RECIRCULATING, PIPELINED, AND MULTISTAGE SIMD INTERCONNECTION NETWORKS

S. Diane Smith and Howard Jay Siegel
School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

Abstract-- Hardware implementations of the Cube, Shuffle-Exchange, and Plus-Minus-2^i (PM2I) networks are examined. An analysis is made of recirculating (single stage) networks, multistage networks of combinational logic, and pipelined multistage networks. An expanded control structure for recirculating networks is introduced. It allows each processor to select its interconnection function independently of other processors. A multistage Shuffle-No Shuffle-Exchange network is presented. This network allows at each stage a Shuffle or no Shuffle followed by an Exchange. Finally, these networks are examined for their behavior in partitionable SIMD machines.

This research was sponsored by the Air Force Office of Scientific Research, Air Force Systems Command, USAF, under grant number AFOSR-78-3581. The United States Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation hereon.

I. Introduction

This paper examines interconnection networks for SIMD (single instruction stream - multiple data stream) computers. In section III, three types of interconnection functions, the Cube, the Shuffle-Exchange, and the Plus-Minus 2^i, are implemented as recirculating (single stage) networks and as multistage combinational logic networks. Comparisons are made on hardware complexity and delay to transfer data. Section IV considers breaking a data word into S segments before passing the datum through the network. Then the width of the network can be smaller than if the wider data word were passed all at once. S passes can be made through a multistage combination logic network. Alternatively, a pipelined multistage network can be used. The cost and delay to pass S segments are compared for these two cases.

Section V introduces the Shuffle-No Shuffle Exchange network. At each stage of this multistage network, the options are no action, Exchange, Shuffle, or Shuffle-Exchange. This network can perform all the functions of a recirculating Shuffle-Exchange. Yet, if all n stages are needed, it passes the data in the time of a multistage Shuffle-Exchange Network.

Section VI examines the ability of these networks to be partitioned. Various researchers have suggested using multiple control units with an SIMD computer so that the machine may be operated as an MIMD computer composed of smaller SIMD machines ([9],[14],[16]). In such a computer, the interconnection network must provide full capabilities to each SIMD sub-machine.

II. Network Definitions

The model for an SIMD (single instruction stream - multiple data stream) computer used here allows each processor a private memory. This combination is referred to as a processing element or PE. The interconnection network links PEs, and this model is referred to as the PE-to-PE model. Each PE is assigned a unique address from 0 to N-1, where $N = 2^n$. The address in binary of an arbitrary PE P is $p_{n-1}p_{n-2}\cdots p_1p_0$, and $\bar{p}_i$ is the complement of p_i. Each PE has special data transfer registers (DTRs) for passing data to and receiving data from the network. PEs load data into DTRin registers, and the data are moved by the interconnection network to the DTRout registers, from which the PEs may access the data.

An interconnection network may be described as a set of interconnection functions, where each interconnection function is a permutation (bijection) on the set of PE addresses [11]. By applying a sequence of interconnection functions, networks can transfer data among PEs. When interconnection function f is applied, PE(i) passes its data to PE(f(i)) for all i, $0 \leq i < N$, simultaneously. To pass data from one PE to another PE, a programmed sequence of interconnection functions must be executed.

The four networks to be discussed are defined as follows.

The Cube : This network consists of the n functions defined by

$$\text{Cube}_i\ (p_{n-1}\cdots p_1p_0) = p_{n-1}\cdots p_{i+1}\bar{p}_ip_{i-1}\cdots p_0,$$

for $0 \leq i < n$ [11]. The network used in the STARAN is a series of Cube functions. In [1, 2, 10, 12, 13] the usefulness of this type of network is shown.

The Shuffle-Exchange: The Shuffle is defined as [11]

$$\text{Shuffle}(p_{n-1}\cdots p_1p_0) = p_{n-2}\cdots p_1p_0p_{n-1}$$

The Exchange is defined as [17]

$$\text{Exchange}(p_{n-1}p_{n-2}\cdots p_1p_0) = p_{n-1}p_{n-2}\cdots p_1\bar{p}_0.$$

This network has been shown to be useful in [6, 7, 12, 13, 17]. It is the basis of Lawrie's omega network [8]. It is also included in the networks of the RAP [3] and Omen [5] systems.

The Plus-Minus 2^i (PM2I): This network consists of the 2n functions defined by

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

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

for $0 \leq j < N$, $0 \leq i < n$ [11]. Feng's data manipulator [4] is a series of PM2I functions. The various data manipulating functions that the PM2I network can perform are discussed in [4, 12, 13, 15].

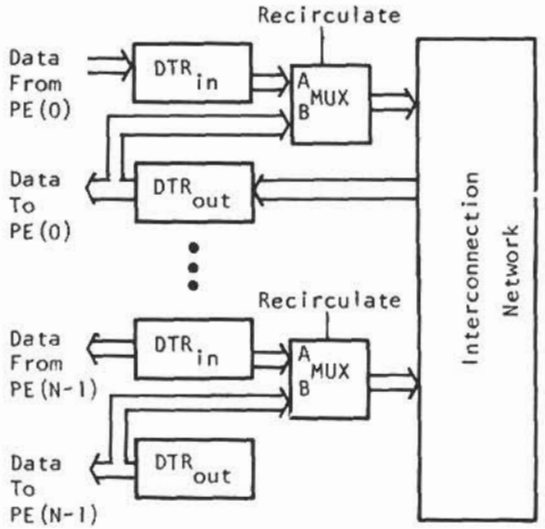

Figure 1: Model for recirculating network.

Two structures for an interconnection network will be considered. A recirculating network is an interconnection network with a single stage of switches. The stage is reused until data reach their final destinations. To facilitate this, the model will allow the network to receive data either from DTRin or DTRout, as shown in figure 1.

