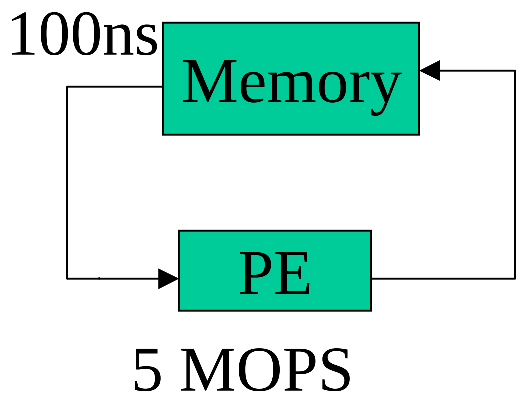
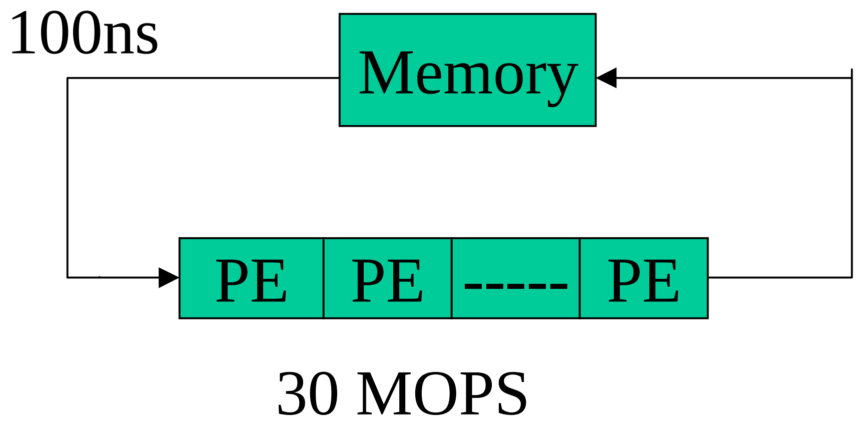
# SYSTOLIC PROCESSORS

# Systolic Computers and Systolic Architecture

# Systolic Systems

- Systolic computers are a new class of pipelined array architecture.
- The Basic Principle of a systolic system.





NOTE: MOPS⇒Millions of Operations Per Second

## Systolic Systems

- Systolic systems consists of an array of PE(Processing Elements)
- processors are called cells,
- each cell is connected to a small number of nearest neighbours in a mesh like topology.
- Each cell performs a sequence of operations on data that flows between them.
- Generally the operations will be the same in each cell, each cell performs an operation or small number of operations on a data item and then passes it to its neighbor.
- Systolic arrays compute in “lock-step” with each cell (processor) undertaking alternate compute/communicate phases.

