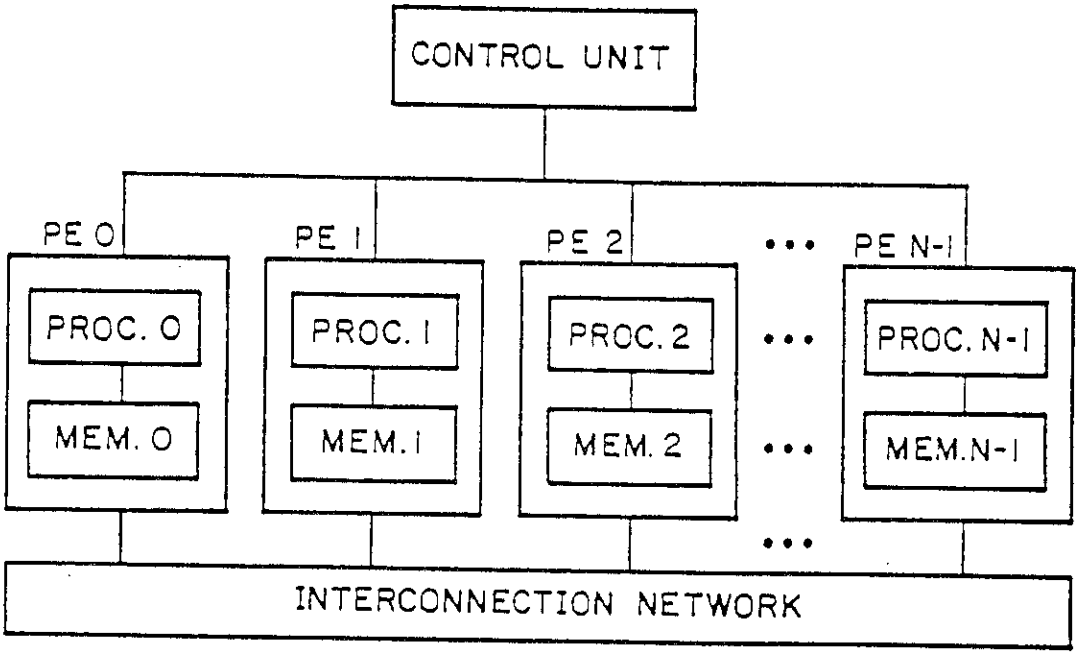
SIMD -

single instruction stream
multiple data stream

Control Unit broadcasts instructions to processors, all active processors execute simultaneously on own data.



PE (processing element) — processor/memory pair

PE-to-PE configuration

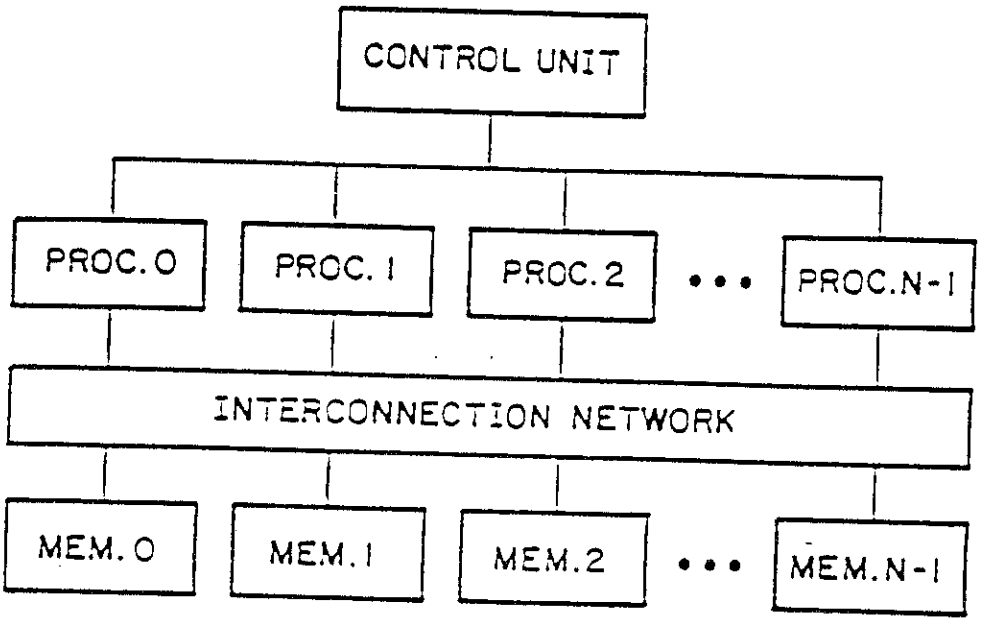
network unidirectional ← PEs send and receive

Ex. Illiac IV, MPP

SIMD — alternative organization

processor-to-memory organization

network bidirectional



processors communicate through memories

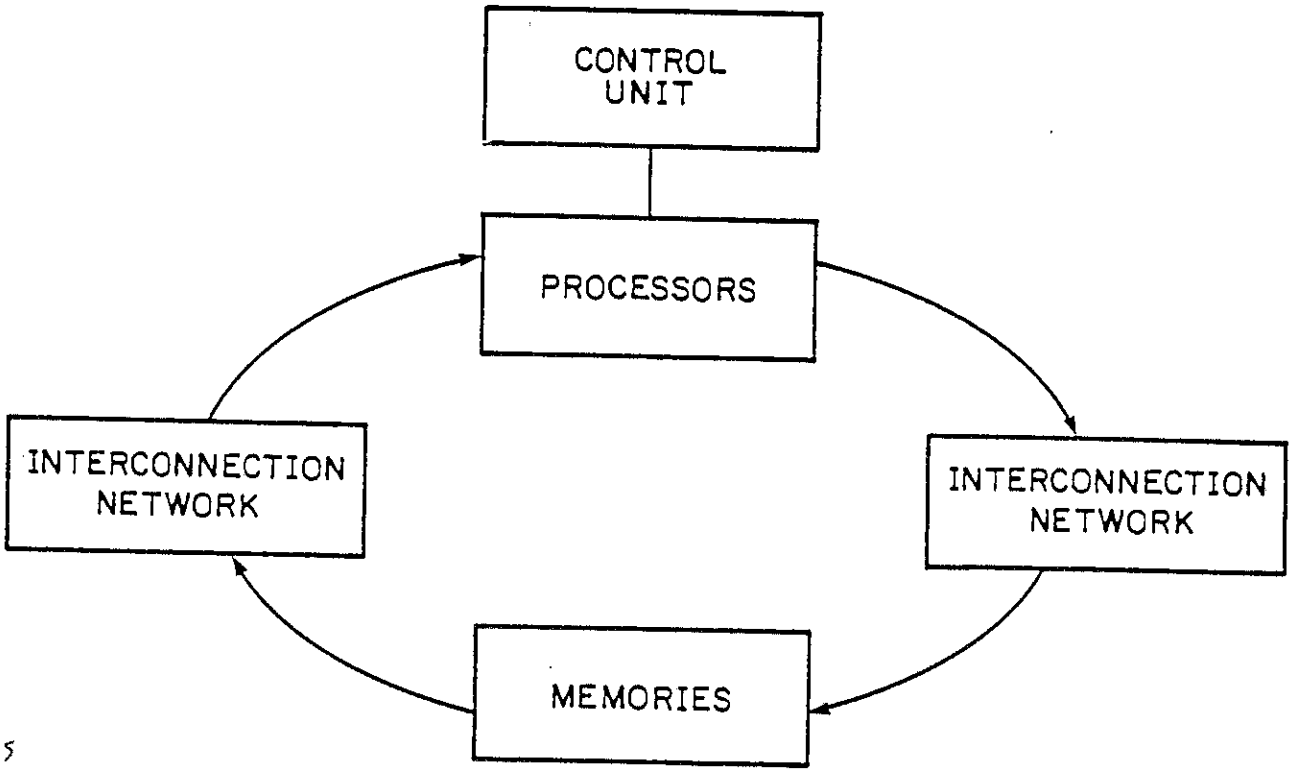
also called shared memory machine

Ex. TRAC

SIMD: single instruction stream - multiple data stream

Variations:

proposed for BSP SIMD machine



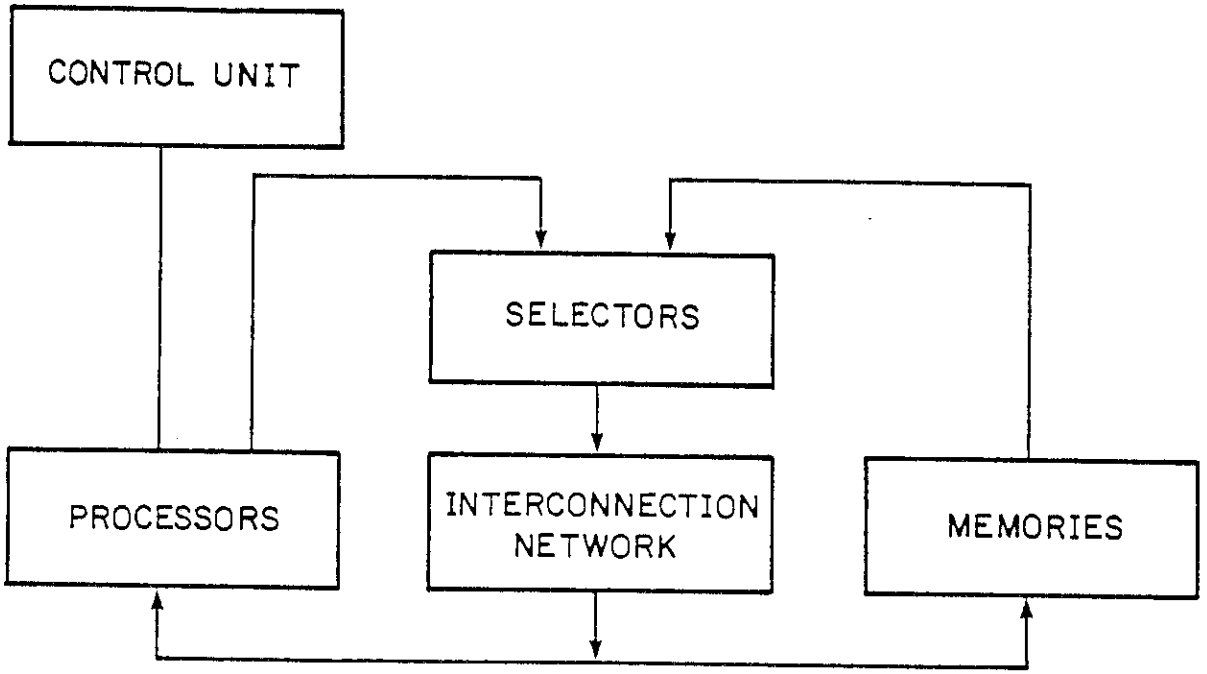
16 PEs

17 memories

relatively prime # of memories

Variations:

STARAN SIMD Machine

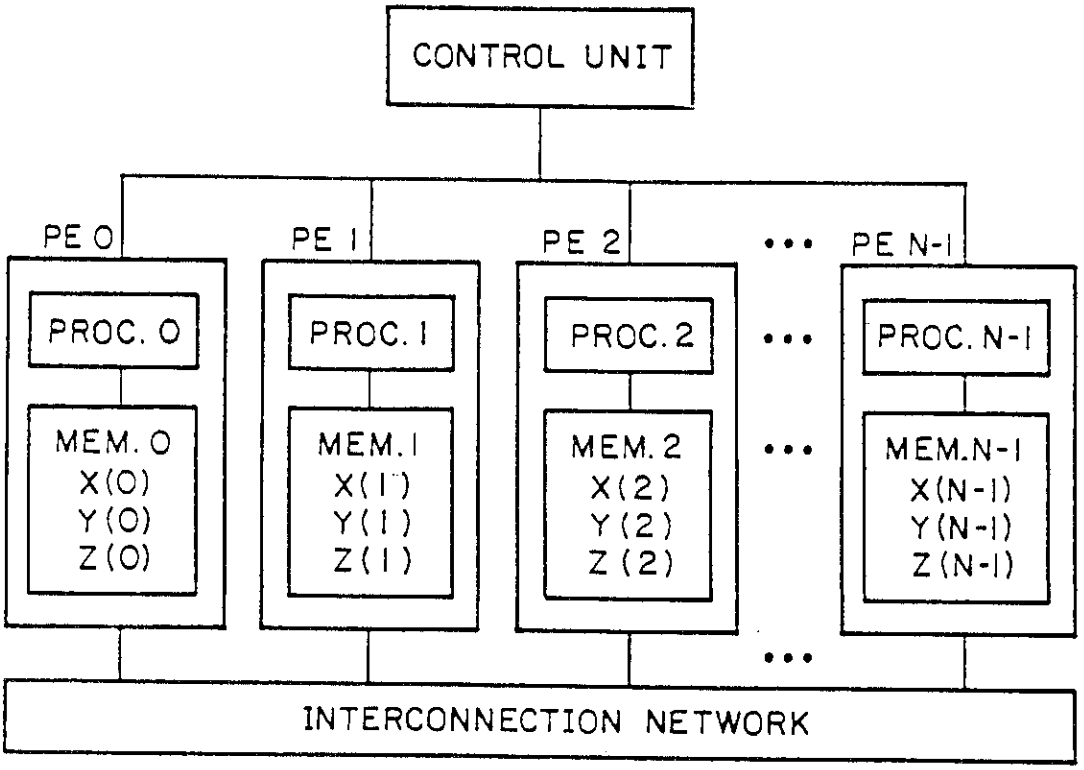


Unidirectional network with multiple modes

PE-to-PE used throughout book -
but results applicable to
other configurations

Ex. SIMD algorithm

x,y,z N element vectors



x(i), y(i), z(i) in PE_i

Serial:

for i = 0 to N−1 do z(i) ← x(i) + y(i)