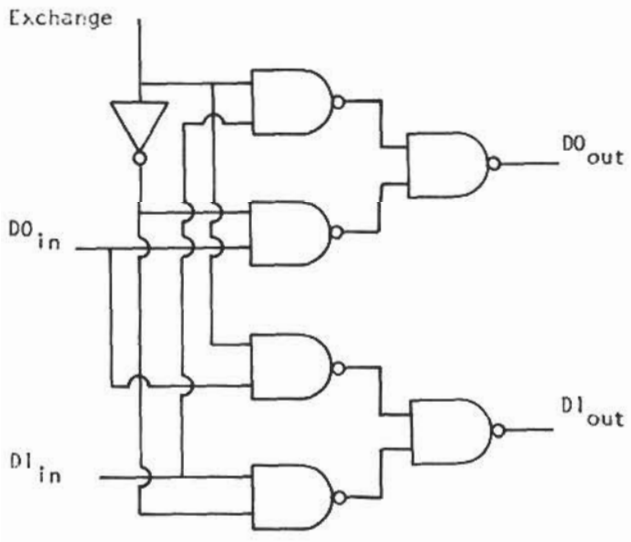

Figure 2: Interchange box.

A complete data transfer may take M passes through the network. A multistage network is an interconnection network composed of several, usually n, stages of combinational logic switches. Multistage networks such as the Cube and Shuffle-Exchange may be implemented using interchange boxes, shown in figure 2, as building blocks for each stage. An example of such a multistage network is shown in figure 3. In general, a single pass through a multistage network is sufficient to route data to their destinations. However, when a single pass is insufficient, multiple passes may be used.

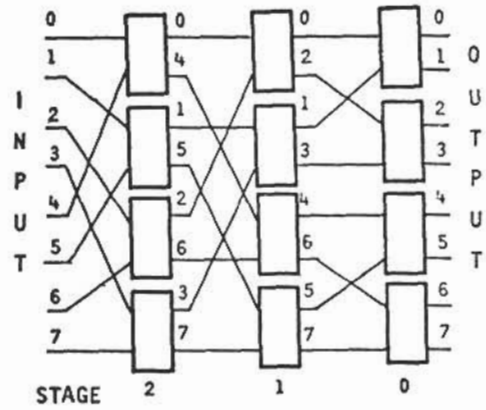

Figure 3: An 8-item Generalized Cube.

III. Hardware Implementations

In order to compare these networks, typical circuits for each are presented. Comparisons of gate count and circuit delay are made for multistage and recirculating Shuffle-Exchange, Cube, and PM2I networks. For simplicity, the following analysis is made on networks that are one bit wide. The costs of DTRin, DTRout, and any hardware needed to interface with the network are not included in the network cost estimates. The delay times presented are intended as ballpark figures and are useful for comparisons. In practice, the speed of the network will also depend on the speed at which control signals can be generated and the technology of the circuit design.

Table 1 summarizes the notation that will be used in the discussion.

Three multistage Cube networks have been presented in the literature: the STARAN flip network [2], the indirect binary n-cube [10], and the Generalized Cube [15]. In [15], it was shown that these three networks and an n-stage Shuffle-Exchange network are all topologically equivalent.

NOTATION	MEANING
dr	delay of a register
cr	cost of a register
dm	delay of a multiplexer
cm	cost of a multiplexer
dms	delay of the logic for one stage of a multistage network
cms	cost of the logic for one stage of a multistage network
drn	delay of the logic for a recirculating network
drs	delay of the logic for a recirculating Shuffle-Exchange network
cbr	cost of the logic (buffers) for a recirculating network
Cr	cost of a recirculating network
Cp	cost of a pipelined multistage network
Cm	cost of a combinational logic multistage network
Tr	time delay of a recirculating network
Tm	time delay of a combinational logic multistage network
Tp	time delay of a pipelined multistage network
W	width of a data word to be transmitted through the network
S	number of segments into which a data word is divided
M	the number of passes made through a recirculating network to complete a desired transfer
N	the number of PEs in the system
n	$\log_2 N$

Table 1: Definitions and abbreviations

With this in mind, a circuit for an 8-item Generalized Cube network is presented in figure 3. The interchange box (figure 2), on which this Cube network is based, conditionally interchanges the data at its inputs, thus performing a conditional exchange. So, stage i of the network forms $Cube_i$, $0 \leq i < n$. The circuit uses n*N/2 interchange boxes, or 7n*N/2 gates, and has a delay of dms*n + 2*dr, where dms is the delay through the interchange box, and 2*dr represents the delay through DTRin and DTRout.

As a design example, suppose it is desired to build a multistage Generalized Cube network for N = 1024. This 10 stage network could be designed using off-the-shelf components. An inverting interchange box made from AND-OR-INVERT gates (SN7451) can be used in place of the NAND gates in figure 2. This circuit performs the same function as the interchange box, but the outputs are complemented. Since interchange boxes are cascaded to form a multistage network, if n is even, these inverting outputs cancel. For example, after stage n-1, the data are complemented, but after stage n-2, the data are true. A design for a 10 stage 1 bit wide Generalized Cube network would require 512*10 = 5120 dual 2-wide 2-input AND-OR-INVERT chips (SN7451) and 5120/6 = 854 hex inverter chips (SN7404). This is a total of 5974 integrated circuit packages, or less that 6 integrated circuit packages bit per PE.

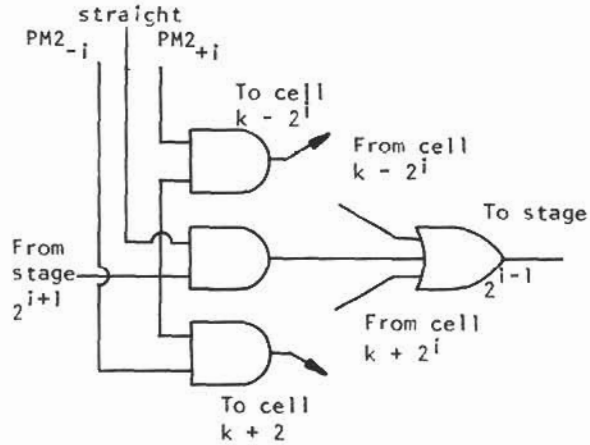

Figure 4. A PM2I module for row k.