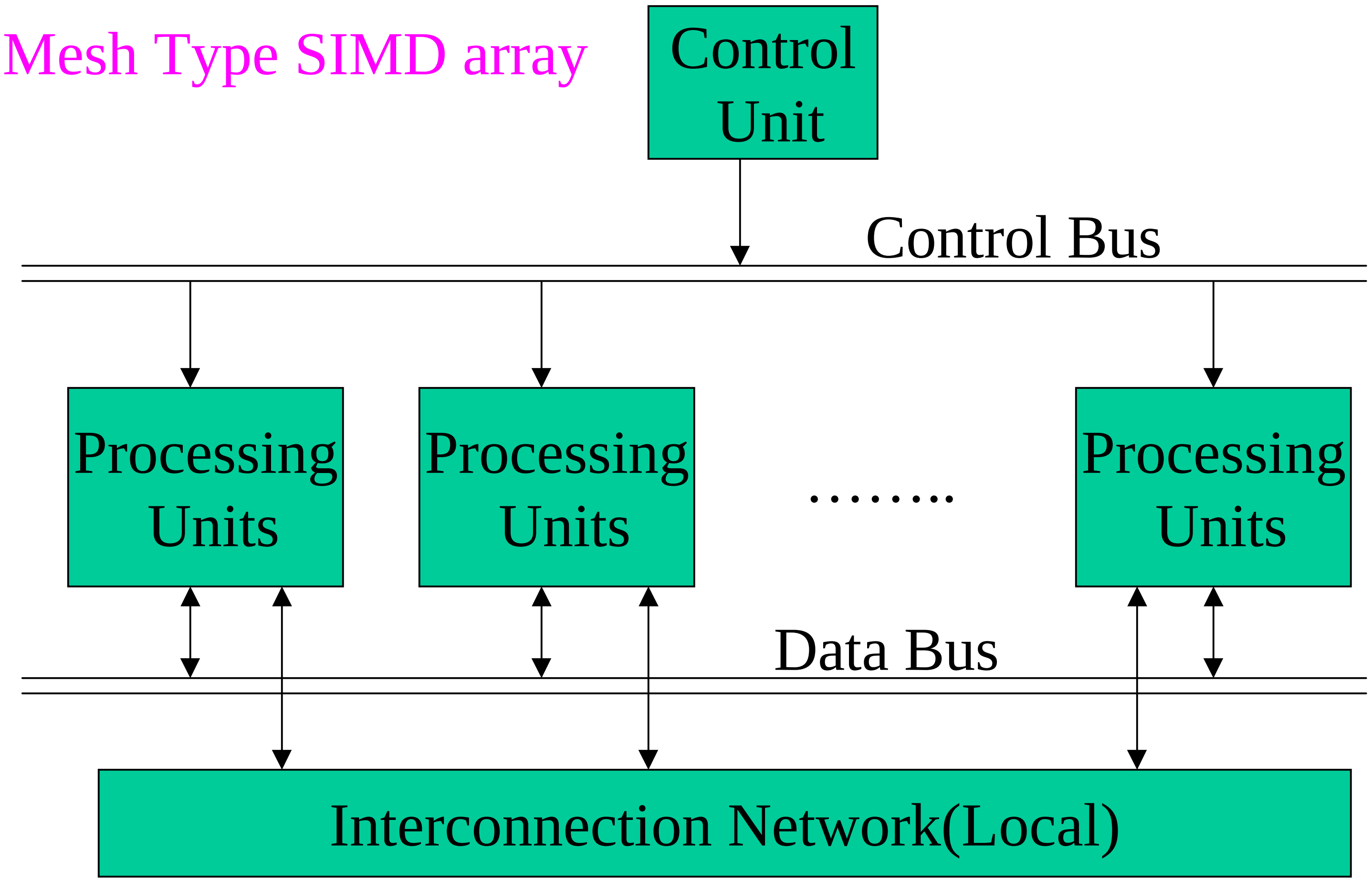
## Features of Systolic arrays

- A Systolic array is a computing network possessing the following features:
  - Synchrony,
  - Modularity,
  - Regularity,
  - Spatial locality,
  - Temporal locality,
  - Pipelinability,
  - Parallel computing.
- **Synchrony** means that the data is rhythmically computed (Timed by a global clock) and passed through the network.
- **Modularity** means that the array(Finite/Infinite) consists of modular processing units.
- **Regularity** means that the modular processing units are interconnected with homogeneously.

