Theory of Multicore Algorithms

Jeremy Kepner and Nadya Bliss

MIT Lincoln Laboratory

HPEC 2008

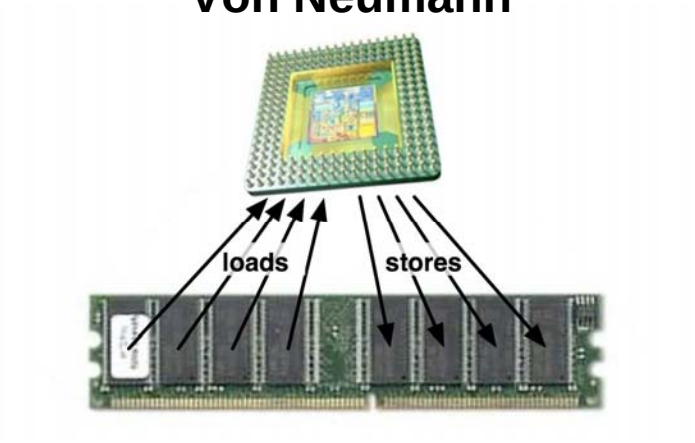
This work is sponsored by the Department of Defense under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the author and are not necessarily endorsed by the United States Government.

Outline

- **Parallel Design**
  - ***Programming Challenge***
  - ***Example Issues***
  - ***Theoretical Approach***
  - ***Integrated Process***
- Distributed Arrays
- Kuck Diagrams
- Hierarchical Arrays
- Tasks and Conduits
- Summary

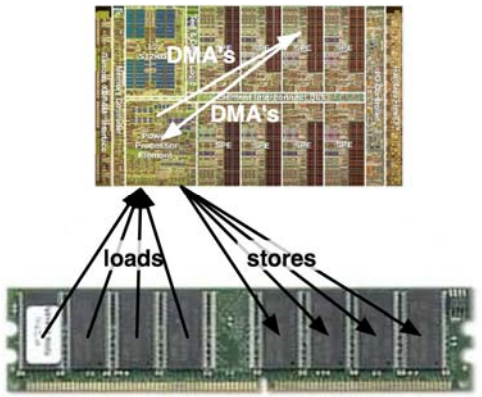
Multicore Programming Challenge

Past Programming Model: Von Neumann



- Great success of Moore's Law era
  - Simple model: load, op, store
  - Many transistors devoted to delivering this model
- Moore's Law is ending
  - Need transistors for performance

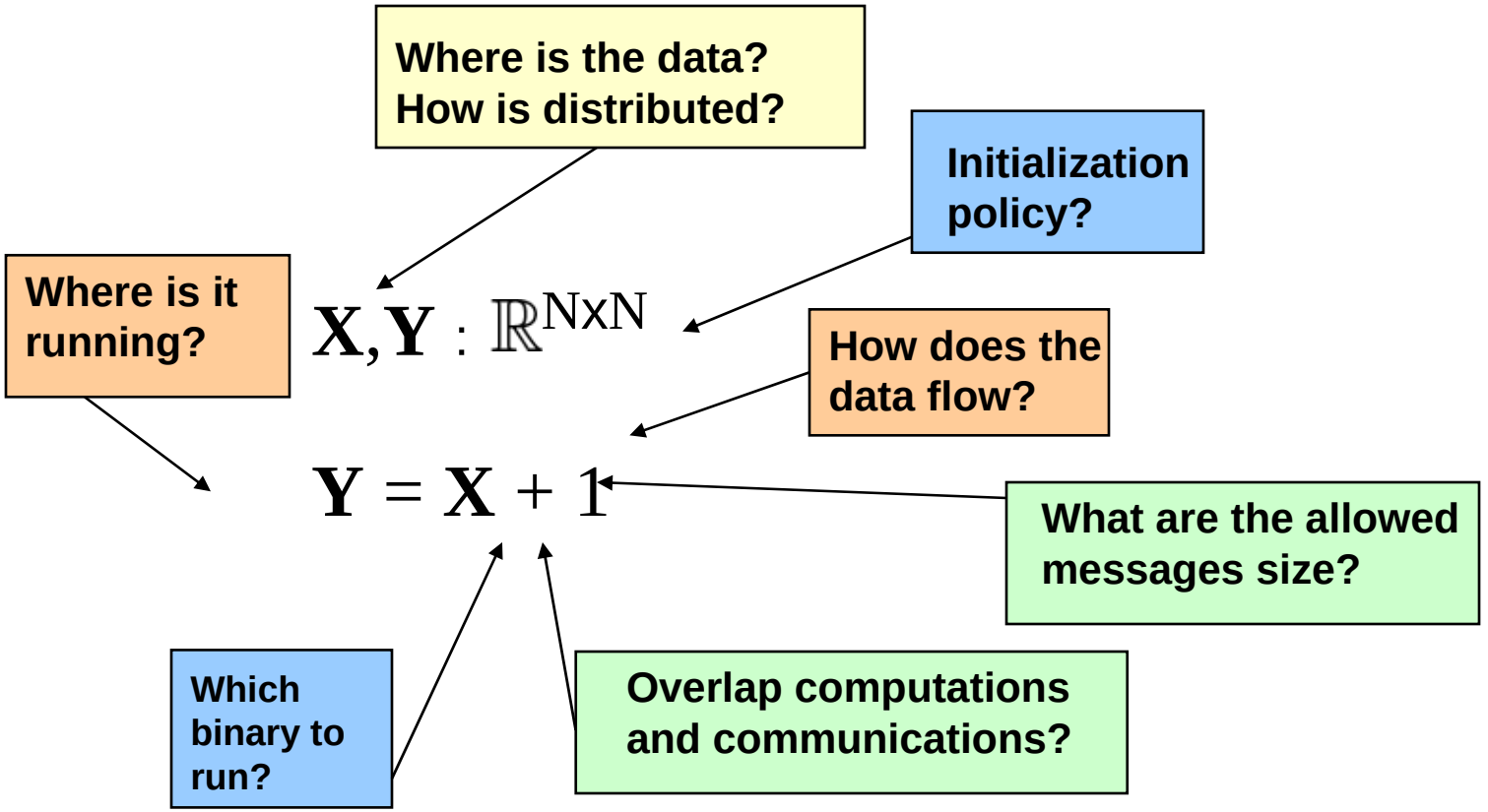
Future Programming Model: ???