A multistage PM2I network can be built from the modules of figure 4 [4]. Referring to the multistage network for N=8 in figure 5, a logic module of stage 2 has the construction of the left-hand side of the module of figure 4, while the receiving right-hand side of the module lies in stage 1 of figure 5. The number of gates needed for an n stage PM2I network is 4*N*n, and the delay is dms*n+2dr.

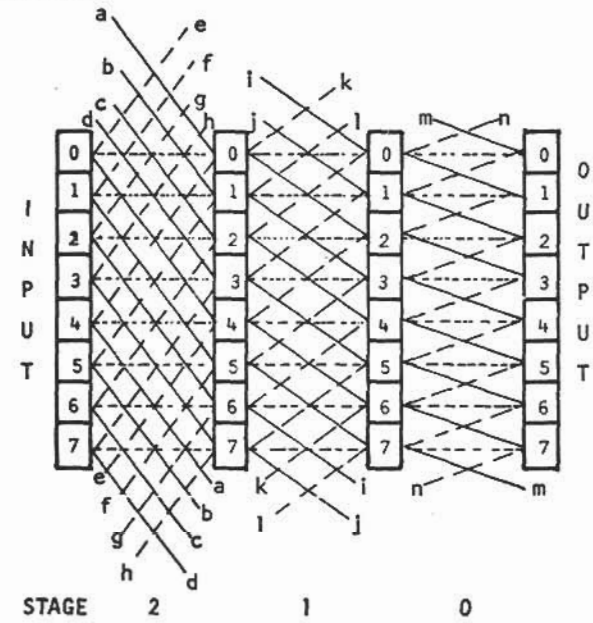

Figure 5: 8-item data manipulator network [4].

A 10 stage 1 bit wide PM2I network could be designed from discrete components using 3*1024*10/4 = 7680 2-input NAND packages (SN7400) and |1024*10/3| = 3414 3-input NAND packages (SN7410). This is 11,094 integrated circuit packages, or less than 11 chips per bit per PE.

Figure 6 shows a recirculating Shuffle-Exchange network. At any pass through the network, either a Shuffle or an Exchange may take place. M passes through the network may be required to complete a desired transfer of data between PEs. This circuit uses 3N gates and has a delay of dr + M(dr + dm + drs).

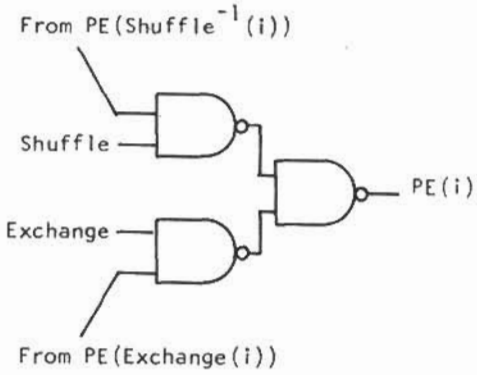

Figure 6: Circuit for a recirculating Shuffle-Exchange network for PE(i), $0 \leq i < N$.

The recirculating Cube network can be built using tri-state buffers with outputs that can be OR-TIED together, as shown in figure 7. A recirculating PM2I could be constructed similarly.

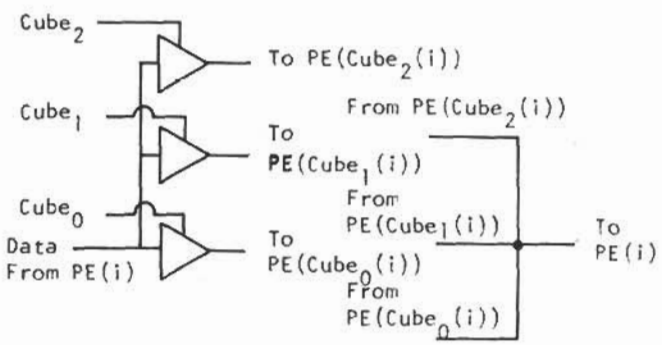

Figure 7: Circuit for a recirculating Cube network for PE(i), $0 \leq i < 8$.

The Cube network uses N*n buffers while the PM2I network uses $2*N*(n-1)$ buffers, since $PM2_{+(n-1)} = PM2_{-(n-1)}$. Both have delay dr + M (dr + dm + drn). For small n, the Cube network could be built using 2 levels of NAND gates, where the input to the network is composed of 2-input NANDs and the receiving side is composed of n-input NANDS. For large n, due to fan-in and integrated circuit package count limitations, the tri-state buffer design is preferable.

When selecting an interconnection network, the time to complete a data transfer is an important consideration. For a recirculating network, this time is proportional to the number of passes made through the network, M , $0 \leq M$. In the case of the multistage network, for any data transfer, all n stages must be traversed, so as the number of PEs grows, the time to pass data increases.

For some value of M, a multistage network and a recirculating network will have the same delay time. If dr = 6 ns, dm = 5 ns, drn = 4.5 ns, and dms = 6 ns, then the two networks will require the same delay time if M = .39 (1 + n). This indicates that the choice between a recirculating network and a multistage network should depend on the number of interconnection functions used on the average to complete a data transfer, and thus depend on the types of problems that a system is designed to perform. For example, if skewed storage [18] is used, there will be uniform shifts which may require all n Cube functions of a multistage network. However, sorting using the Cube will need only one interconnection function at a time [13].