N steps on serial machine

SIMD: z ← x+y

1 step on SIMD machine

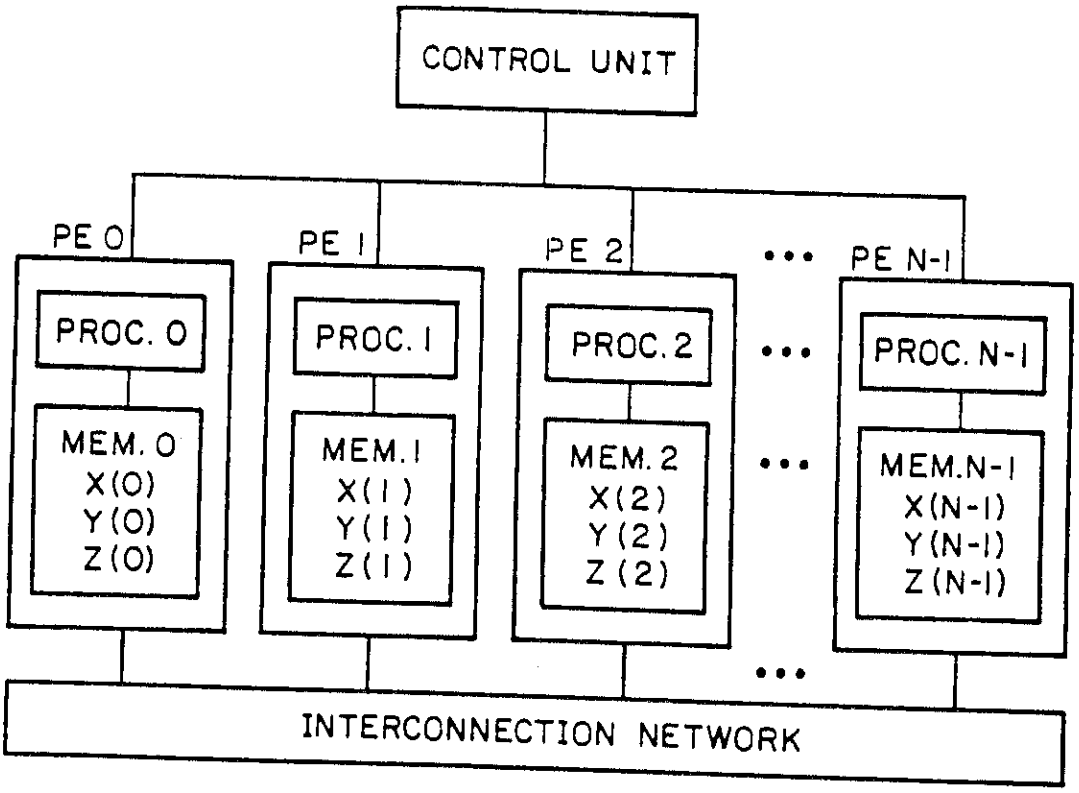
Factor of N speedup — best possible

BUYER BEWARE!

"Factor of N Speedup"

comparing one processor of type X
to N processors of type X

may not be factor of N speedup if
one Cray versus N IBM PCs



Serial:

for i = 1 to N−1 do
    z(i) ← x(i) + y(i−1)
z(0) ← x(0)
    N steps on serial machine

SIMD machine:

1. y(i−1) from PE i−1 to PE i, $1 \leq i < N$ (simultaneously)
2. in PE i z(i) ← x(i) + y(i−1) (PE 0 disabled)
3. in PE 0 z(0) ← x(0) (all other PEs disabled)

Overhead of parallelism: actions needed for parallel execution not needed for serial execution

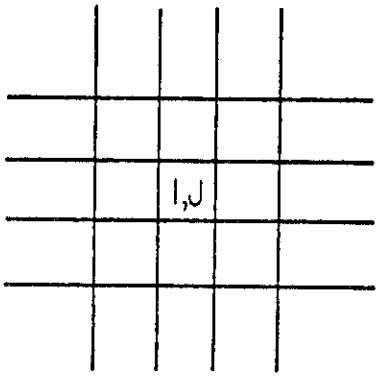
If overhead $\neq 0$, not factor of N speedup

Ex. network use in step 1
    masking—enable/disable PEs, steps 2 and 3

Ex. SIMD algorithm

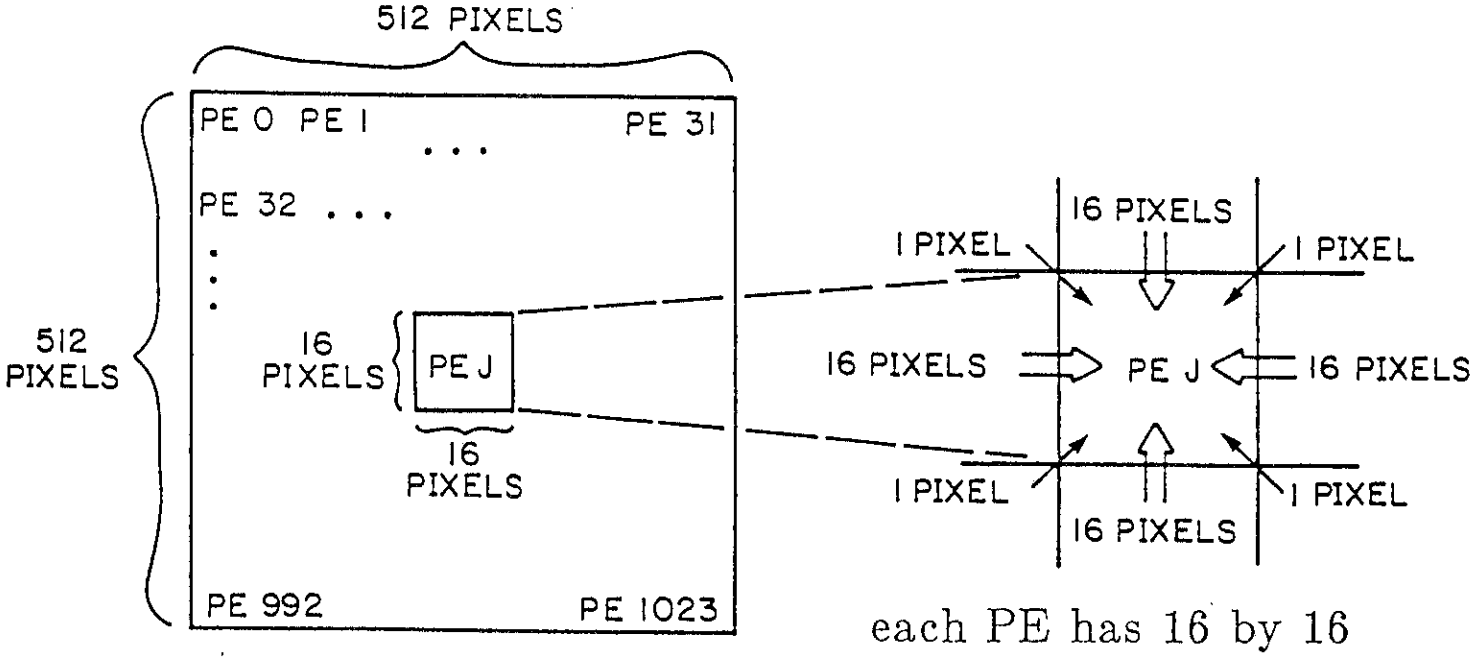
Smoothing an Image
- used for noise reduction

Image - array of pixels
each pixel has value 0 to 255
"gray level"