- Processor topology includes:
  - Registers, cache, local memory, remote memory, disk
- Cell has *multiple* programming models

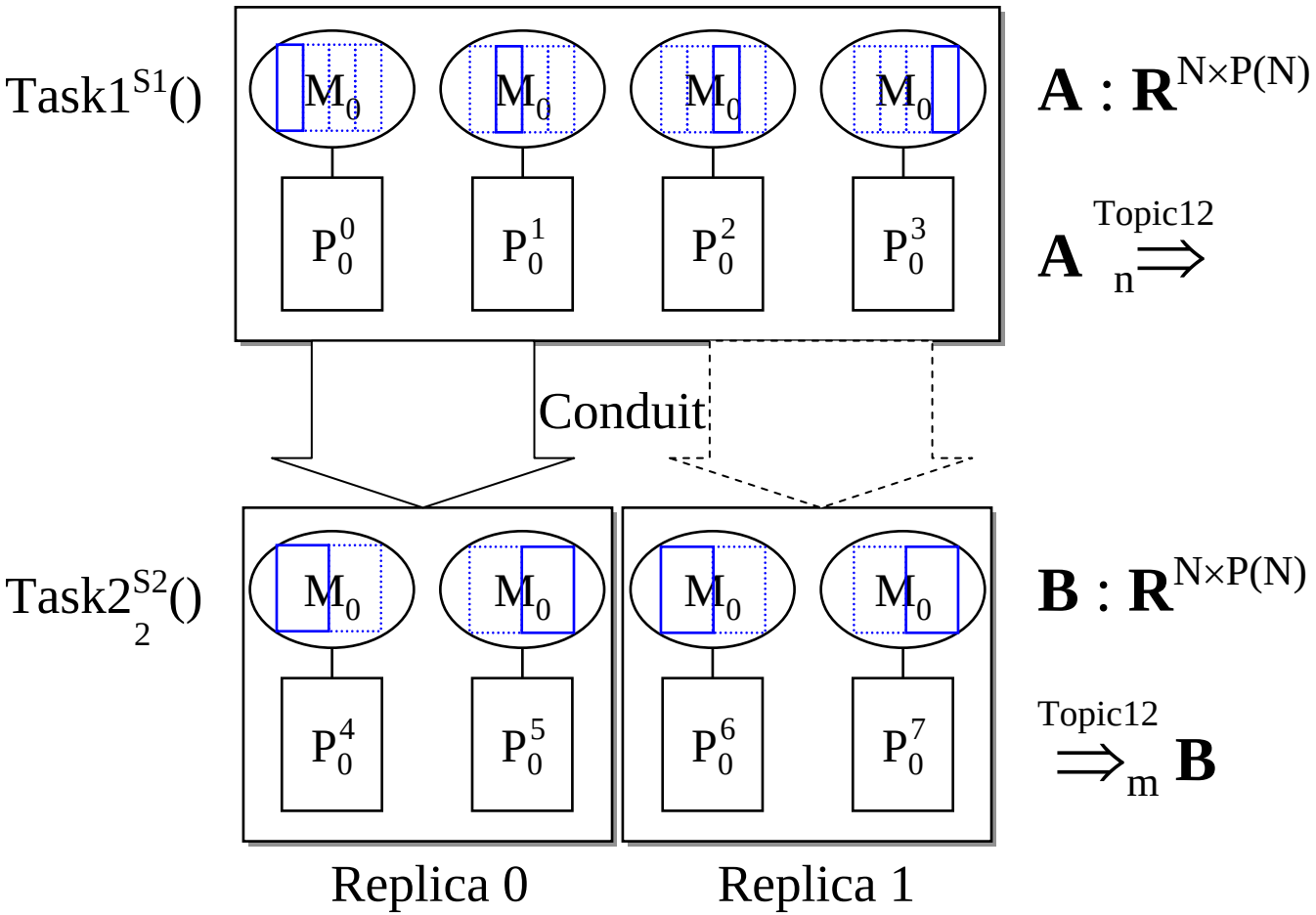
Can we describe how an algorithm is *suppose* to behave on a hierarchical heterogeneous multicore processor?

Example Issues



- A serial algorithm can run a serial processor with relatively little specification
- A hierarchical heterogeneous multicore algorithm requires a lot more information

Theoretical Approach

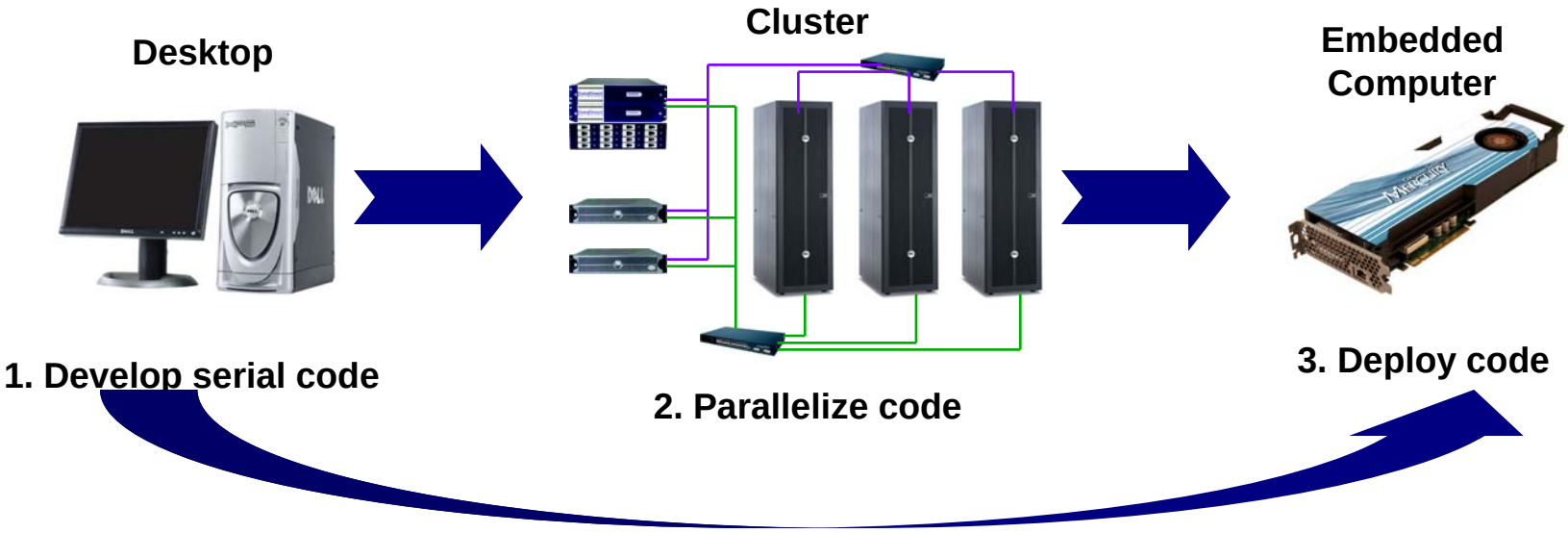


- **Provide notation and diagrams that allow hierarchical heterogeneous multicore algorithms to be specified**