The control structure of a network is an important consideration. For multistage networks, three types of controls are discussed [15]. Individual stage control allows one control signal for each stage of the network. Individual box control uses a separate control signal for each interchange box in the network, using hardware [10] or software (destination tags) [8]. Partial stage control uses more than one but less than N control signals at any stage of the network. The implementations discussed above can be used with any of these control schemes.

The typical control mechanism for a recirculating network assumes that a PE is active or not active. In the model used here, an active PE can send and receive data; an inactive PE can only receive data. For recirculating networks, a new control is defined that differs from the usual SIMD control. Usually, since there is a single instruction stream, all PEs must execute the same interconnection function. For example, if one PE executes $Cube_0$, all active PEs must execute $Cube_0$. By providing each PE with its own control register, this restriction is removed. Independent function control allows each PE to execute any set of interconnection functions. For example, PE P using a Cube network might send data to all PEs whose addresses differ in one bit from P's address by simultaneously executing $Cube_0$, $Cube_1$, and so forth. Also, PE 0 may use $Cube_0$ while PE 1 uses $Cube_1$. The implementations discussed here can be used for conventional or independent function control.

IV. Combinational logic multistage networks vs Pipelined multistage networks

In the previous section, networks were analyzed as if they were one bit wide. Data words may be sent through the network bit serially, but other methods may be more efficient.

Let the width of a data word be W bits. A network may be designed as W planes, where each plane is a one bit wide network [8]. As an example, consider the 10-stage Generalized Cube network for N = 1024 that was described in section III. The number of packages required for W = 32 is 5974 * 32 = 191,168 integrated circuit packages, or about 187 per PE.

The amount of hardware could be reduced by a compromise. The data word could be divided into $W/B = S$ segments of data, the network constructed as a B bit wide network, and then the data word passed in S uses of the network. In this manner, if the delay of the network is D, then the delay to pass the entire data word through the network is $S*D$. If the number of gates in a W wide network is G, the number of gates in the reduced network is $G*B/W = G/S$.

In order to show how this division of the network might be done, three sample hardware designs are presented. Design 1 moves data into and retrieves data from the network B bits at a time under software control. This method is slow, but requires no extra hardware for control. Design 2 multiplexes the S segments into and out of the network. The W bit data word is loaded into DTRin. An S-to-1 multiplexer supplies the network with each segment of data at the proper time. An S-to-1 demultiplexer retrieves the segments from the network and arranges them in DTRout. Since the maximum delay of the network is much less than one instruction cycle, this design is faster than design 1. But, it requires more hardware for the W bit wide DTRin and DTRout registers and the B bit wide multiplexer and demultiplexer. Design 3 constructs DTRin and DTRout from B S-bit shift registers. DTRin is made from parallel-in-serial-out registers, and DTRout is made from serial-in-parallel-out registers. Let $d_{W-1}\cdots d_1 d_0$ be a data word. The first register of DTRin stores bits $d_{S-1}\cdots d_1 d_0$. The second register stores bits $d_{2S-1}\cdots d_{S+1} d_S$, and the last register stores bits $d_{W-1}\cdots d_{W-S+1} d_{W-S}$. Each clock period, the least significant bit of each of the B shift registers is presented to the network, the registers are shifted, and the next B bits are ready to be presented. At the output, each of the S shift registers of DTRout receive one bit from the network each clock period. After S clock periods, the S bits in each of the B shift registers are presented as a W bit word to the PE. DTRin and DTRout will be treated as W-bit registers by the balance of the system. This design is faster than design 1 and requires less hardware than design 2.

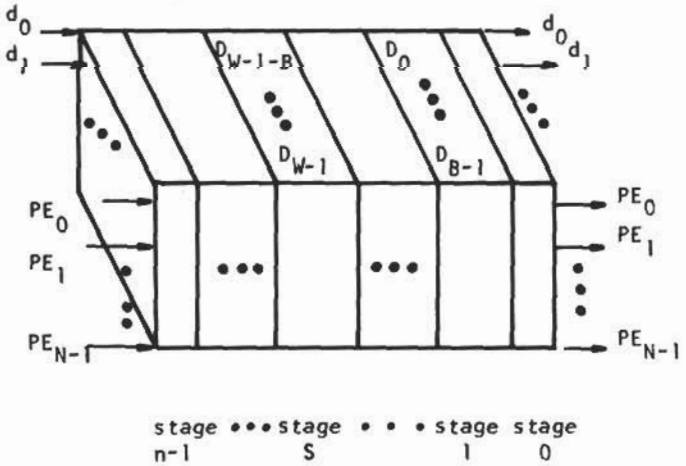

Figure 8: Model for pipelined network of width B for N PEs and data word $D_{W-1}\cdots D_1 D_0$.

For an extra cost, overlap parallelism may be added to a multistage network to reduce the total time to move S segments of data. Assume that the network is B bits wide, that DTRin and DTRout are W bits wide, and that costs for interfacing between the DTRs and the network are small compared to the cost of the network. Let the network be an n stage pipeline with registers of delay dr between each stage, each stage having delay dms. Figure 8 illustrates this arrangement.

The delay for the combinational logic multistage network is the time to load DTRin, plus the time to pass through the network, plus the time to load DTRout. The cost, n*cms*B, considers only the network and not DTRin and DTRout. The delay of the pipelined network is n * (dms + dr) to get the first segment from the network, and the remaining segments arrive at DTRout in the next S-1 time delays. The cost of the pipelined network is that of the multistage network plus cr*(n-1)*B for the N-bit registers that are placed between each of the n stages.

