THE FLIP NETWORK IN STARAN [a]

Kenneth E. Batcher
Digital Technology Department
Goodyear Aerospace Corp.
Akron, Ohio 44315

Abstract - The flip network in each array module of STARAN scrambles and unscrambles multi-dimensional access (MDA) memory data. The flip network can permute data on transfers from memory to PE's, from PE's to memory, and from PE's to PE's. Among the allowable permutations are barrel shifts, barrel shifts on substrings, and FFT-butterflies. The network can be used for such data manipulations as shifting, mirroring (flipping end-for-end), irregular spreading, or compressing and replicating. These manipulations are useful for sorting, fast Fourier transforms, image warping, and solving partial differential equations on multi-mesh regions.

Introduction

An earlier paper (Ref. 1) describes the multi-dimensional access (MDA) memories in STARAN. Memory data can be accessed (fetched or stored) by words, by bit-slices, by byte-slices, etc. MDA memories are built with ordinary RAM chips, and data is scrambled a certain way when stored in memory so that it can be accessed in various ways.

A scramble/unscramble network is required to scramble the data when it is stored into memory and to unscramble the data when it is read from memory. The flip network (Figure 1) does the scrambling and unscrambling and can also perform a number of other useful permutations. Bauer (Ref. 2) has shown how a number of data manipulating functions can be performed using the flip network with appropriate PE masking.

Here, we show the construction of the flip network and then a method of irregularly spreading and compressing data that is faster than the method shown in Ref. 2.

Flip Network Construction

Notation

A 2^n -item flip network has 2^n input-data-lines labeled with n-bit binary vectors ranging from (00...00) to (11...11). It has 2^n output-data-lines also labeled with n-bit binary vectors. The network has two control inputs:

1. An n-bit flip control that specifies one of 2^n flip-permutations.

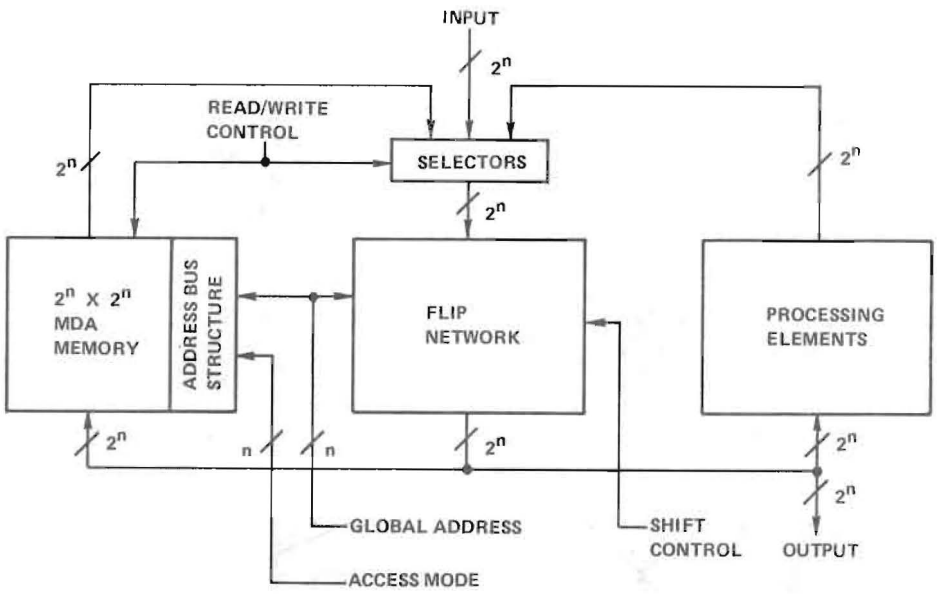

Figure 1. STARAN Array Module (n = 8)

[a] Trademark, Goodyear Aerospace Corporation, Akron, Ohio.

2. A shift-control that specifies one of $(n^2 + n + 2)/2$ shift-permutations.

The flip network permutes the input data first according to the specified flip-permutation, then according to the specified shift-permutation, and presents the permuted data on its output-data-lines.

To scramble and unscramble MDA memory data, the data is fed through the flip network while the flip-control is driven by the MDA memory global address to cause the desired flip-permutation.