Integrated Development Process

$$\mathbf{X},\mathbf{Y} : \mathbb{R}^{N \times N} \qquad \mathbf{X},\mathbf{Y} : \mathbb{R}^{P(N) \times N} \qquad \mathbf{X},\mathbf{Y} : \mathbb{R}^{P(P(N)) \times N}$$

$$\mathbf{Y} = \mathbf{X} + 1 \qquad \mathbf{Y} = \mathbf{X} + 1 \qquad \mathbf{Y} = \mathbf{X} + 1$$



- **Should naturally support standard parallel embedded software development practices**

Outline

- Parallel Design
- **Distributed Arrays**
  - ***Serial Program***
  - ***Parallel Execution***
  - ***Distributed Arrays***
  - ***Redistribution***
- Kuck Diagrams
- Hierarchical Arrays
- Tasks and Conduits
- Summary

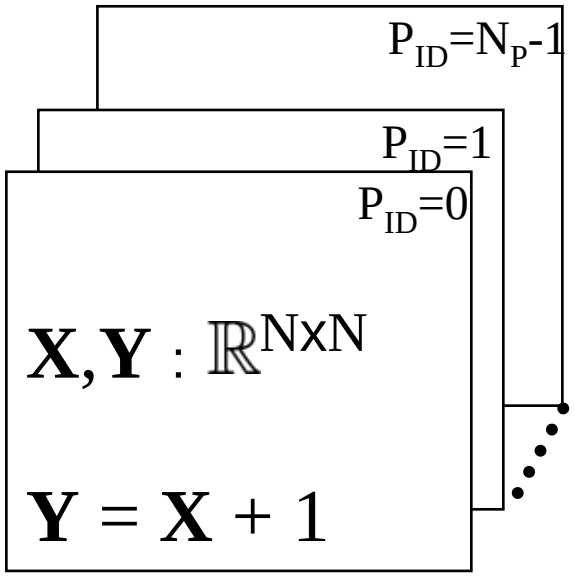
Serial Program

$$\mathbf{X}, \mathbf{Y} : \mathbb{R}^{N \times N}$$

$$\mathbf{Y} = \mathbf{X} + 1$$

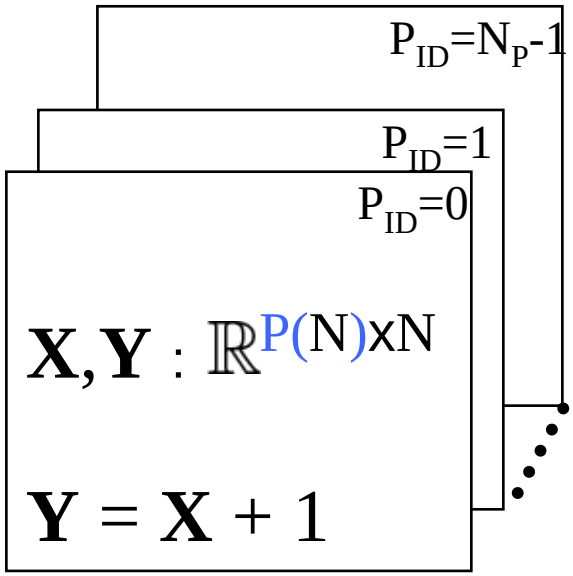
- **Math is the language of algorithms**
- **Allows mathematical expressions to be written concisely**
- **Multi-dimensional arrays are *fundamental* to mathematics**

Parallel Execution



- Run N_P copies of same program
  - Single Program Multiple Data (SPMD)
- Each copy has a unique P_{ID}
- Every array is *replicated* on each copy of the program

Distributed Array Program



- **Use P() notation to make a distributed array**
- **Tells program which dimension to distribute data**
- **Each program implicitly operates on only its own data (owner computes rule)**

Explicitly Local Program

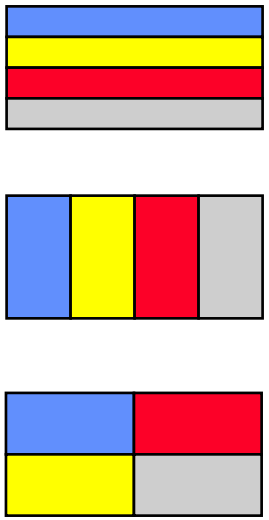
$$\mathbf{X}, \mathbf{Y} : \mathbb{R}^{P(N) \times N}$$

$$\mathbf{Y}.\mathrm{loc} = \mathbf{X}.\mathrm{loc} + 1$$

- **Use** .loc **notation to explicitly retrieve local part of a distributed array**
- **Operation is the same as serial program, but with different data on each processor (recommended approach)**

Parallel Data Maps

Array

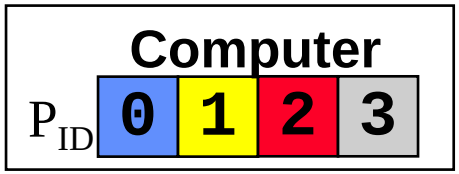


Math

$\mathbb{R}^{P(N)xN}$

$\mathbb{R}^{NxP(N)}$

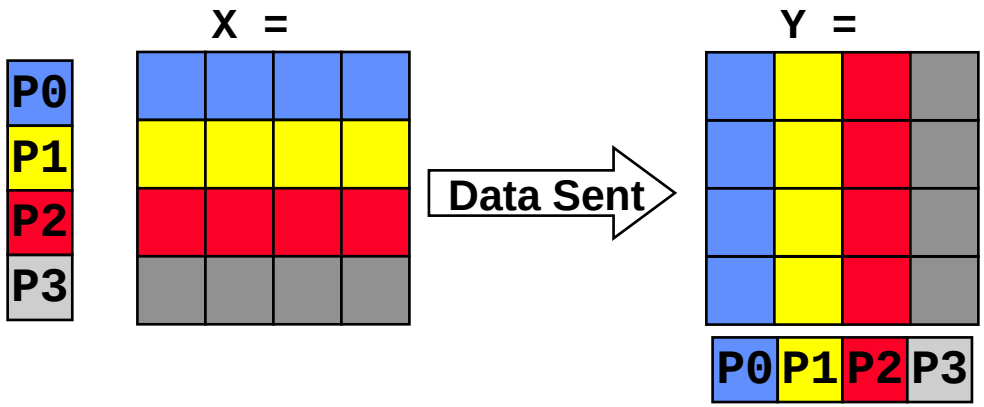
$\mathbb{R}^{P(N)xP(N)}$



- A map is a mapping of array indices to processors
- Can be block, cyclic, block-cyclic, or block w/overlap
- Use P() notation to set which dimension to split among processors