The S segments of data may be transferred using a pipelined network in time

$$Tp = (dr+dms)(n + S - 1).$$

The unpipelined network transfers the same data in time

$$Tm = S(dms * n + 2 * dr).$$

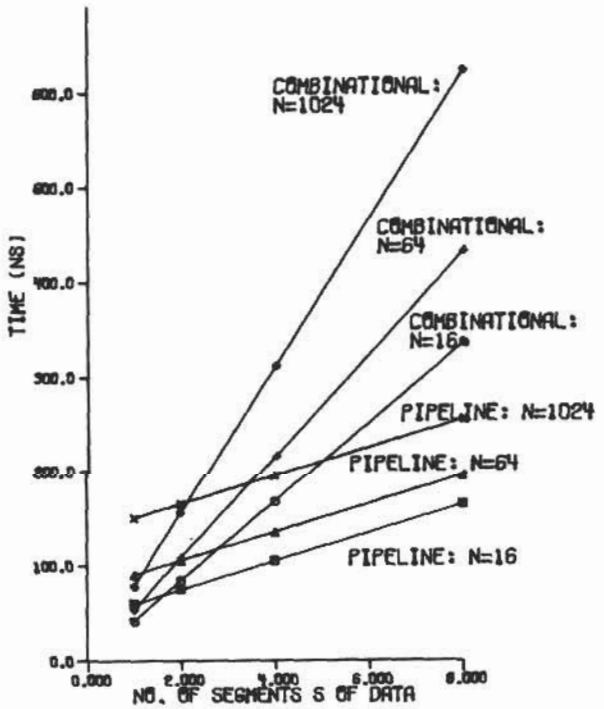

Figure 9: Tp vs. Tm.

Figure 9 plots Tp vs Tm for various values of N and S, for dr = 9 ns and dms = 10 ns. These time approximations are based on schottky logic according to [19].

Suppose that a combinational logic multistage network and a pipelined multistage network have equal cost. Since the pipelined network uses more hardware than the combinational logic multistage network, the width of the pipelined network is less than the width of the combinational logic multistage network. The relationship between S_m, the number of data segments for the multistage network, and S_p, the number of data segments for the pipelined network, is

$$(n * cms + (n-1) * cr)\ W/S_p = n * cms * W/S_m$$

If cms = cr, that is, the cost of the logic of a stage is about that of the register at the output of the stage, then

$$S_p = (2*n - 1)\ S_m / n$$

Table 2 lists S_p and S_m for various values of N for combinational and pipelined networks of equal cost.

N	S_m	S_p
128	1	1.86
128	4	7.43
128	8	14.86
128	16	29.71
1024	1	1.90
1024	4	7.60
1024	8	15.20
1024	16	30.40

Table 2: N, S_p, and S_m for equal cost networks.

Consider the average time to pass one segment of data. The combinational logic multistage network passes, on the average, one segment of data in (dms*n+2dr) time units. The pipelined network uses an average of

$$\frac{(n+S-1)(dms+dr)}{S} \quad \text{time units/segment.}$$

The larger S is, the smaller this average time will be. These two average times are equal for

$$S = \frac{(n-1)(dms+dr)}{(dms(n-1)+dr)}$$

For example, if dms=dr and n=10, then

$$S = \frac{(10-1)(2dms)}{dms(10-1+1)} = \frac{18}{10}$$

So, for $S \geq 2$, the average delay per segment is less for the pipelined network than for the combinational logic multistage network, although the total time to pass one data item may be greater, due to the time to fill and empty the pipe. This suggests that a pipelined multistage network is most applicable where many segments of data are passed, such as passing blocks of data at once rather than one data word.

V. The Shuffle-No Shuffle-Exchange Network

For the Cube and PM2I networks, any data transfer that can be accomplished by passes through a multistage network can also be accomplished by a recirculating network, and vice versa. This is not true for the multistage Shuffle-Exchange network. From theorem 2 of [8], it can be shown that the multistage Shuffle-Exchange network cannot perform the Shuffle permutation. To try to rectify this, a Shuffle-No Shuffle-Exchange (SNSE) network is introduced here. At each stage of this multistage network (figure 10), the options are do

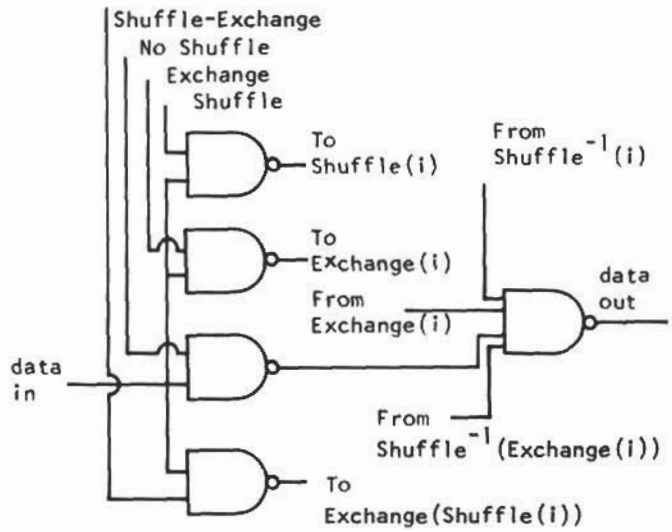

Figure 10: Shuffle-No Shuffle-Exchange circuit for row i, $0 \leq i < N$.

nothing, Shuffle, Exchange, or Shuffle-Exchange Assume that at any stage, a Shuffle affects all PEs, and that any Exchange signal affects both PE(i) and PE(Exchange(i)). Unlike Lawrie's omega network, no broadcast functions will be allowed. This network can produce all the permutations of the recirculating Shuffle-Exchange network. Thus, it has the advantage over the multistage Shuffle-Exchange network of being able to perform one to n Shuffles in one pass through the network.

Let $P = \{ (Si, Di) \mid 0 \leq i < N \}$ be a permutation mapping of source PE address, Si, to destination PE address, Di, where the binary representation of Si is $p_{n-1} \cdots p_1 p_0$ and that of Di is $d_{n-1} \cdots d_1 d_0$. A network passes a permutation P if and only if P conforms to the acceptable form for that network, and no conflicts result in the passage through the network [8]. W ↑ P means the network, W, passes a permutatioh P.