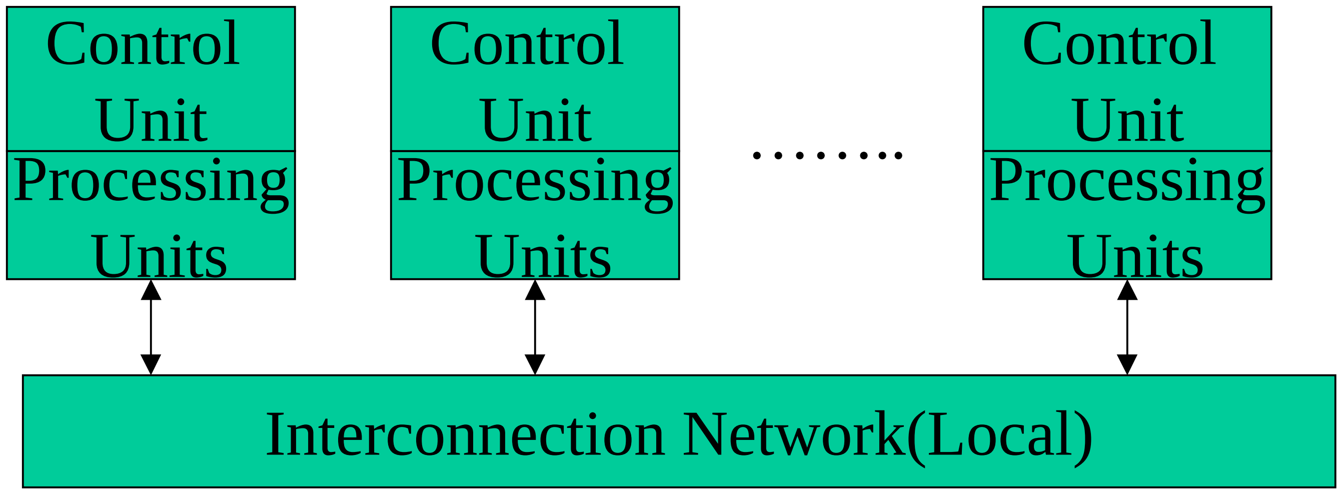
- **Spatial Locality** means that the cells has a local communication interconnection.
- **Temporal Locality** means that the cells transmits the signals from from one cell to other which require at least one unit time delay.
- **Pipelinability** means that the array can achieve a high speed.

# Difference between SIMD array and Systolic array

- Mesh Type SIMD array



- Systolic Array.



- An SIMD array is a synchronous array of PEs under the supervision of one control unit and all PEs receive the same instruction broadcast from the control unit but *operate on different data sets* **from distinct data streams**.
- SIMD array usually loads data into its local memories before starting the computation.
- Systolic arrays usually pipe data from an outside host and also **pipe the results back to the host**.

## Why Systolic Architecture

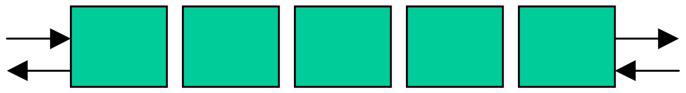
- It can be used for special purpose processing architecture because of
  - 1. Simple and Regular Design.
  - 2. Concurrency and Communication.
  - 3. Balancing Computation with I/O.
- The systolic arrays has a regular  and simple design (i.e)
- They are:
  - cost effective,
  - array is modular (i.e) adjustable to various performance goals ,
  - large number of processors work together,
  - local communication in systolic array is advantageous for communication to be faster.

## Why Systolic Architecture

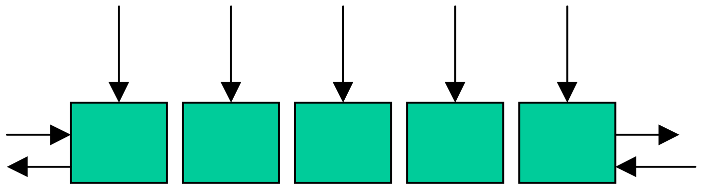
- A systolic array is used as attached array processor,
  - it receives data and o/p the results through an attached host computer,
  - therefore the performance goal of array processor system is a computation rate that balances I/o bandwidth with host.
- With relatively low *bandwidth* of current I/O devices, to achieve  a faster computation rate it is necessary to perform multiple computations per I/O access.
- Systolic arrays does this  efficiently.

# Types of systolic arrays

- Early systolic arrays are linear arrays and one dimensional(1D) or two dimensional I/O(2D).
- Most recently, systolic arrays are implemented as planar array with perimeter I/O to feed data through the boundary.
- Linear array with 1D I/O.
  This configuration is suitable for single I/O.
- Linear array with 2D I/O.
  It allows more control over linear array.