Redistribution of Data

$$\mathbf{X} : \mathbb{R}^{P(N) \times N}$$
$$\mathbf{Y} : \mathbb{R}^{N \times P(N)}$$
$$\mathbf{Y} = \mathbf{X} + 1$$



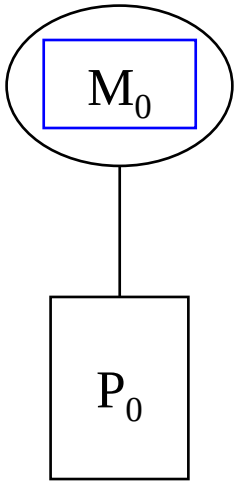
- Different distributed arrays can have different maps
- Assignment between arrays with the "=" operator causes data to be redistributed
- Underlying library determines all the message to send

Outline

- Parallel Design
- Distributed Arrays
- **Kuck Diagrams**
  - ***Serial***
  - ***Parallel***
  - ***Hierarchical***
  - ***Cell***
- Hierarchical Arrays
- Tasks and Conduits
- Summary

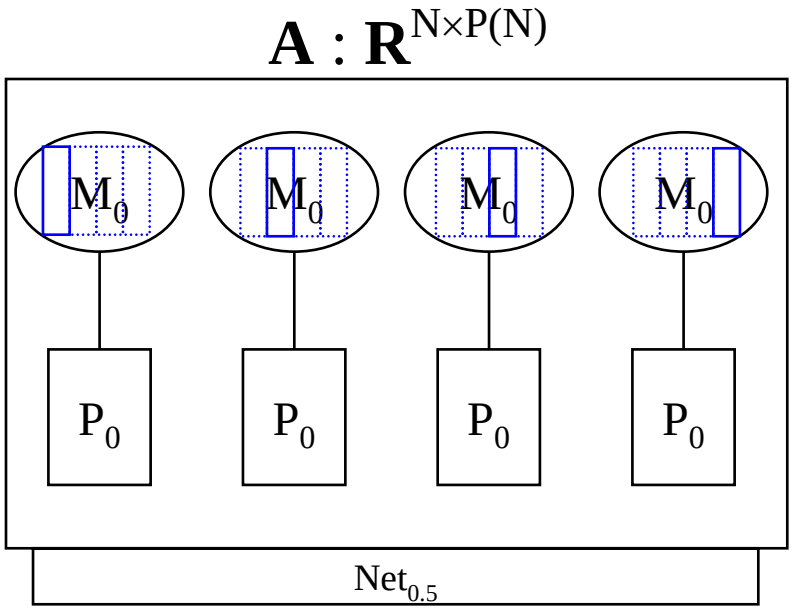
Single Processor Kuck Diagram

$$\mathbf{A} : \mathbf{R}^{N \times N}$$



- Processors denoted by boxes
- Memory denoted by ovals
- Lines connected associated processors and memories
- Subscript denotes level in the memory hierarchy

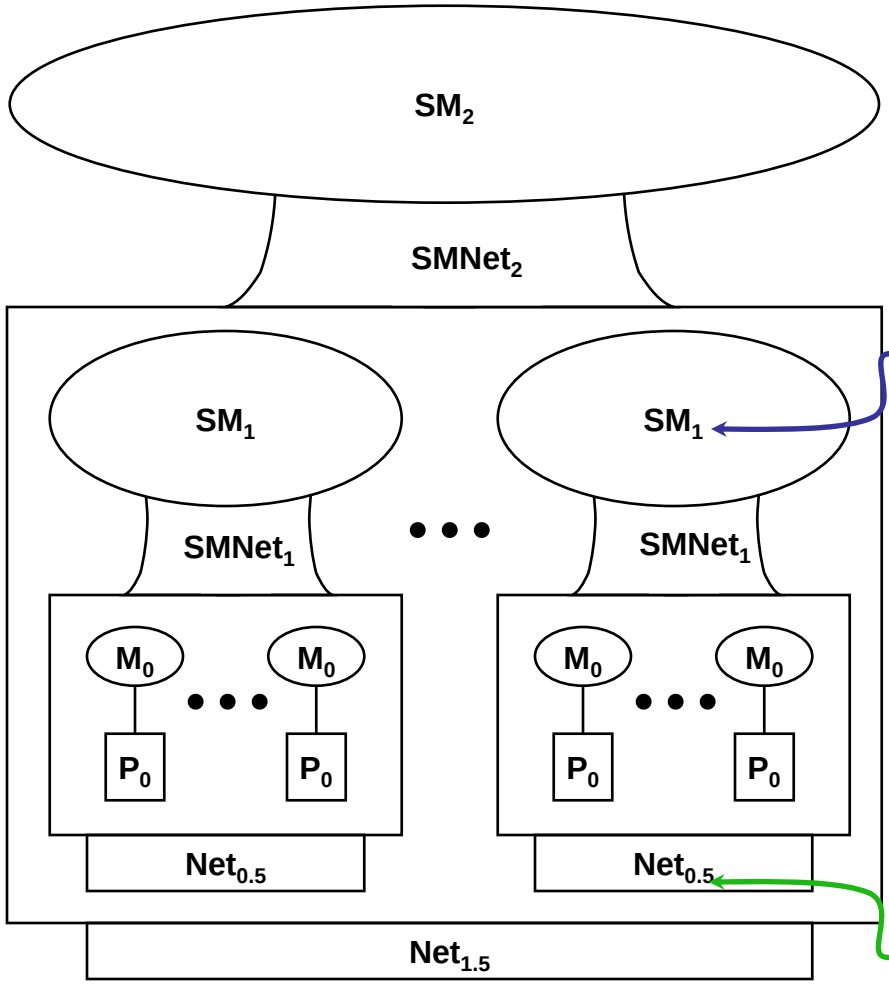
Parallel Kuck Diagram



- **Replicates serial processors**
- Net **denotes network connecting memories at a level in the hierarchy (incremented by 0.5)**
- **Distributed array has a local piece on each memory**