Theorem 1: An n-stage SNSE network can form the following permutations.

For all t, $n > t \geq 0$,

$$\left\lfloor \frac{Si}{2} \right\rfloor \bmod 2^{n-(t+1)} = \left\lfloor \frac{Di}{2^{t+1}} \right\rfloor \bmod 2^{n-(t+1)}$$

$$\text{AND } (Si \neq Sj \rightarrow \text{NOT } (\bigvee_{K=0}^{t} (\lfloor \frac{Si}{2} \rfloor \bmod 2^{n-K-1} = \lfloor \frac{Sj}{2} \rfloor \bmod 2^{n-K-1} \text{ AND } \lfloor \frac{Di}{2^{t-K}} \rfloor \bmod 2^{K+1} = \lfloor \frac{Dj}{2^{t-K}} \rfloor \bmod 2^{K+1}))),$$

and $Si \neq Sj \rightarrow \text{NOT } (Si \bmod 2^K = Sj \bmod 2^K$ AND $\lfloor \frac{Di}{2^K} \rfloor \bmod 2^{n-K} = \lfloor \frac{Dj}{2^K} \rfloor \bmod 2^{n-K})$ for $1 \leq K \leq n$.

Proof: An n-stage SNSE network, designated W, will be analyzed by parts. W0 will refer to a network constructed with only an Exchange function, i.e., W0 = E. W1 prefixes an Exchange-Shuffle stage to W0, i.e., W1 = (ES)W0 = ESE. W2 prefixes an Exchange-Shuffle to W1, i.e., W2 = (ES)W1 = (ES)(ES)W0 = (ES)(ES)E. For $0 < t < n$, Wt = (ES)W(t-1) = $(ES)^t E$. The last prefix creates Wn, an n stage Shuffle-Exchange network, by prefixing a Shuffle stage to Wn-1, i.e., Wn = $S(ES)^{n-1}E$ = $(SE)^n$. The permutations that W can pass are the union of the permutations that W0, W1,... , and Wn can pass.

Note that certain cases are not explicitly considered here. For example, a Shuffle followed by k No Shuffles is equivalent to k No Shuffles followed by a Shuffle. Also, a Shuffle followed by two Exchanges has the same effect as just a Shuffle. This first cases are not considered in this argument; the second equivalent ones are.

W0 accepts the permutations

$$p_{n-1}\cdots p_1 p_0 \rightarrow p_{n-1}\cdots p_1 \bar{p}_0.$$

A conflict, C0, between two different sources Si and Sj results iff Di = Dj, that is
C0 = (|Si/2| = |Sj/2|) AND (Di mod 2 = Dj mod 2). So,

$$W0 \text{ ↑ } P \Leftrightarrow (|Si/2| = |Di/2|) \text{ AND } (Si \neq Sj \Rightarrow \text{NOT}(C0)),\ 0 \leq i,j < N.$$

Recall that Wt = $(ES)^t E$. For $0 < t < n$, the transition of data is

$$p_{n-1}\cdots p_0 \rightarrow p_{n-1}\cdots p_1 d_t \rightarrow p_{n-2}\cdots d_t d_{t-1} \rightarrow p_{n-3}\cdots d_t d_{t-1} d_{t-2} \rightarrow \cdots \rightarrow p_{n-1-t}\cdots p_1 d_t d_{t-1}\cdots d_0$$

The acceptable permutations are, thus,

$$\lfloor Si/2 \rfloor \bmod 2^{n-(t+1)} = \lfloor Di/2^{t+1} \rfloor \bmod 2^{n-(t+1)}.$$

For $Si \neq Sj$, a conflict results after k Shuffles and k+1 possible Exchanges, $0 \leq k \leq t$, iff

$$Ct(k) = (\lfloor Si/2 \rfloor \bmod 2^{n-k-1} = \lfloor Sj/2 \rfloor \bmod 2^{n-k-1}) \text{ AND } (\lfloor Di/2^{t-k} \rfloor \bmod 2^{k+1} = \lfloor Dj/2^{t-k} \rfloor \bmod 2^{k+1}).$$

Thus, the expression for a conflict, Ct, is

$$Ct = \bigvee_{k=0}^{t} Ct(k).$$

Therefore,

$$Wt \text{ ↑ } P \Leftrightarrow (\lfloor Si/2 \rfloor \bmod 2^{n-(t+1)} = \lfloor Di/2^{t+1} \rfloor \bmod 2^{n-(t+1)})$$

AND ($Si \neq Sj \Rightarrow$ NOT(Ct)), $0 \leq i,j < N$.

Wn is a Shuffle stage concatenated with Wn-1 and is an n-stage Shuffle-Exchange network. From [8], it is seen that Wn accepts a permutation P iff

$$Si \neq Sj \Rightarrow \text{NOT}(Si \bmod 2^k = Sj \bmod 2^k \text{ AND } \lfloor Di/2^k \rfloor \bmod 2^{n-k} = \lfloor Dj/2^k \rfloor \bmod 2^{n-k}),\ 1 \leq k \leq n.$$

The n-stage network W passes all permutations that are passed by W0, W1,..., Wn. □

The SNSE network can form all the permutations of the recirculating Shuffle-Exchange. If x passes through the recirculating network are needed, then $\lceil x/n \rceil$ passes through the SNSE network accomplish the same data transfer. This flexibility has been accomplished for the cost of a few extra gates per PE, 5*N*n for the SNSE network versus 7/2 N*n for the multistage Shuffle-Exchange, and for more control signals, n*N/2 for the multistage Shuffle-Exchange and (1(Shuffle) + N/2(Exchange) + N/2(Exchange Shuffle)) * n signals for the SNSE network.