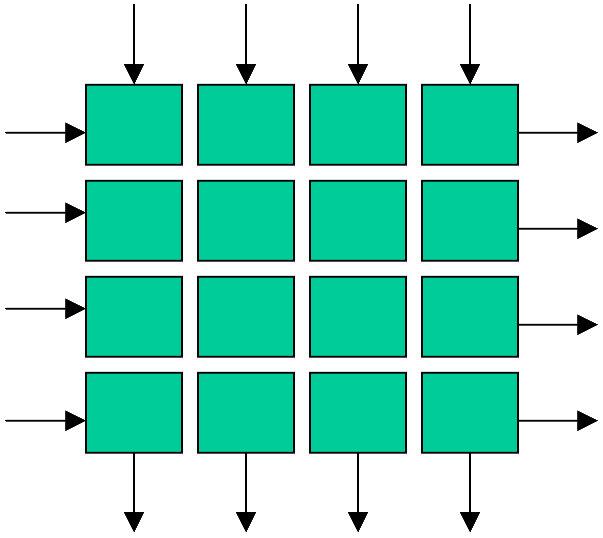


1D Linear Array

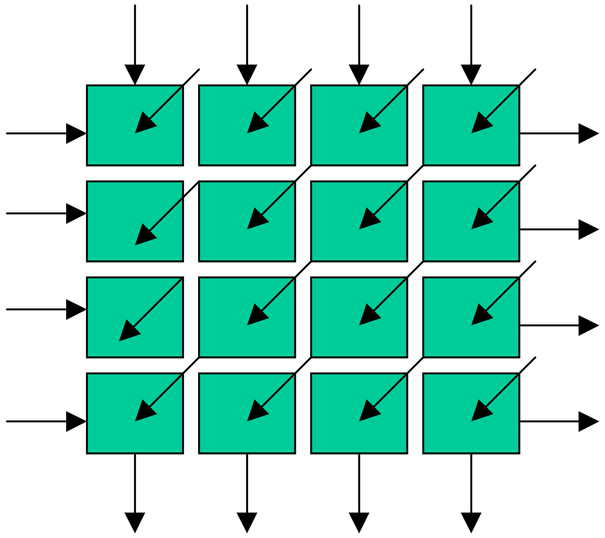


2D Linear Array

- Planar array with perimeter I/O. This configuration allows I/O only through its boundary cells.



- Focal Plane array with 3D I/O. This configuration allows I/O to each systolic cell.



## Variations

- Systolic arrays can be built with variations in:
  - **1**. Connection Topology
    - 2D Meshes
    - hypercubes
  - **2.** Processor capability: ranging through:
    - trivial- just an ALU
    - ALU with several registers
    - Simple CPU- registers, run own program
    - Powerful CPU- local memory also

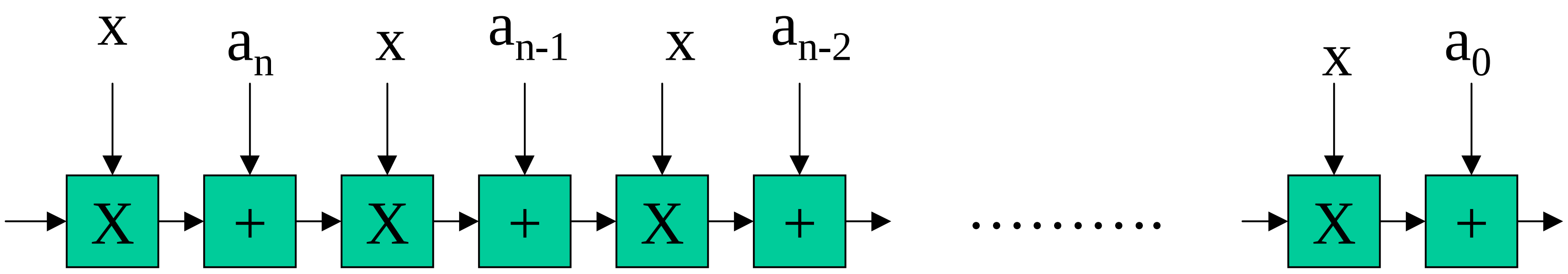
  - **3.** Re-configurable Field programmable Gate Arrays (FPGAs) offer the possibility that re-programmable, re-configurable arrays can be constructed to efficiently compute certain problems.
- In general, FPGA technology is excellent for building small systolic array-style processors.
- Special purpose ALUs can be constructed and linked in a topology to, which the target application maps well.

## Example 1

- Polynomial Evaluation is done by using a Linear array with 2D.
  Expression:
  $Y = ((((a_n x+a_{n-1})*x+a_{n-2})*x+a_{n-3})*x\ldots\ldots a_1)*x + a_0$
- Function of PEs in pairs
  1. Multiply input by x
  2. Pass result to right.
  3. Add $a_j$ to result from left.
  4. Pass result to right.