Hierarchical Kuck Diagram

2-LEVEL HIERARCHY



The Kuck notation provides a clear way of describing a hardware architecture along with the memory and communication hierarchy

Subscript indicates hierarchy level

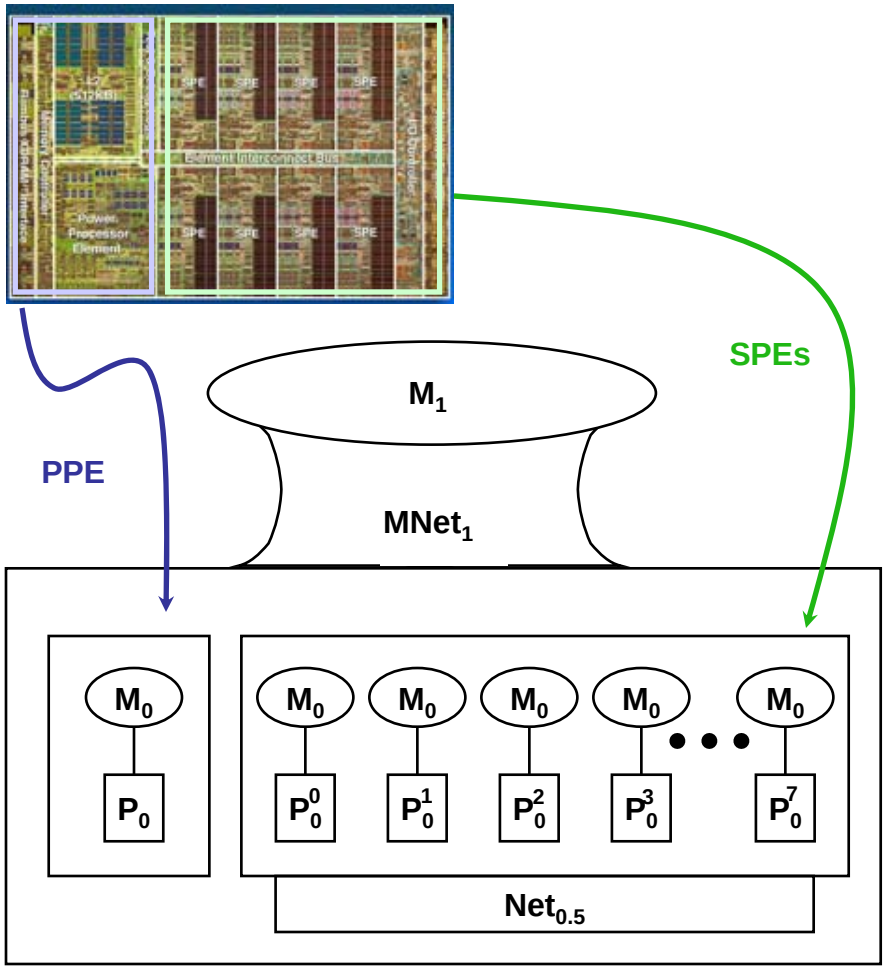
Legend:
- **P - processor**
- **Net - inter-processor network**
- **M - memory**
- **SM - shared memory**
- **SMNet - shared memory network**

x.5 subscript for Net indicates indirect memory access

**High Performance Computing: Challenges for Future Systems*, David Kuck, 1996

Cell Example

Kuck diagram for the Sony/Toshiba/IBM processor



P_{PPE} = PPE speed (GFLOPS)

$M_{0,PPE}$ = size of PPE cache (bytes)

P_{PPE} -$M_{0,PPE}$ =PPE to cache bandwidth (GB/sec)

P_{SPE} = SPE speed (GFLOPS)

$M_{0,SPE}$ = size of SPE local store (bytes)

P_{SPE} -$M_{0,SPE}$ = SPE to LS memory bandwidth (GB/sec)

$Net_{0.5}$ = SPE to SPE bandwidth (matrix encoding topology, GB/sec)

$MNet_1$ = PPE,SPE to main memory bandwidth (GB/sec)

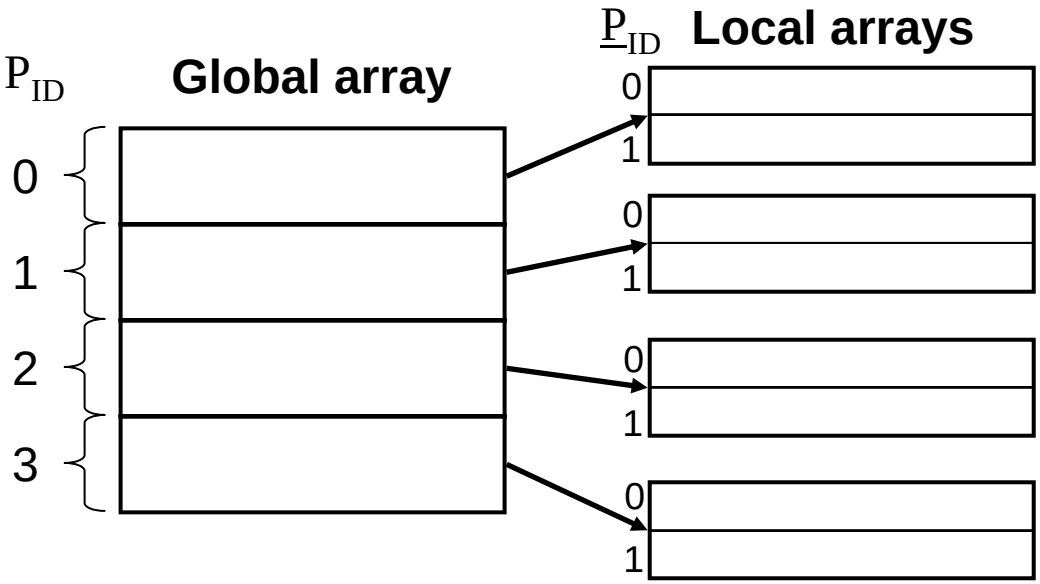
M_1 = size of main memory (bytes)

Outline

- Parallel Design
- Distributed Arrays
- Kuck Diagrams
- **Hierarchical Arrays**
- Tasks and Conduits
- Summary