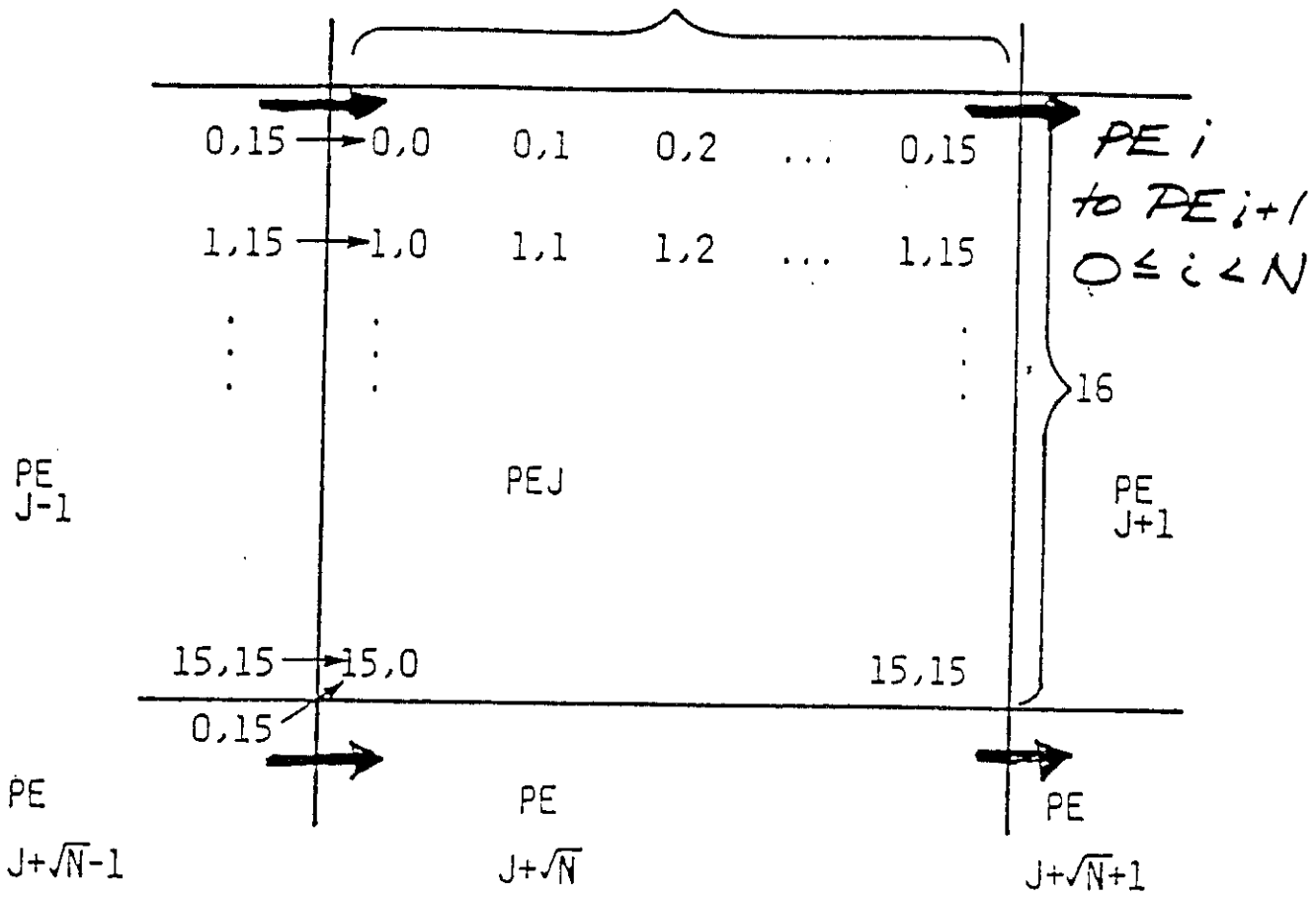


$$A'(I,J) = (A(I,J) + A(I-1,J) + A(I+1,J) + A(I,J-1) + A(I,J+1) + A(I-1,J-1) + A(I-1,J+1) + A(I+1,J+1) + A(I+1,J-1))/9$$

512 by 512 image,
1024 PE's logically arranged 32 by 32



each PE has 16 by 16 subimage



Serial: $1/9 \sum$ (X + 8 Neighbors) 512 × 512 Image

$512 * 512 = 262{,}144$ Smoothing Ops.

SIMD: N = 1024 512 × 512 Image

$1/9 \sum$ (X + 8 Neighbors)

$16 * 16 = 256$ Smoothing Ops.

Transfers

$4 * 16 + 4 = 68$

Can Overlap Transfers and Smoothing Operations

Model of SIMD Machine

- PE
- Control Unit Instructions
- PE Instructions
- Masking Schemes
- Interconnection Networks

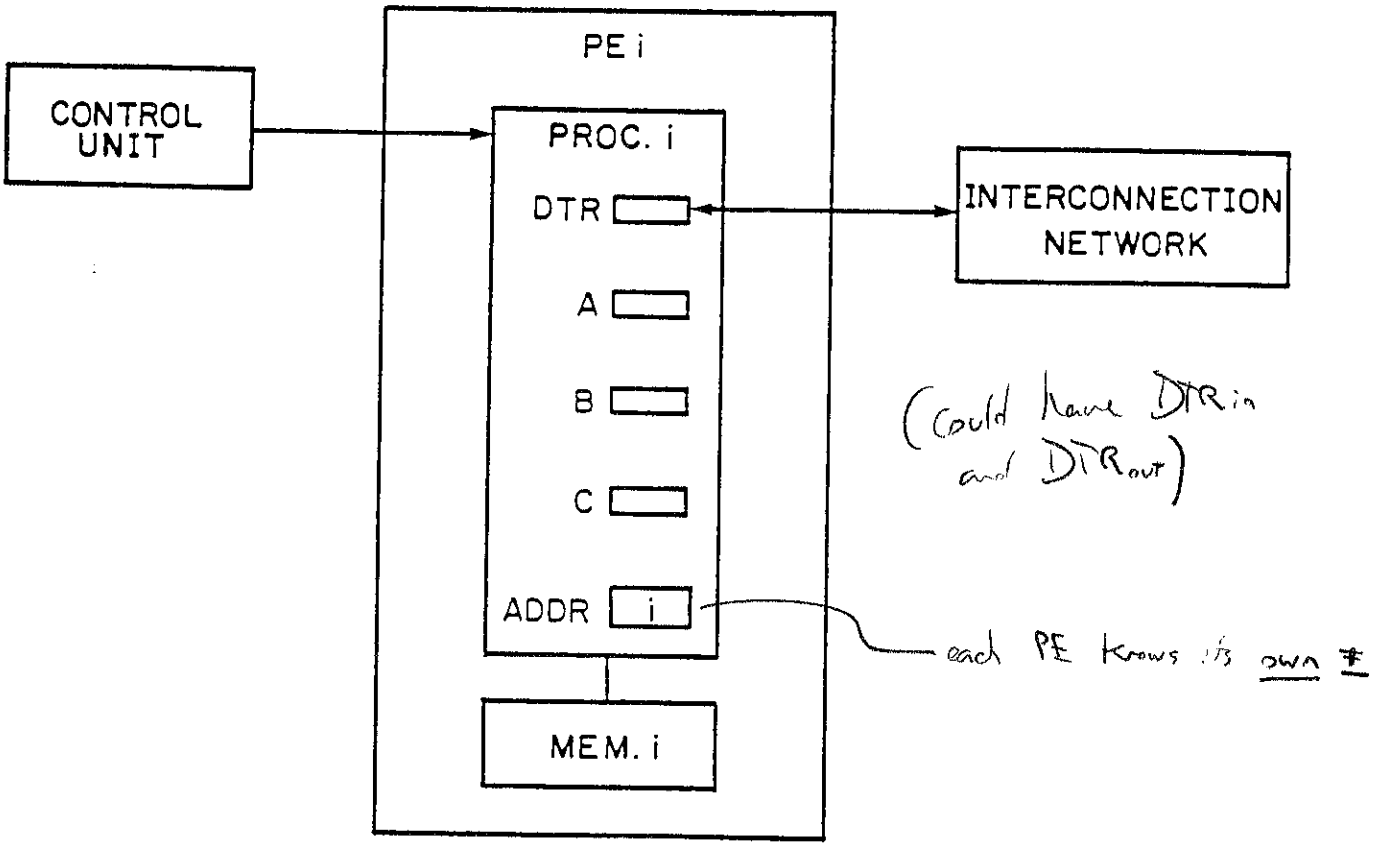
PE

A,B,C — processor registers

DTR — data transfer register

PEs numbered (addressed) 0 to N–1

register ADDR = i (PE i address)



PE active/inactive — controlled by masking scheme

Control Unit (CU) Instructions

- stores SIMD programs
- executes control flow instructions
  ex. for i=0 until 15 do ...
- broadcasts PE instructions -
  all active PEs execute

PE Instructions

- operations PE performs on own data
- include: move data among own registers

Masking Schemes

- enable/disable PEs
- can use several methods
- N-bit vector
  - — bit i=1, PE i active
  - — bit i=0, PE i inactive
  - — maximum flexibility
  - — impractical for large N
- PE address masks
  - — m-position mask ($N=2^m$)
  - — each position 0, 1, or X (don't care)
  - — match mask → PE active
  - — Examples:
    $[X^2 01] = [XX01]$ PEs 1,5,9,13

    $[X^{m-1}0]$ even PEs

    $[0^{m-i}X^i]$ first 2^i PEs
  - — usage (one way)
    $A \leftarrow -A\ [X^{m-2}0^2]$
    complement A in all PEs multiple of 4

Masking Schemes

- Data Conditional Masks

  where (data conditional — PE data)

  do ...

  elsewhere ...