VI. Partitioning

In [15], two types of partitioning for SIMD machines were introduced. Single control partitioning uses the computer system as one to 2^{n-r} SIMD machines, each having 2^r PEs, each performing the same algorithm (using the same instruction stream), but each on a different data set. Multiple control partitioning uses the system as one to 2^{n-r} SIMD machines, each of size 2^r, where each may be performing a different algorithm on a different data set. Both types of partitioning require that each partition have available a complete interconnection network for 2^r PEs. Multiple control partitioning assumes multiple control units ([9], [14], [16]). This means that the PEs in a group can choose interconnection functions independently of other groups. In [15], the capabilities of combinational logic multistage networks to perform in partitioned environments were discussed. The analysis is extended here for pipelined networks and recirculating networks.

Theorem 2: A pipelined multistage network can be partitioned into 2^{n-r} groups of 2^r PEs in the same manner as the combinational logic multistage network upon which it is based.

Proof: The transition from a combinational logic multistage network to a pipelined multistage network does not change the overall interconnection or control structure of the network. Therefore, the pipelined network must partition in the same manner as the combinational logic multistage network upon which it is based. The analysis for combinational logic multistage Cube, Shuffle-Exchange, and data manipulator (PM2I) networks are in [15]. □

Theorem 3: A recirculating Cube network can be partitioned into 2^{n-r} groups of 2^r PEs under single or multiple control partitioning.

Proof: A recirculating Cube may be partitioned in

a manner analogous to that of a multistage Cube [15]. Let the addresses of all PEs in each partition have the same n-r most significant bits. In order to partition the recirculating Cube network, the n-r interconnection functions $Cube_r$, $Cube_{r+1}$,..., and $Cube_{n-1}$ are not utilized for any data transfer. So, each group of 2^r PEs has available to it $Cube_{r-1}$,...,$Cube_1$, and $Cube_0$. Thus, single and multiple control partitioning are possible.

This argument can be extended to group PEs that have any set of n-r bits in common. □

Theorem 4: A recirculating PM2I network may be partitioned, under single or multiple control partitioning, into 2^{n-r} groups of 2^r PEs so that all PEs in a partition have the same n-r least significant bits.

Proof: In [15], it was shown that a multistage PM2I network can be partitioned into groups that have the n-r least significant bits in common by not utilizing the last n-r stages of the network, that is, the stages for $PM2_{\pm i}$, $0 \leq i < n-r$. In this manner, each group has available to it 2*r interconnection functions, where $PM2_{\pm(n-r+i)}$ now behaves as $PM2_{\pm i}$ for a group, $0 \leq i < r$. No group can send data outside of its group, since the n-r least significant bits never change for these 2*r functions. So, each group may use the network independently of any other group. Thus, single and multiple control partitioning are possible. □

In [15], the machine was partitioned into 2^{n-r} blocks of 2^r PEs. Here, a more general control structure for partitioning will be used, where the machine may be partitioned into blocks of varying sizes with the restriction that the sizes are powers of two. The following merging theorem shows a translation from the restricted partitioning to the more general control structure.

Theorem 5: Let a Generalized Cube network be partitioned into 2^{n-r} groups of 2^r such that all PEs in a partition have the same n-r most significant bits. Let G and H be partitions in the machine. The addresses of the PEs in G are

$$g_{n-1}\cdots g_r * 2^r + P,\ 0 \leq P < 2^r,$$

and the addresses of the PEs in H are

$$h_{n-1}\cdots h_r * 2^r + Q,\ 0 \leq Q < 2^r.$$

If for all i, $n > i > r$, $h_i = g_i$ and $h_r \neq g_r$, then G and H may be merged into one partition of size 2^{r+1} with no effect on the remainder of the network.

Proof: In [15], it was shown that a Generalized Cube network (with independent box control) can be partitioned into 2^{n-r} groups of 2^r PEs under multiple control partitioning. So, G and H can use the network without interfering with each other. At stage r, $Cube_r$ moves data between PEs in G and PEs in H, since $h_r \neq g_r$. No other PEs can be affected, since if the groups are of size 2^r and for $n > i > r$, $h_i = g_i$, when a PE in G uses $Cube_r$ it must connect to a PE in H and vise versa. Therefore, the use of $Cube_r$ by G and H does not affect the remainder of the network. Therefore, G and H can be merged into a partition of size 2^{r+1} and remain independent of the rest of the network. Furthermore, the new group of 2^{r+1} has a complete Cube network of r+1 functions.

Analogous arguments can be made to show that the theorem is true for systems that are partitioned based on any set of n-r bits of the PE addresses. □

In [15], it was shown that the multistage Shuffle-Exchange network and the Generalized Cube network are topologically equivalent. This fact can be used for the following corollary.

Corollary 1: Let a multistage Shuffle-Exchange network be divided into 2^{n-r} groups of 2^r such that all PEs in a group have the same n-r most significant bits. Let G and H be two partitions as defined above. Then, G and H may be merged into one partition of size 2^{r+1} with no effect on the rest of the network.

Proof: This proof follows from Theorem 5 and the fact that the Generalized Cube and the multistage Shuffle-Exchange networks are topologically equivalent. In this case, allowing an exchange at stage r permits communication between G and H.

Since a recirculating Cube network performs the same functions as a multistage Cube network, two appropriate partitions of a recirculating Cube network can be merged. □

Corollary 2: Let a recirculating Cube network be partitioned into 2^{n-r} groups of 2^r PEs such that all PEs in a group have the same n-r most significant digits. Let G and H be two partitions of the system as defined above. Then G and H can be merged into one partition of size 2^r with no effect on the remainder of the network.

Proof: This follows from Theorems 3 and 5. G and H communicate by allowing $Cube_r$. □

Feng's data manipulator [4], which is an n-stage PM2I network, can be partitioned under single control partitioning into 2^{n-r} groups of 2^r PEs if the PEs in each group have the same n-r least significant digits [15]. Due to its restricted control structure, multiple control partitioning is not possible. In [15], an augmented data manipulator, capable of multiple control partitioning is described. Multiple control partitioning is possible for a recirculating PM2I network.