# Representation using Systolic arrays

- Using systolic array for polynomial evaluation.



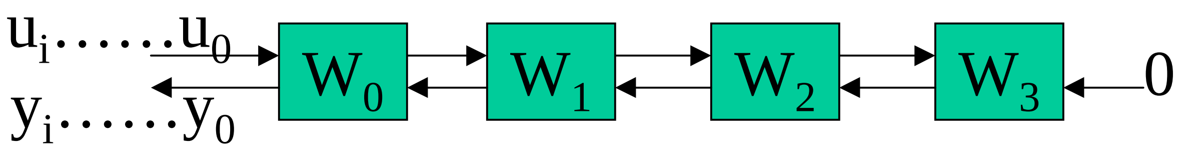
- This array can produce a polynomial on every cycle - after 2n stages.
- This is an example of a deeply pipelined computation-
  – The pipeline has 2n stages.

## Example 2

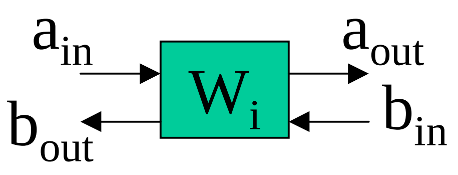
- There are many ways to implement convolution using systolic arrays, one of them is shown:
  - **u(n) :** The input of sequence from left.
  - **w(n) :** The weights preloaded in n PEs.
  - **y(n) :** The sequence from right (Initial value: 0) and having the same speed as u(n).
- In this operation each cell's function is:
  - 1. Multiply the inputs coming from left with weights and output the input received to the next cell.
  - 2. Add the final value to the inputs from right.

# Systolic array for convolution

- Systolic array.



- Each cell operation.