  — each does "do" or "elsewhere" (not both)

  — PE evaluates conditional and sets internal flag to activate for just "do" or "elsewhere"

  — ex. where $(x > 0)$

  do $y \leftarrow x$

  elsewhere $y \leftarrow -x$

  $y = |x|$ in all PEs

  — can be nested

  — can simulate PE address masks

Interconnection Network

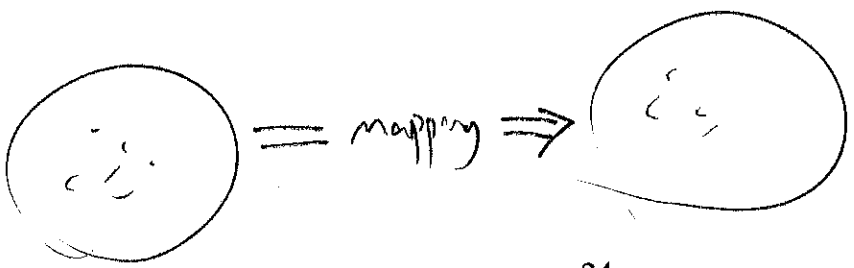
- set of interconnection functions
- interconnection function (i.f.) — bijection (permutation) on PE addresses
- execute i.f. f: PE i → PE f(i) ∀i simultaneously,
  PE i active
- multiple transfers may be necessary

  if have f(P) = P+1 mod N
  to move from PE i to PE i+2 mod N
  do f twice
- mathematically: f maps address i to address f(i)
  "maps address x to address y" =
  "causes PE x to send data to PE y"
- i.f. part of PE instructions

Bijection

Domain

Co-Domain



function is one-to-one if each element of codomain is mapped to by at most one element of the domain

ex: $f(x) = x+1$
domain codomain

$f(x) = |x|$ then NOT one-to-one

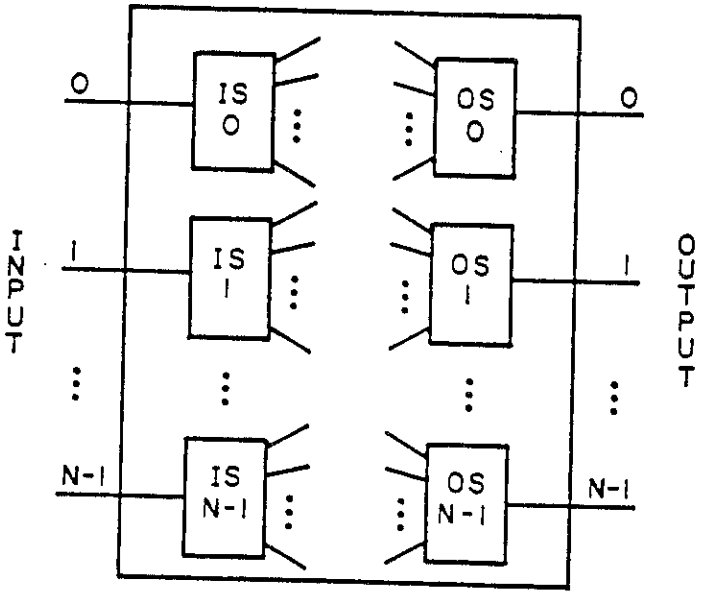
Formal Model of SIMD Machine

(N, C, I, M, F)

- N = # PEs ($N = 2^m$)
- C = CU instructions (loop control)
- I = PE instructions (move data among registers)
- M = set of masking schemes (PE address masks, data conditional)
- F = set of i.f.s = interconnection network (will vary)

The Interconnection Networks

- Single-stage network model used for theoretical development (multistage later)
- Conceptual view



IS Input Selector

OS Output Selector

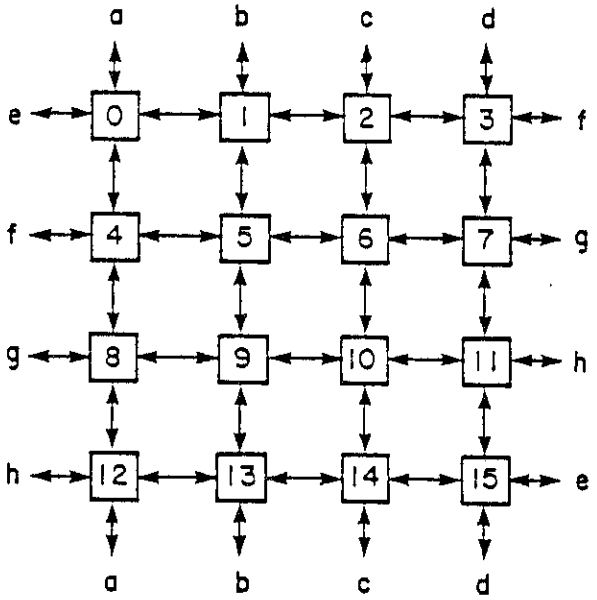
Illiac - 4 i.f.s (where $n = \sqrt{N}$):

$$\text{Illiac}_{+1}(P) = P+1 \bmod N$$

$$\text{Illiac}_{-1}(P) = P-1 \bmod N$$

$$\text{Illiac}_{+n}(P) = P+n \bmod N$$

$$\text{Illiac}_{-n}(P) = P-n \bmod N$$



Ex. $\text{Illiac}_{-n}(9) = 5$ for $N = 16$

four nearest neighbor

IS i sends to OS i+1, i−1, i+n, i−n

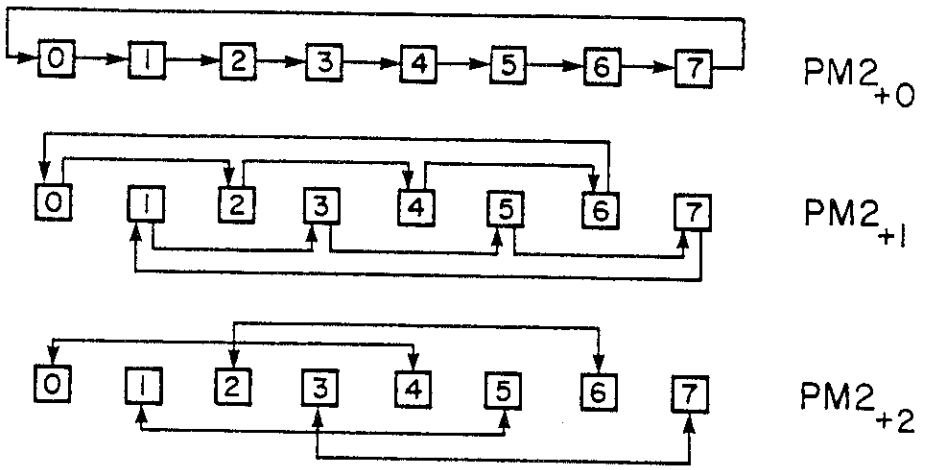
OS j receives from IS j−1, j+1, j−n, j+n

PM2I - Plus Minus 2^i - 2m i.f.s

$$PM2_{+i}(P) = P+2^i \bmod N$$

$$PM2_{-i}(P) = P-2^i \bmod N$$

$$0 \leq i < m$$



Ex. $PM2_{+1}(3) = 5$ for $N > 4$

$PM2_{+(m-1)} = PM2_{-(m-1)}$: $P+2^{m-1} = P-2^{m-1} \bmod N$

superset of Illiac

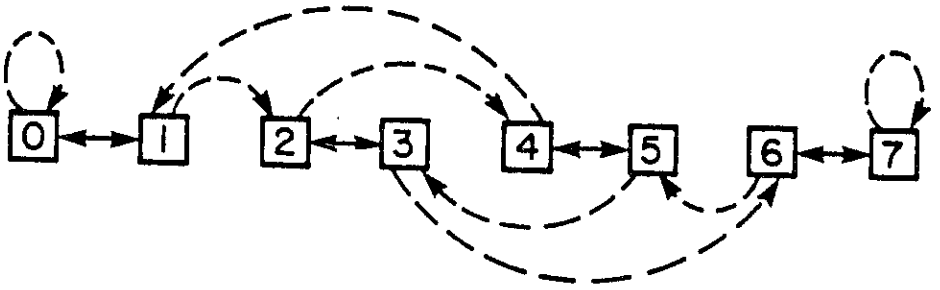
IS j sends to OS $j+2^i$ and $j-2^i$, $0 \leq i < m$