Theorem 6: Let a recirculating PM2I network be partitioned into 2^{n-r} groups of 2^r PEs where the PEs in each group have the same n-r least significant bits. Let G and H be two partitions. The addresses of the PEs in G are

$$P * 2^{n-r} + g_{n-r-1}\cdots g_1 g_0,\ 0 \le P < 2^r,$$

and the addresses of the PEs in H are

$$Q * 2^{n-r} + h_{n-r-1}\cdots h_1 h_0,\ 0 \le Q < 2^r.$$

Then, if for all i, $(n-r-1) > i \ge 0$, $h_i = g_i$ and $h_{n-r-1} \neq g_{n-r-1}$, then G and H can be merged into one partition of size 2^{r+1}, where the addresses of all PEs in the merged partition have the same n-r-1 least significant bits.

Proof: For the controls $PM2_{\pm i}$, $(n-r) \le i < n$, no group can send data outside of its group, since the n-r least significant bits never change for these 2*r functions. Therefore, multiple control partitioning is possible. Let G and H be merged into one partition such that the PE addresses have the same n-r-1 least significant bits. The new controls needed are $PM2_{\pm(n-r-1)}$. These two controls send data only between groups G and H, since the n-r-1 least significant digits are not affected. Therefore, PEs in G and H can exchange data without interfering with the rest of the network. So, for the recirculating PM2I network, G and H can be merged under multiple contol partitioning. □

Corollary 3: Let a multistage augmented data manipulator be divided into 2^{n-r} groups of 2^r such that all PEs in a group have the same n-r least significant bits. Let G and H be defined as in Theorem 6. Then, G and H may be merged into one partition of size 2^{r+1} with no effect on the rest of the network.

Proof: Follows from [15] and Theorem 6. □

VII. Conclusions

The Cube, Shuffle-Exchange, and PM2I networks were implemented as both recirculating and multistage networks. For each design, the amount of hardware and delay were calculated. Pipelined multistage networks were examined and compared to combinational logic multistage networks.

A new multistage network, the Shuffle-No Shuffle-Exchange, was introduced. It performs all the permutations of an n-stage Shuffle-Exchange network in the same time. It can also perform all the permutations of a recirculating Shuffle-Exchange network.

Partitioning interconnection networks for reconfigurable SIMD machines was considered. It was shown that pipelined and recirculating networks can be partitioned in the same way as their combinational logic multistage counterparts. Merging theorems were introduced which show that any network which can be partitioned into 2^{n-r} groups of 2^r PEs in a multiple control environment can be partitioned such that groups may be of varying sizes of powers of two.

References

1 Batcher, K. E., "The multidimensional access memory in STARAN," IEEE Trans. Comput., vol. c-26 (Feb. 1977), pp. 174-177.

2 Batcher, K.E. , "The flip network in STARAN," 1976 Int'l. Conf. Parallel Processing, (Aug.1976), pp. 65-71.

3 Couranz, G. R., M. S. Gerhardt, and C. J. Young, "Programmable RADAR signal processing using the RAP," 1974 Sagamore Computer Conf. on Parallel Processing (Aug. 1974), pp. 37-52.

4 Feng, T. , "Data manipulating functions in parallel processors and their implementations," IEEE Trans. Comput., vol. C-23 (Mar. 1974), pp. 309-318.

5 Higbie, L. C., "The Omen computer associative array processor," in Compcon 72, IEEE Computer Society Conf. Proceedings (Sept. 1972), pp. 287-290.

6 Lang, T., "Interconnections between processors and memory modules using the shuffle-exchange network," IEEE Trans. Comput., vol. c-25 (May 1976), pp. 496-503.

7 Lang, T., and Stone, H. S., "A shuffle-exchange network with simplified control," IEEE Trans. Comput., vol. c-25 (Jan. 1976), pp. 55-65.

8 Lawrie, D. H., "Access and alignment of data in an array processor," IEEE Trans. Comput., vol. c-24 (Dec. 1975), pp. 1145-1155.

9 Nutt, G. J., "Microprocessor implementation of a parallel processor," Fourth Annual Symp. Computer Architecture (Mar. 1977), pages 147-152.

10 Pease, M. C. , "The indirect binary n-Cube microprocessor array," IEEE Trans. Comput., vol. c-26 (May 1977), pp. 458-473.

11 Siegel, H. J. , "Analysis techniques for SIMD machine interconnection networks and the effects of processor address masks," IEEE Trans. Comput., vol. c-26 (Feb. 1977), pp. 153-161.

12 Siegel, H. J., "Single instruction stream - multiple data stream machine interconnection network design," 1976 Int'l. Conf. Parallel Processing (Aug. 1976), pp. 273-282.

13 Siegel, H. J., "The universality of various types of SIMD machine interconnection networks," Fourth Annual Symp. Computer Architecture (Mar. 1977), pp. 70-79.

14 Siegel, H. J., "Preliminary design of a versitile parallel image processing system," Third Biennial Conf. on Computing in Indiana (Apr. 1978), pages 11-25.

15 Siegel, H.J., and Smith, S. D., "Study of multistage SIMD interconnection networks," Fifth Annual Symp. Computer Architecture, (Apr. 1978), pp. 223-229.

16 Siegel, H. J., Mueller, Jr., P. T., Smalley, H. E., "Control of a partitionable multimicroprocessor system," 1978 Int'l. Conf. Parallel Processing (Aug.1978).

17 Stone, H. S., "Parallel processing and the perfect shuffle," IEEE Trans. on Comput., vol. c-20 (Feb. 1971), pp. 153-161.

18 Stone, H. S., "Parallel Computers," in Introduction to Computer Architecture, Science Research Associates, Inc., Chicago, Ill, 1975, pp. 318-374.

19 The Data Book for Design Engineers, Second Edition, Texas Instruments, Inc., 1976.