$
$$y(i+1,j-1) = y(i,j) + w(i+1,j-1)x(i+1,j-1)$$


Space-time representation of B_1 design

- We can express it in standard output RIA form as

$$w(i,j) = w(i-1,j)$$
$$x(i,j) = x(i,j-1)$$
$$y(i,j) = y(i-1,j+1) + w(i,j)x(i,j)$$



- It is obvious that the FIR filtering problem cannot be expressed in standard input RIA form

Selection of s^T

- Capture all the fundamentals edge in the reduced dependence graph (RDG), which is constructed by the regular iteration algorithm (RIA)
- Construct the scheduling inequalities according $S_Y \geq S_x + T_x$, if there is an edge X → Y

$$s^T I_y + \gamma_y \geq s^T I_x + \gamma_x + T_x$$

Example

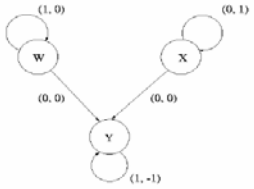
X → Y

$s^T I_y + \gamma_y \geq s^T I_x + \gamma_x + T_x$

→ $s^T (I_y - I_x) + \gamma_y - \gamma_x \geq T_x$

↑

e



Example :

$T_{mult} = 5$, $T_{add} = 2$, $T_{com} = 1$

Applying the scheduling equations to the five edges of the above figure we get ;

W-->Y : $e = (0\ 0)^T$, $\gamma_y - \gamma_w \geq 0$

X -->X : $e = (0\ 1)^T$, $s_2 + \gamma_x - \gamma_x \geq 1$

W-->W: $e = (1\ 0)^T$, $s_1 + \gamma_w - \gamma_w \geq 1$

X -->Y : $e = (0\ 0)^T$, $\gamma_y - \gamma_x \geq 0$

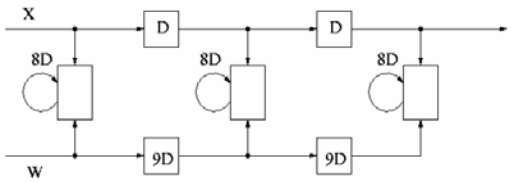
Y --> Y: $e = (1\ -1)^T$, $s_1 - s_2 + \gamma_y - \gamma_y \geq 5 + 2 + 1$

For linear scheduling $\gamma_x = \gamma_y = \gamma_w = 0$. Solving we get, $s_1 \geq 1$, $s_2 \geq 1$ and $s_1 - s_2 \geq 8$.

22

- Taking $s^T = (9\ 1)$, $d = (1\ -1)$ such that $s^T d \neq 0$ and $p^T = (1,1)$ such that $p^T d = 0$ we get HUE = 1/8. The edge mapping is as follows :

e	$p^T e$	$s^T e$
wt(1 0)	1	9
i/p(0 1)	1	1
result(1 -1)	0	8

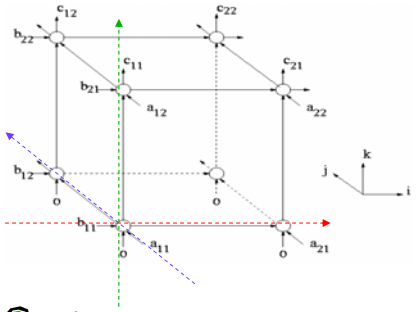


Systolic architecture for the example

Matrix-Matrix Multiplication

$$\begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$$

$$\begin{aligned} C_{11} &= a_{11}b_{11} + a_{12}\,b_{21} \\ C_{12} &= a_{11}b_{12} + a_{12}\,b_{22} \\ C_{21} &= a_{21}b_{11} + a_{22}\,b_{21} \\ C_{22} &= a_{21}b_{12} + a_{22}\,b_{22} \end{aligned}$$



a(i,j+1,k) = a(i,j,k)

b(i+1,j,k) = b(i,j,k)

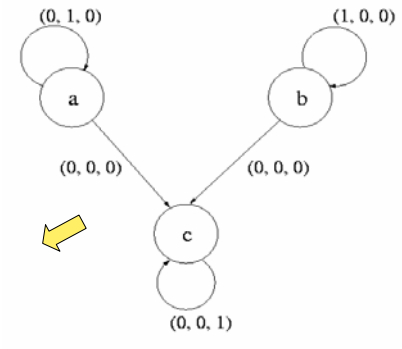
c(i,j,k+1) = c(i,j,k)+ a(i,j,k+1) b(i,j,k+1)

⇩ Standard output RIA form

a(i,j,k) = a(i,j-1,k)
b(i,j,k) = b(i-1,j,k)
c(i,j,k) = c(i,j,k-1) + a(i,j,k) b(i,j,k)

Example

$a(i,j,k) = a(i,j-1,k)$
$b(i,j,k) = b(i-1,j,k)$
$c(i,j,k) = c(i,j,k-1) + a(i,j,k)\, b(i,j,k)$



- Applying scheduling inequality with $T_{mult\text{-}add} = 1$, and $T_{com} = 0$ we get $s_2 \geq 0$, $s_1 \geq 0$, $s_3 \geq 1$, $\gamma_c - \gamma_a \geq 0$ and $\gamma_c - \gamma_b \geq 0$. Take $\gamma_a = \gamma_b = \gamma_c = 0$ for linear scheduling.
- Solution 1 :
  $s^T = (1,1,1)$, $d^T = (0,0,1)$, $p_1 = (1,0,0)$, $p_2 = (0,1,0)$, $P^T = (p_1\ p_2)^T$
- Solution 2 :
  $s^T = (1,1,1)$, $d^T = (1,1,-1)$, $p_1 = (1,0,1)$, $p_2 = (0,1,1)$, $P^T = (p_1\ p_2)^T$

Example

Sol. 1			Sol. 2		
e	p^Te	s^Te	e	p^Te	s^Te
a(0, 1, 0)	(0, 1)	1	a(0, 1, 0)	(0, 1)	1
b(1, 0, 0)	(1, 0)	1	b(1, 0, 0)	(1, 0)	1
C(0, 0, 1)	(0, 0)	1	C(0, 0, 1)	(1, 1)	1