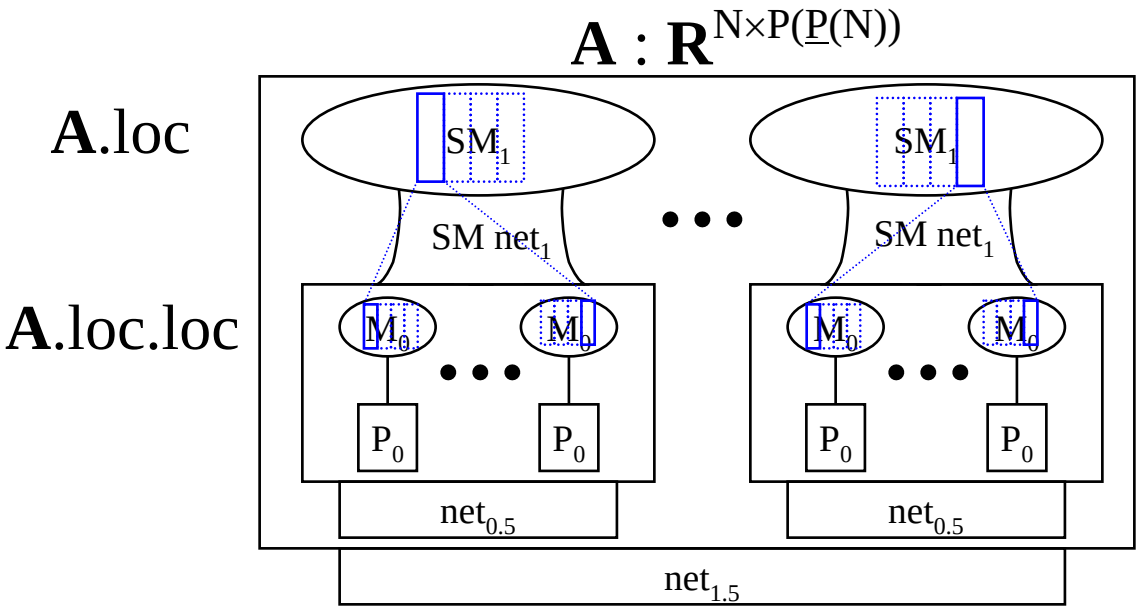
- ***Hierarchical Arrays***
- ***Hierarchical Maps***
- ***Kuck Diagram***
- ***Explicitly Local Program***

Hierarchical Arrays



- Hierarchical arrays allow algorithms to conform to hierarchical multicore processors
- Each processor in P controls another set of processors P
- Array local to P is sub-divided among local P processors

Hierarchical Array and Kuck Diagram



- **Array is allocated across SM_1 of P processors**
- **Within each SM_1 responsibility of processing is divided among local P processors**
- **$\underline{P}$ processors will move their portion to their local M_0**

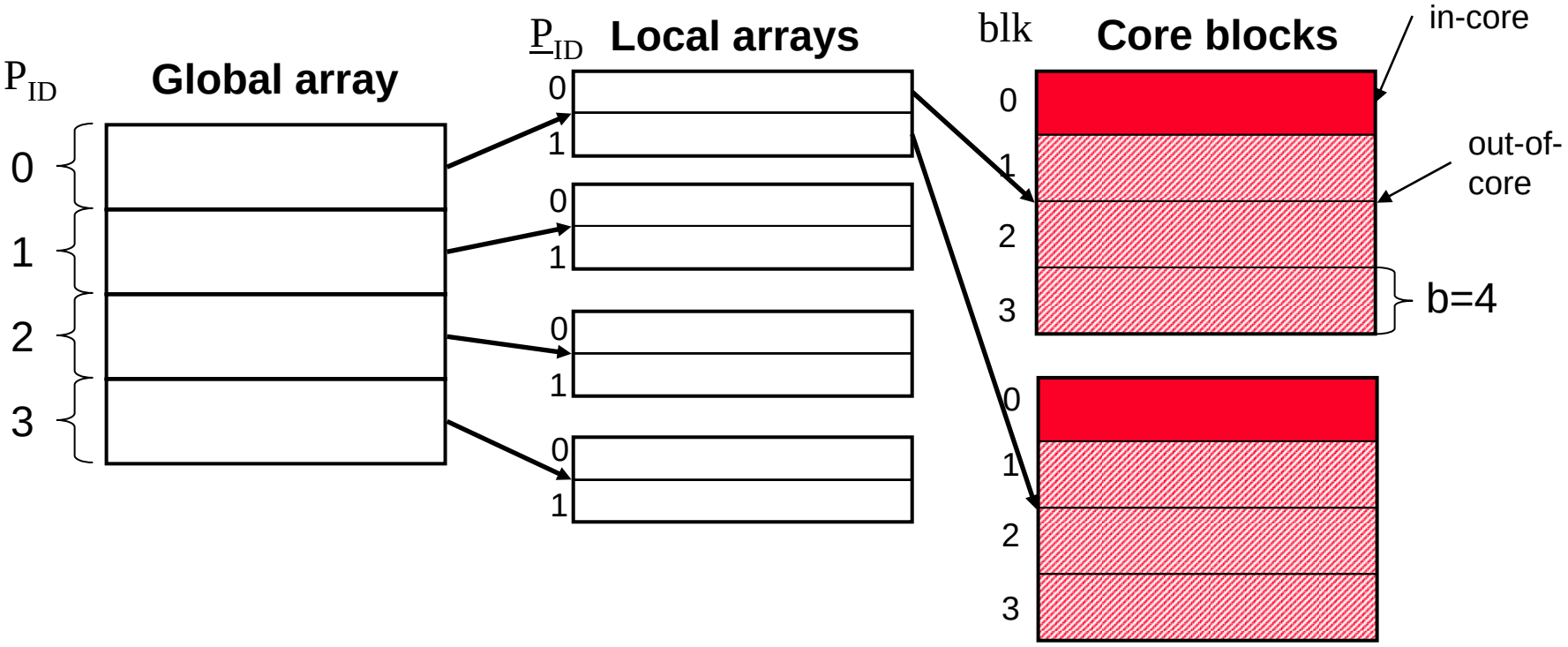
Explicitly Local Hierarchical Program

$$\mathbf{X}, \mathbf{Y} : \mathbb{R}^{P(\underline{P}(N))xN}$$

$$\mathbf{Y}.\mathrm{loc}_{p}.\mathrm{loc}_{\underline{p}} = \mathbf{X}.\mathrm{loc}_{p}.\mathrm{loc}_{\underline{p}} + 1$$

- **Extend** .loc **notation to explicitly retrieve local part of a local distributed array** .loc.loc **(assumes SPMD on** $\underline{P}$**)**
- **Subscript** p **and** $\underline{p}$ **provide explicit access to (implicit otherwise)**

Block Hierarchical Arrays



- **Memory constraints are common at the lowest level of the hierarchy**
- **Blocking at this level allows control of the size of data operated on by each P**