OS j receives from IS $j+2^i$ and $j-2^i$, $0 \leq i < m$

Shuffle-Exchange - 2 i.f.s

$\text{shuffle}(p_{m-1}\cdots p_1 p_0) = p_{m-2}p_{m-3}\cdots p_1 p_0 p_{m-1}$

$\text{exchange}(p_{m-1}\cdots p_1 p_0) = p_{m-1}\cdots p_1 \bar{p}_0$



Ex. for $N = 8$

$\text{shuffle}(5) = 3$

$\text{exchange}(0) = 1$

dashed line - shuffle

solid line - exchange

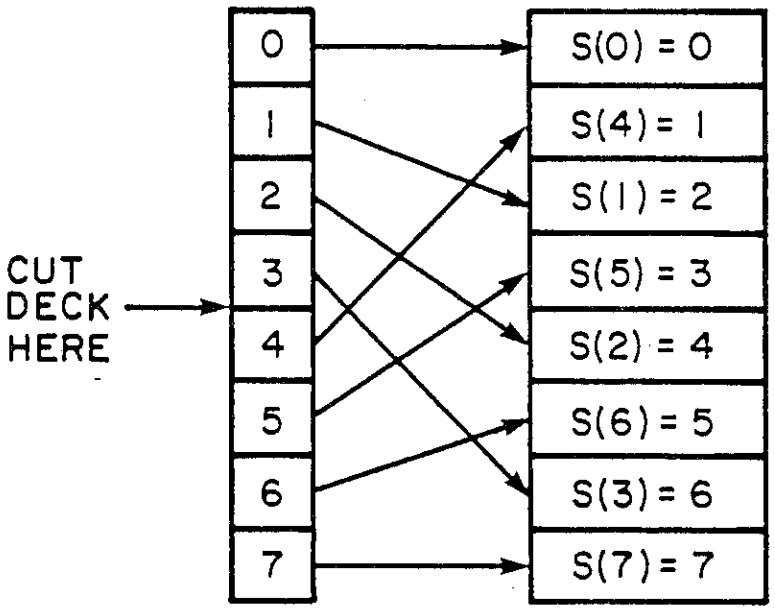
IS i sends to shuffle(i) and exchange(i)

OS j receives from shuffle^{-1}(j) and exchange(j)

$\text{shuffle}^{-1}(p_{m-1}\cdots p_1 p_0) = p_0 p_{m-1}\cdots p_2 p_1$

shuffle $(p_{m-1}\cdots p_1 p_0) = p_{m-2}p_{m-3}\cdots p_1 p_0 p_{m-1}$

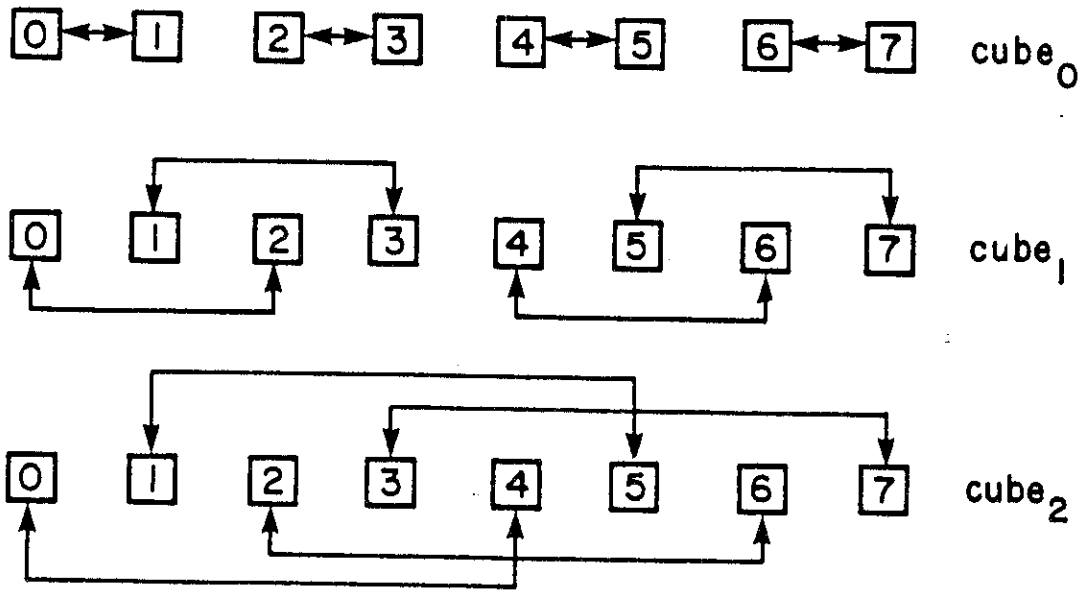
also called "perfect shuffle"



Cube - m i.f.s

$$\text{cube}_i(p_{m-1}\cdots p_1 p_0) = p_{m-1}\cdots p_{i+1}\bar{p}_i p_{i-1}\cdots p_0$$

$$0 \leq i < m$$

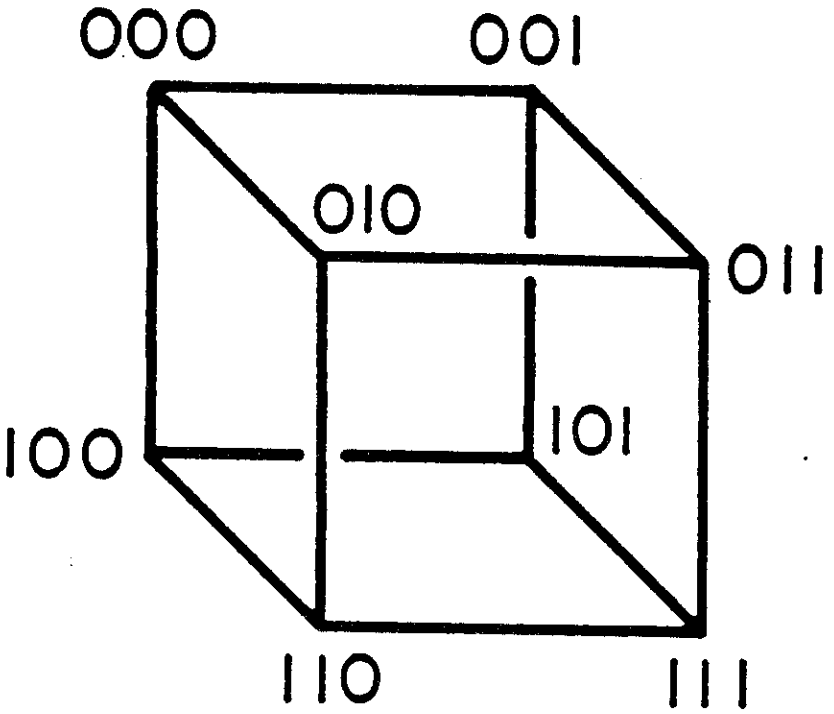


Ex. $\text{cube}_1(3) = 1$ for $N = 8$

IS j sends to $\text{cube}_i(j)$, $0 \leq i < m$

OS j receives from $\text{cube}_i(j)$, $0 \leq i < m$

Reason why called "Cube"