$$a_{out} = a_{in}$$

$$b_{out} = b_{in} + a_{in} * w_i$$

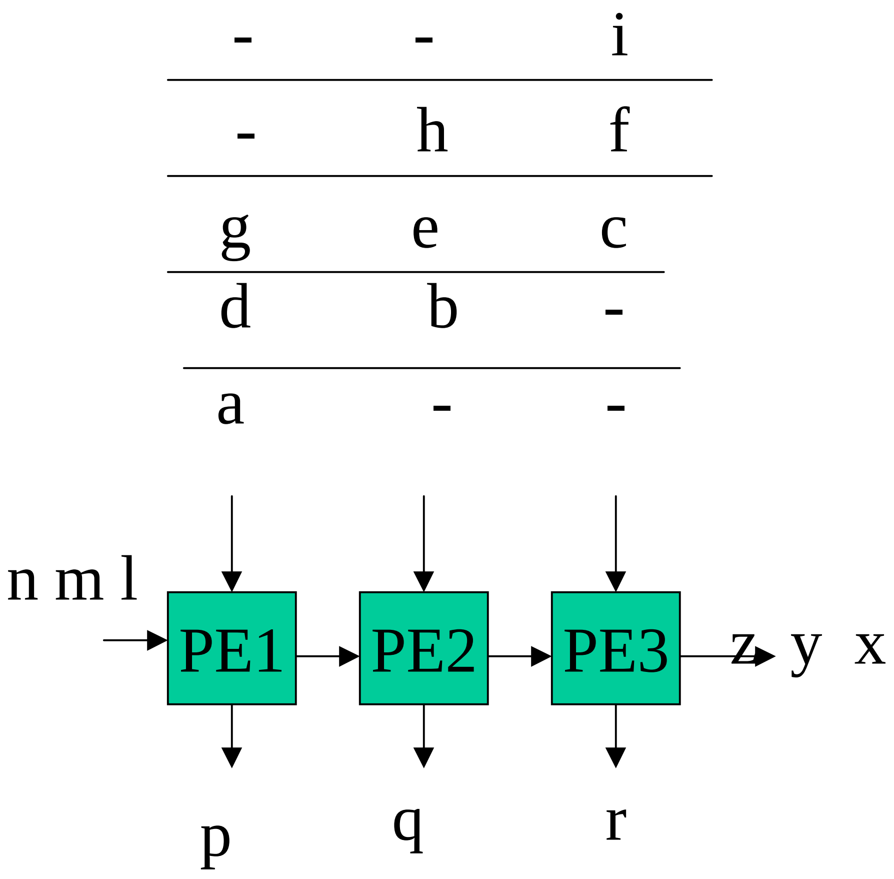
## Example 3
## Matrix Multiplication

- There are many ways to solve a matrix multiplication using systolic arrays, some of the methods are:
  - Triangular Array performing gaussian elimination with neighbor pivoting.
  - Triangular Array performing orthogonal triangularization.
- Simple matrix multiplication methods are shown in examples.

## Example 3

- Matrix Multiplication:
- Each cell's <u>function</u> is:
  - 1. To multiply the top and bottom inputs.
  - 2. Add the left input to the product just obtained.
  - 3. Output the final result to the right.
- Each cell consists of an <u>adder</u> and a *few registers*.
- At time t0 the array receives 1, a, p, q, and r ( The other inputs are all zero).
- At time t1, the array receive m, d, b, p, q, and r ….e.t.c
- The results emerge after 5 steps.