Flip-Permutations

If $F = (f_{n-1}\ f_{n-2} \cdots f_1 f_0)$ is the n-bit binary vector fed to the flip-control, the flip-network moves the data on input-data-line $I = (i_{n-1}\ i_{n-2} \ldots i_1 i_0)$ to output-data-line $I \oplus F = (i_{n-1} \oplus f_{n-1}, i_{n-2} \oplus f_{n-2}, \cdots, i_1 \oplus f_1, i_0 \oplus f_0)$, where $\oplus$ means the exclusive-OR logic function.

Figure 2 shows the flip-permutations for an 8-item flip network. When F = (00...00), there is no permutation (the identity permutation); when F = (11...11), there is a complete reversal of data end-for-end (the mirror permutation). Each flip-permutation is its own inverse, and any two permutations commute with each other. If $F = F_1 \oplus F_2$, then flip-permuation F can be performed by doing permutation F_1 followed by F_2.

If the control input F has a single 1 and n-1 0's, then flip permutation F is called an atom (for the 8-item flip network, the atoms are (001), (010), and (100)). The set of n atoms forms a basis for all flip-permutations (any flip-permutation can be formed from atoms). This suggests one way of constructing flip networks. A 2^n-item flip network can be formed from n levels of logic. Each level is controlled by one of the flip-control bits and performs one of the atom permutations whenever the control bit is 1.

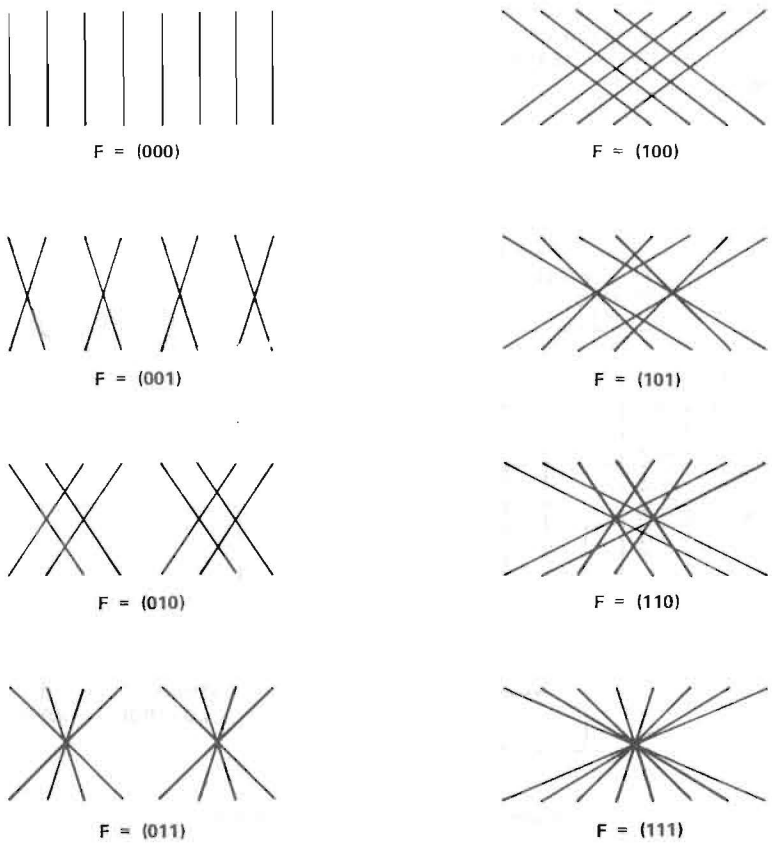

Figure 2. Flip Permutations for 8-Item Flip Network

Figure 3 shows an 8-item flip network constructed this way. The first level of logic performs flip-permutation (001) if the least-significant flip-control bit is 1 and identity permutation if the control bit is 0. Similarly, the second level does flip-permutation (010) when the middle control bit is 1 and the last level does flip-permutation (100) when the most significant control bit is 1. With this construction method, a 2^n-item flip network requires n levels of logic, with each level comprising 2^n two-way data selectors.