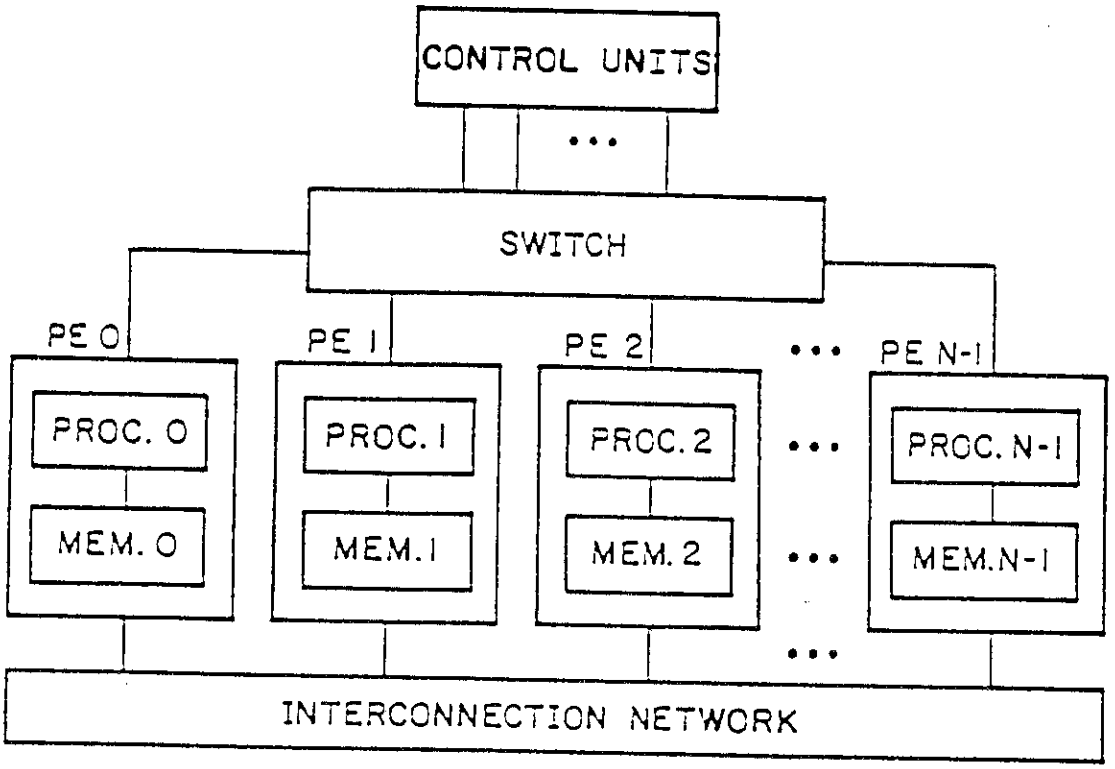
$cube_0$ horizontal

$cube_1$ diagonal

$cube_2$ vertical

Multiple - SIMD Machines

Can be dynamically reconfigured to operate as one or more independent virtual SIMD machines of various sizes



Advantages over SIMD:

1. fault detection
2. fault tolerance
3. multiple simultaneous users
4. program development
5. efficiency
6. subtask parallelism

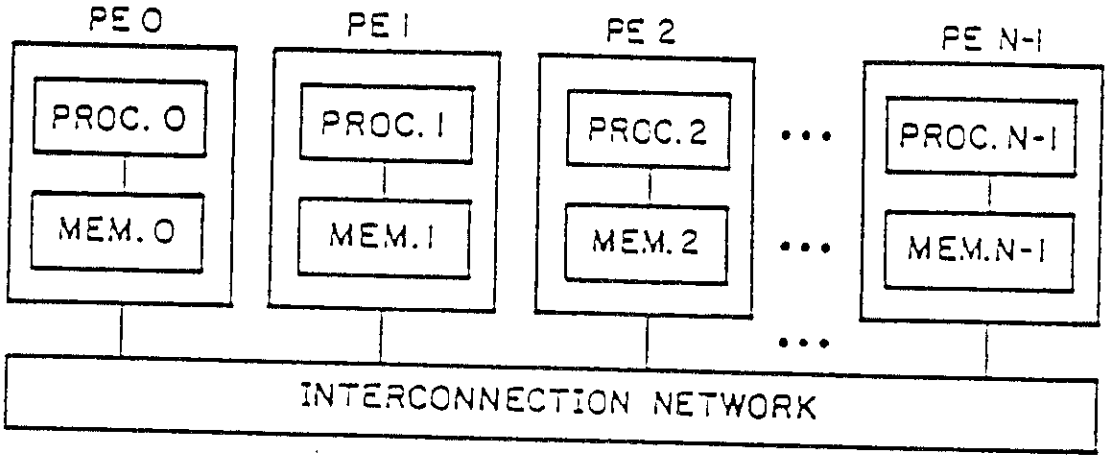
Ex. original design for Illiac IV, Connection Machine

MIMD -

Multiple Instruction stream

Multiple Data stream

each processor has own instructions and data stream



PE (processing element) — processor/memory pair

PE-to-PE configuration

network unidirectional

Ex. CHoPP, Cosmic Cube

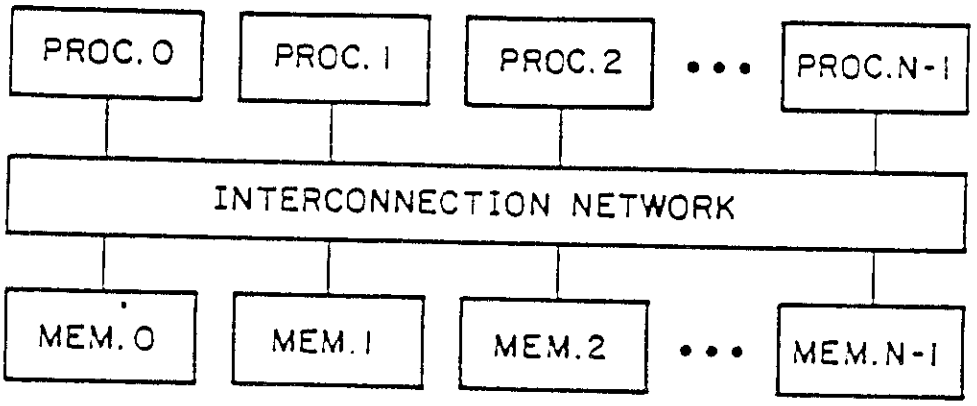
MIMD - PEs operate asynchronously w.r.t. one another
different programs