Block Hierarchical Program

$$\mathbf{X},\mathbf{Y} : \mathbb{R}^{P(\underline{P}_{b(4)}(N))xN}$$

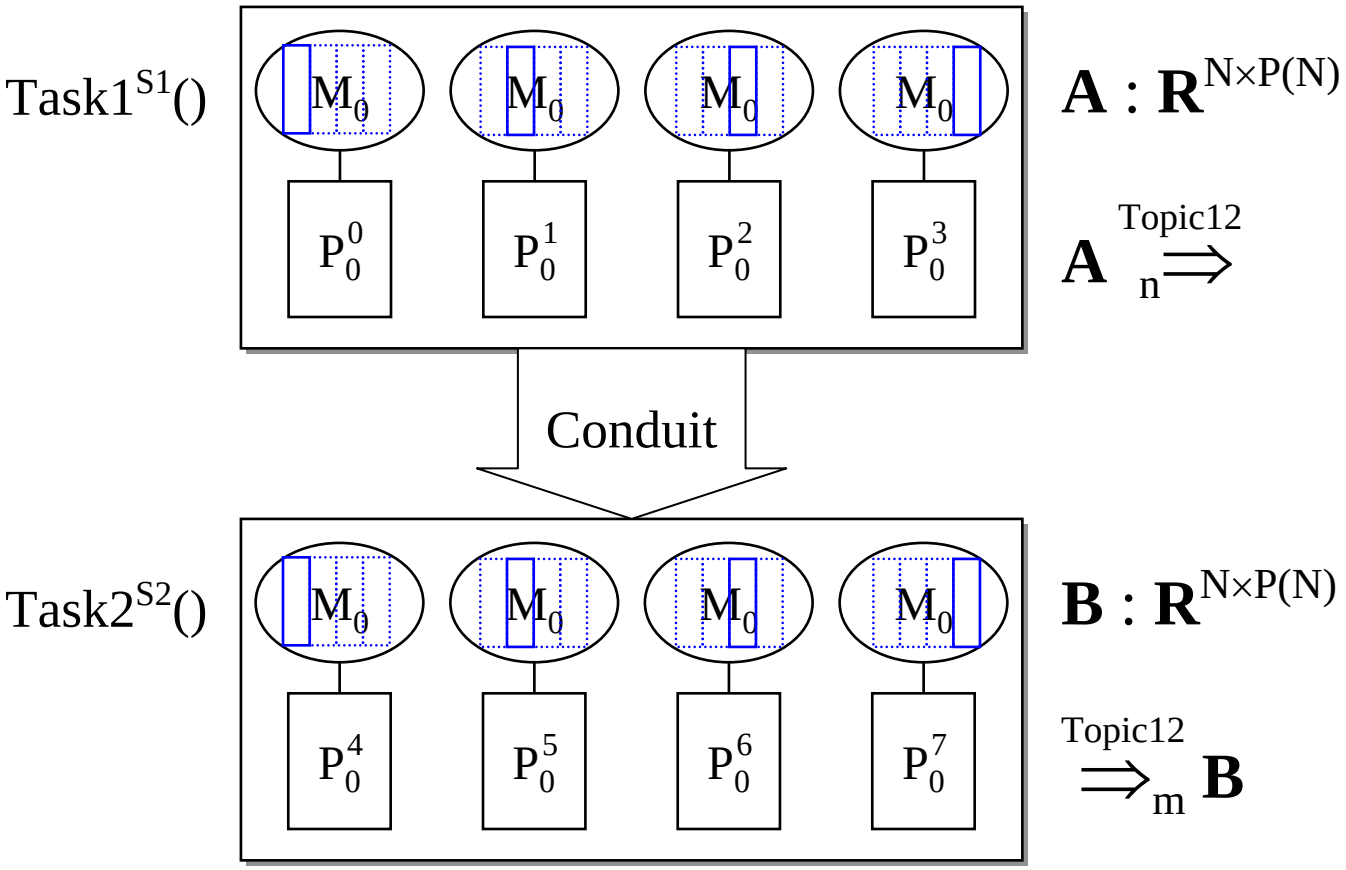
for i=0, $\mathbf{X}$.loc.loc.n_{blk}-1
  $\mathbf{Y}$.loc.loc.blk_i = $\mathbf{X}$.loc.loc.blk_i + 1

- $\underline{P}_{b(4)}$ **indicates each sub-array should be broken up into blocks of size 4.**
- $.n_{blk}$ **provides the number of blocks for looping over each block; allows controlling size of data on lowest level**

Outline

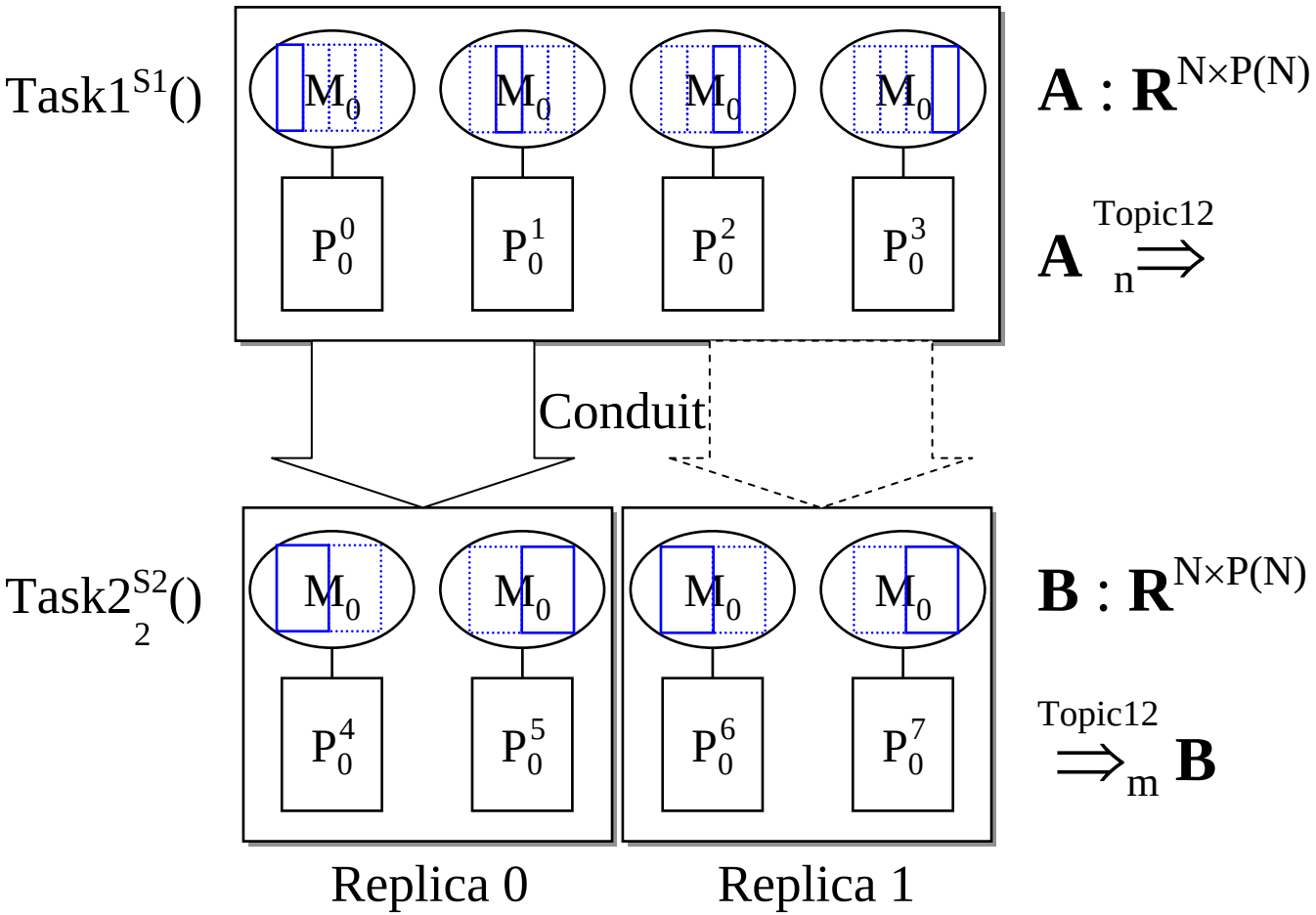
- Parallel Design
- Distributed Arrays
- Kuck Diagrams
- Hierarchical Arrays
- **Tasks and Conduits**
  - ***Basic Pipeline***
  - ***Replicated Tasks***
  - ***Replicated Pipelines***
- Summary