Figure 4 is an 8-item flip network redrawn to illustrate that the levels of data selectors are alike when the data is shuffled between levels. This means that a flip network can be built from a number of identical modules. It also means that the data can be recirculated n times through one level of data selectors if it is shuffled at each pass. Thus, one can use a shuffle-exchange network (Ref. 3) as a flip network.

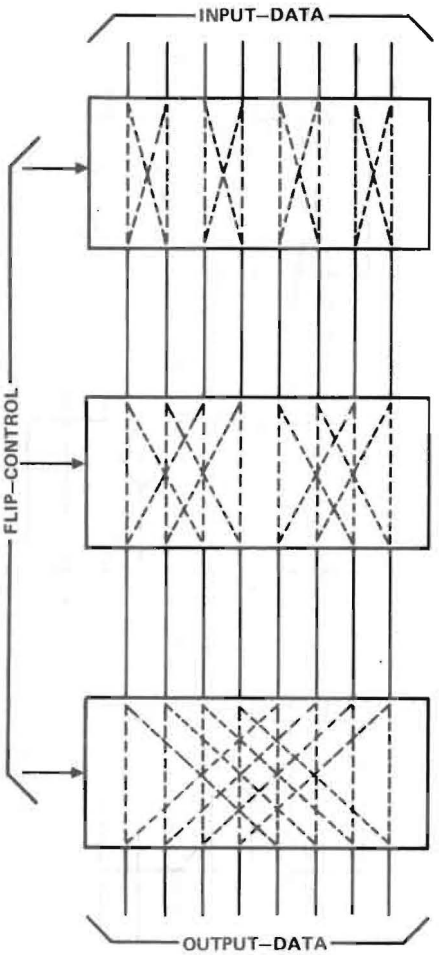

Figure 3. An 8-Item Flip Network

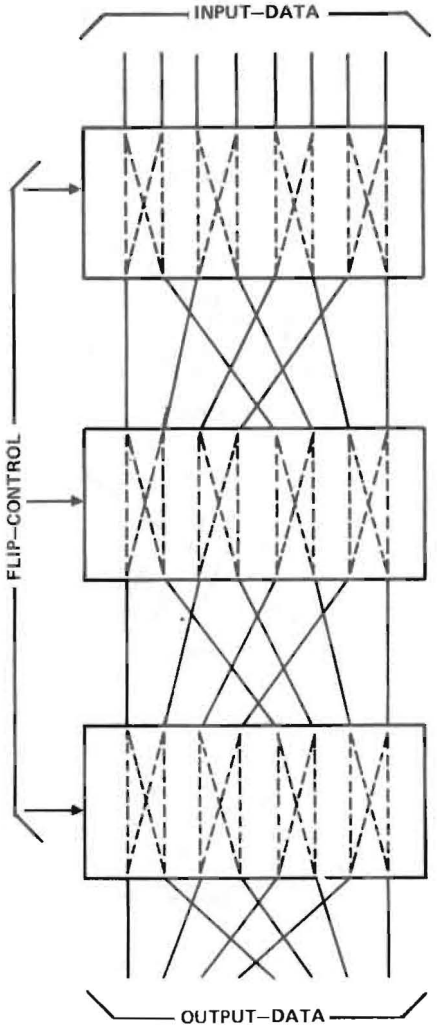

Figure 4. An 8-Item Flip Network Redrawn

One level of four-way data selectors can take the place of two levels of two-way data selectors. If n is even, a 2^n item flip network can be built from n/2 levels of four-way selection. The 256-item flip networks in the current STARAN each have four levels of four-way data selectors.