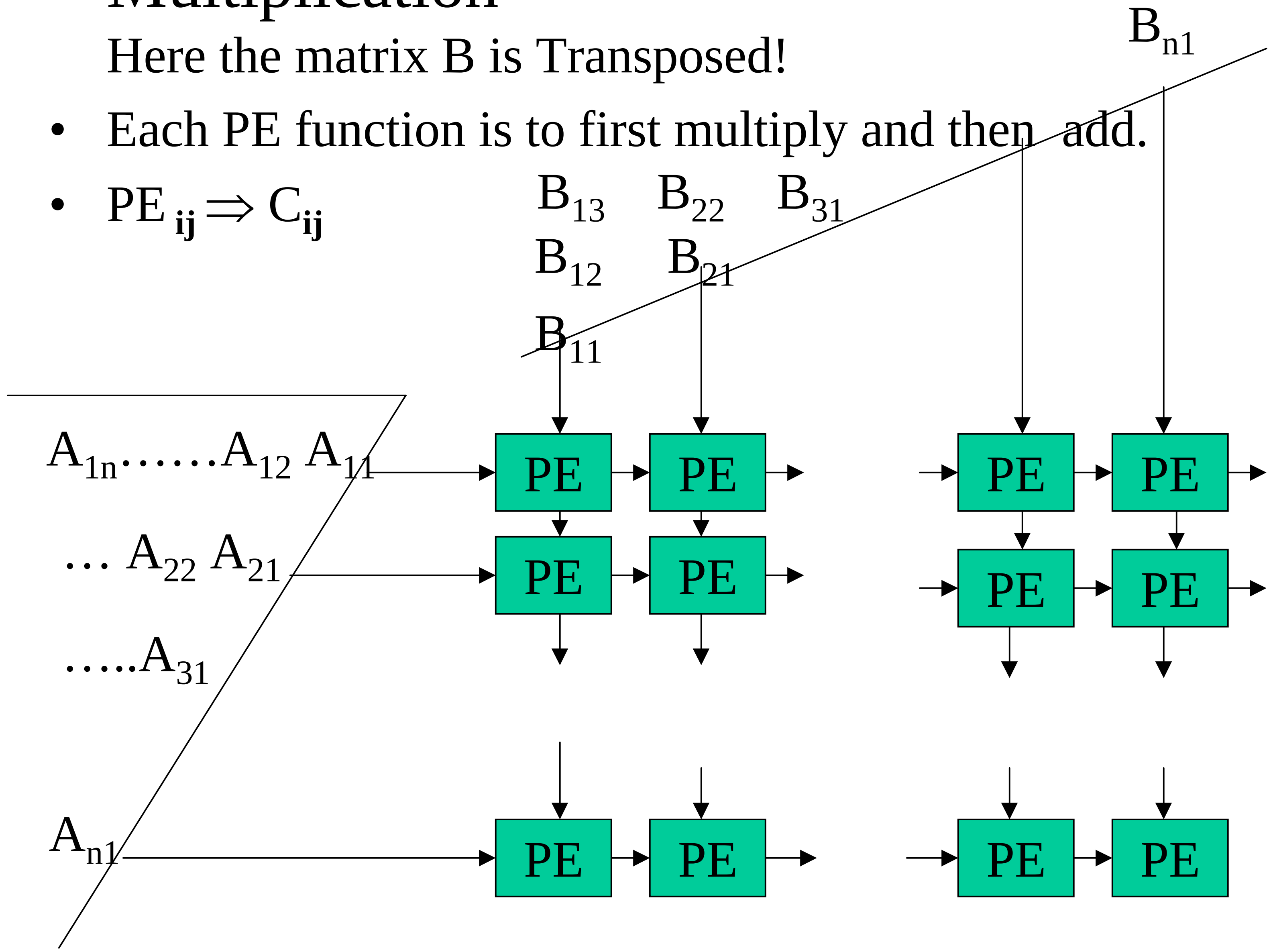
# Matrix Multiplication



$$\begin{pmatrix}X\\Y\\Z\end{pmatrix} = \begin{pmatrix}l\\m\\n\end{pmatrix} + \begin{pmatrix}a&b&c\\d&e&f\\g&h&i\end{pmatrix}\begin{pmatrix}p\\q\\r\end{pmatrix}$$

- Multiplication

  Here the matrix B is Transposed!
- Each PE function is to first multiply and then add.
- PE $_{\mathbf{ij}} \Rightarrow C_{\mathbf{ij}}$



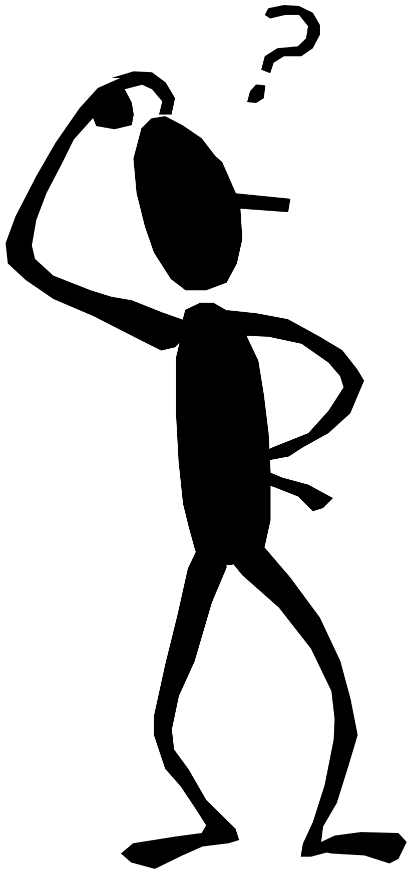
## Advantages of Systolic arrays

- Advantages of systolic arrays are:
  - 1. Regularity and modular design(Perfect for VLSI implementation).
  - 2. Local interconnections(Implements algorithms locality).
  - 3. High degree of pipelining.
  - 4. Highly synchronized multiprocessing.
  - 5. Simple I/O subsystem.
  - 6. Very efficient implementation for great variety of algorithms.
  - 7. High speed and Low cost.
  - 8. Elimination of global broadcasting and modular expansibility.

## Disadvantages of systolic arrays

- The main disadvantages of systolic arrays are:
  - 1. Global synchronization limits due to signal delays.
  - 2. High bandwidth requirements both for periphery(RAM) and between PEs.
  - 3. Poor run-time fault tolerance due to lack of interconnection protocol.

## Parallel overhead.



- Running time for a program running on several processors including an allowance for parallel overhead compared with the ideal running time.
- There is often a point beyond which adding further processors doesn’t result in further efficiency.
- There may also be a point beyond which adding further processors results in slower execution.

## Applications Of Systolic Arrays

- The various applications of systolic arrays are:
  1. Matrix Inversion and Decomposition.
  2. Polynomial Evaluation.
  3. Convolution.
  4. Systolic arrays for matrix multiplication.
  5. Image Processing.
  6. Systolic lattice filters used for speech and seismic signal processing.
  7. Artificial neural network.

# Sources

Syeda Mohsina Afroze

No#990-00-4223

Advanced Logic Synthesis

ECE 572