Tasks and Conduits



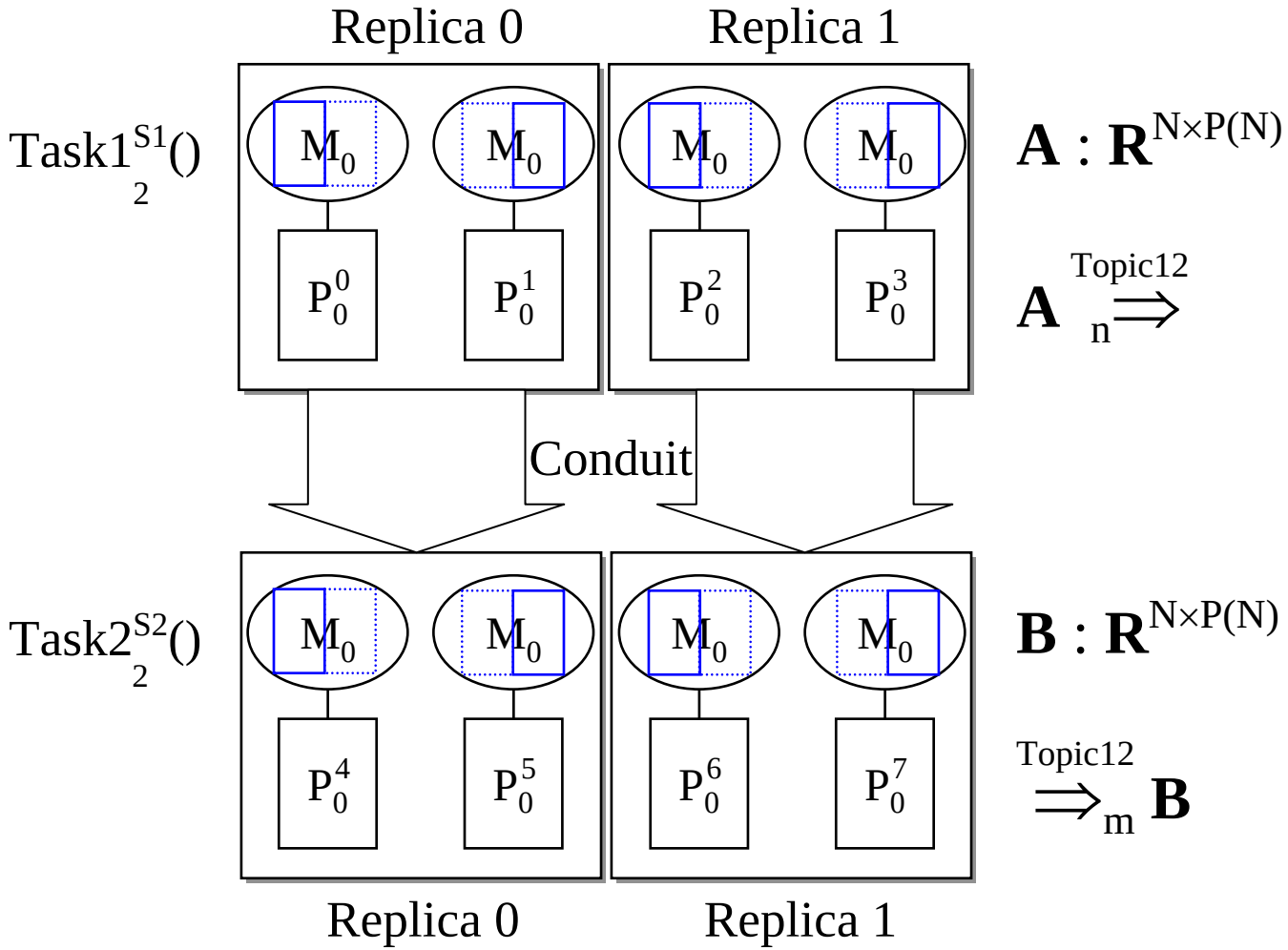
- S1 **superscript runs task on a set of processors; distributed arrays allocated relative to this scope**
- **Pub/sub conduits move data between tasks**

Replicated Tasks



- $_2$ **subscript creates tasks replicas; conduit will round-robin**

Replicated Pipelines



- $_2$ **identical subscript on tasks creates replicated pipeline**

Summary

- **Hierarchical heterogeneous multicore processors are difficult to program**
- **Specifying how an algorithm is *suppose* to behave on such a processor is critical**
- **Proposed notation provides mathematical constructs for concisely describing hierarchical heterogeneous multicore algorithms**