SIMD - PEs operate synchronously (lockstep)
single program

MIMD - alternative organization

processor-to-memory organization

network bidirectional



processors communicate through memories

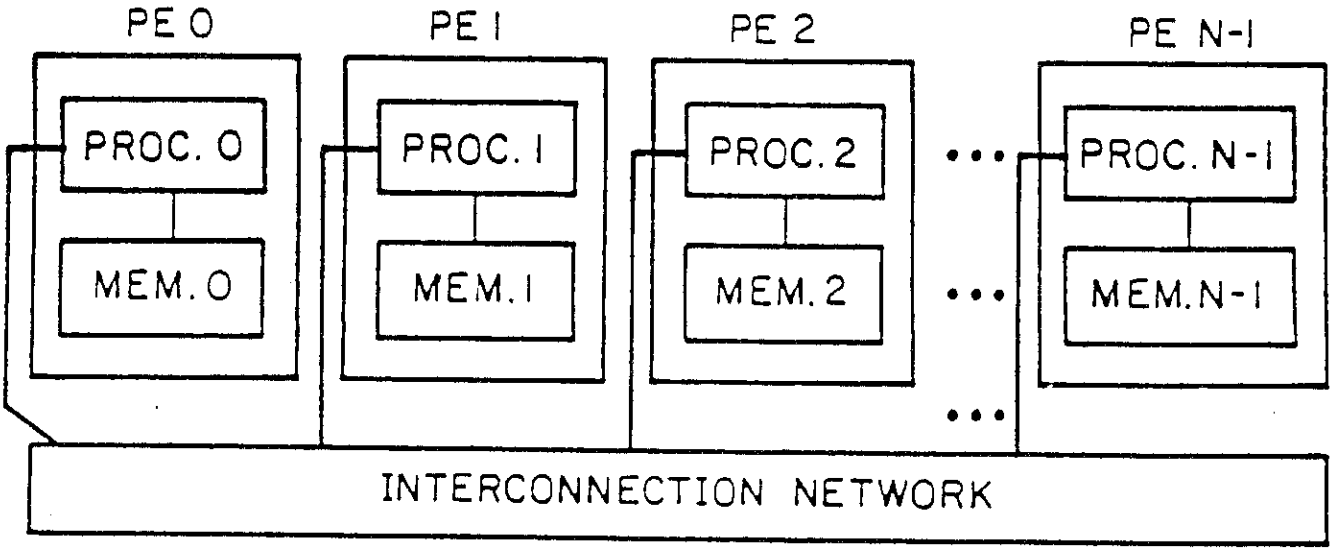
also called shared memory machine

Ex. Ultracomputer (also has local cache),

C.mmp

PE-TO-PE ORGANIZATION

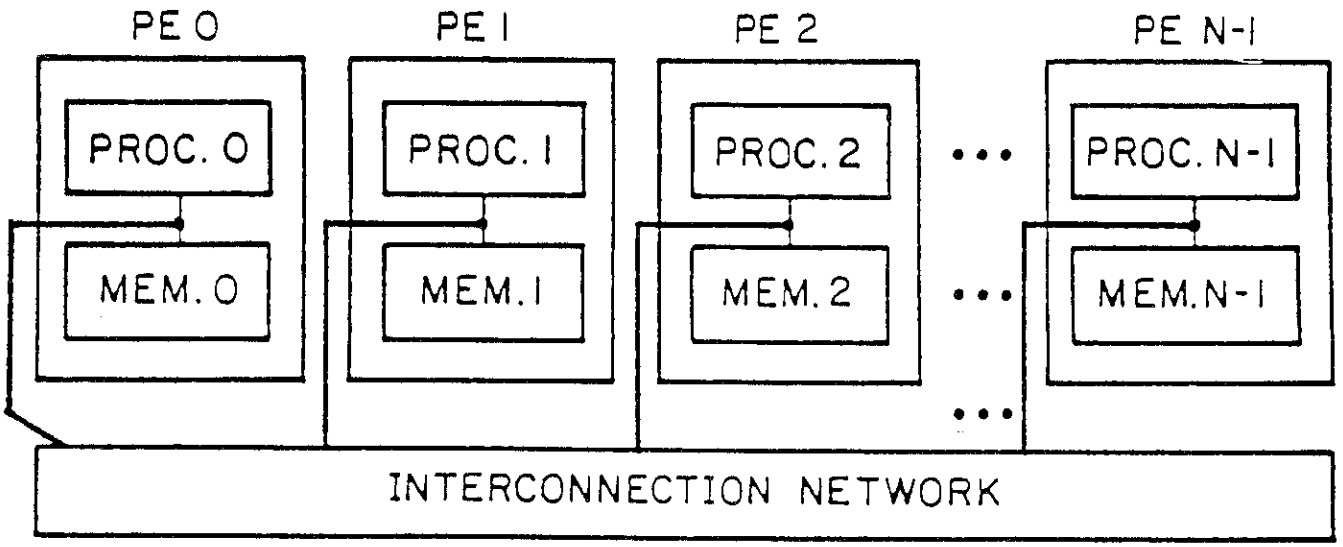
assumes connections are
processor-to-processor



network unidirectional

PE: processing element - proc./mem. pair

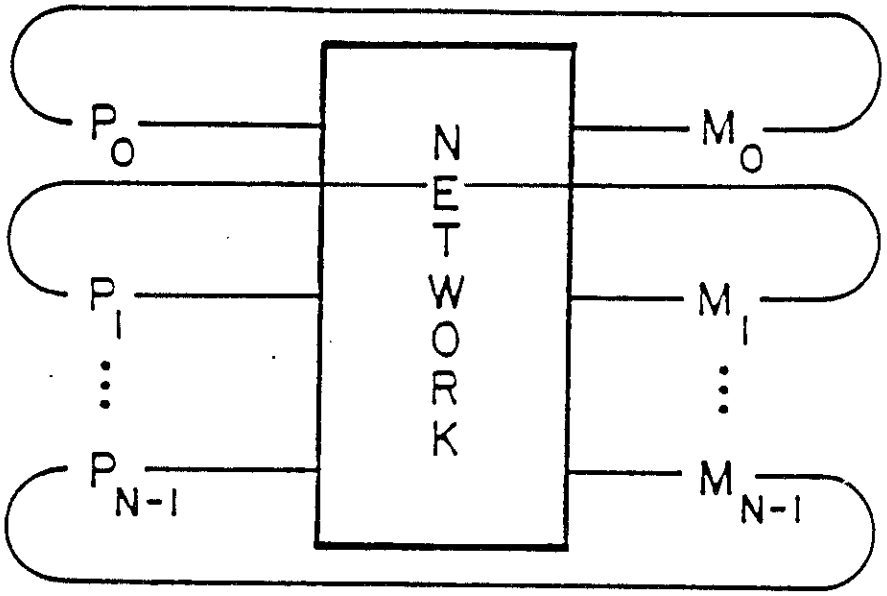
HYBRID OF PE-TO-PE AND PROCESSOR-TO-MEMORY ORGANIZATIONS (BBN Butterfly MIMD Machine)



network unidirectional

PE: processing element - proc./mem. pair

HYBRID OF PE-TO-PE AND PROCESSOR-TO-MEMORY ORGANIZATIONS



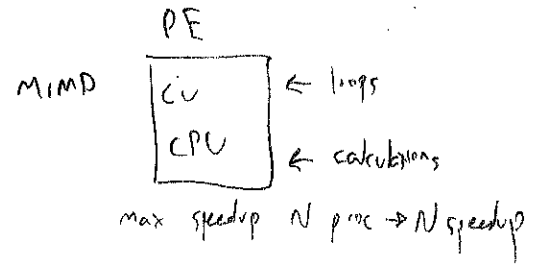
P_i directly connected to M_i
plus P_i connected to one end of network
M_i connected to other end

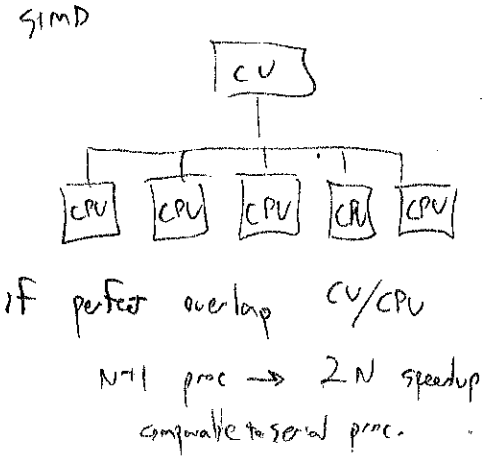
network bidirectional

SIMD vs. MIMD

- flexibility (applicability) - MIMD
- inter-PE transfer overhead - SIMD
- synchronization overhead - SIMD
- overlap loop control - SIMD
- single program - SIMD

"best" mode task dependent





Partitionable SIMD/MIMD Machines

- can be dynamically reconfigured to operate as one or more independent virtual SIMD and/or MIMD machines of various sizes
- structure like multiple-SIMD
- advantages of multiple-SIMD plus choice of SIMD or MIMD modes
- ex. TRAC, PASM