Shift-Permutations

The shift-control input to a 2^n-item flip network allows one of $(n^2 + n + 2)/2$ shift-permutations to be applied after any flip-permutation. One of the shift-permutations is the identity permutation (no shifting); the other $(n^2 + n)/2$ permutations are shifts of 2^m places modulo 2^p where m and p are integers so that $0 \leq m < p \leq n$. A shift of 2^m modulo 2^p divides the 2^n data items into groups of 2^p items each

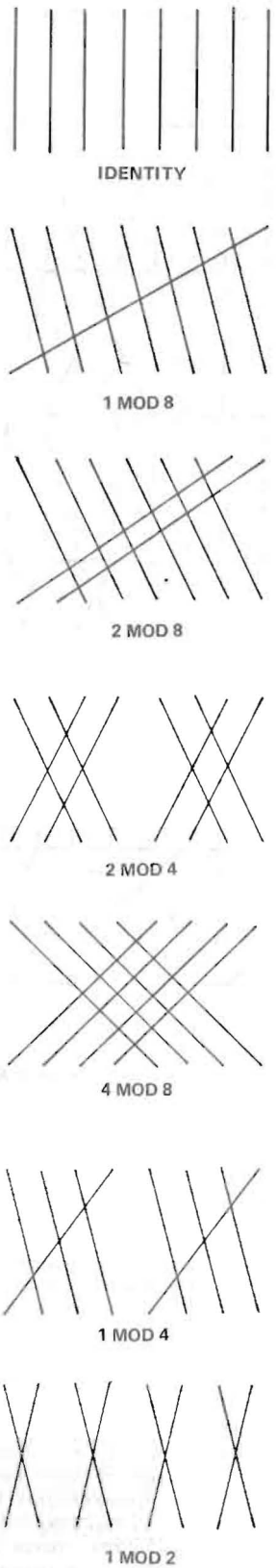

Figure 5. Shift Permutations in an 8-Item Flip Network

and shifts the items within each group right end-around 2^m places. Figure 5 illustrates the seven shift permutations in an 8-item flip network.

When $m = p - 1$, the shift-permutation of 2^m modulo 2^p is the same as a flip-permutation (compare the 1 mod 2, 2 mod 4, and 4 mod 8 shift-permutations of Figure 5 with the (001), (010), and (100) flip-permutations, respectively, of Figure 2). Other shift-permutations are performed in the flip network by selectively controlling the data selectors on certain levels. Figure 6 shows how a 1 mod 8 shift-permutation is performed in the 8-item flip network of Figure 3.

The selective control of data selectors on a level required for the shift-permutations is accomplished by expanding the number of control signals for the level; each control signal controls a fixed subset of the selectors on the level. With levels of two-way selectors, the first level has one control signal, the second level uses two

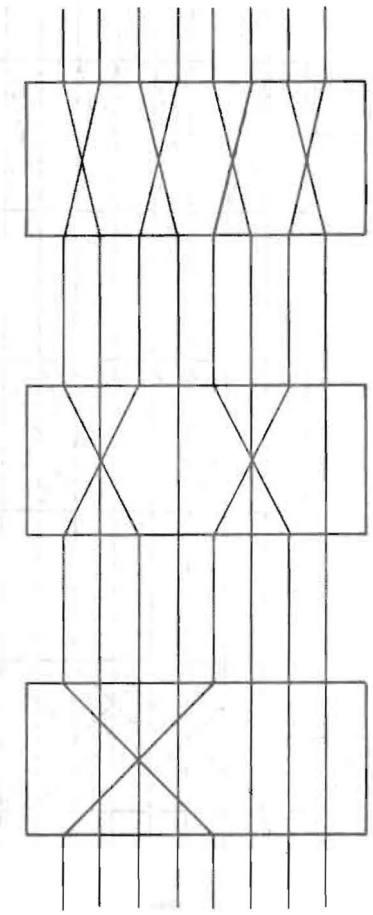

Figure 6. The 1 Mod 8 Shift Permutation in an 8-Item Flip Network

control signals, the third level uses three control signals, etc. A 2^n-item flip network requires n (n + 1)/2 control signals. Figure 7 shows how six control signals control the data selectors of an 8-item flip network so that both flip and shift permutations can be performed. The control table for this network follows (when the control signal is 1, the selectors swap data):

Permutation	Control Signal 0A	1A	1B	2A	2B	2C
1 mod 8	1	1	0	1	0	0
2 mod 8	0	1	1	1	1	0
4 mod 8	0	0	0	1	1	1
1 mod 4	1	1	0	0	0	0
2 mod 4	0	1	1	0	0	0
1 mod 2	1	0	0	0	0	0
Identity	0	0	0	0	0	0

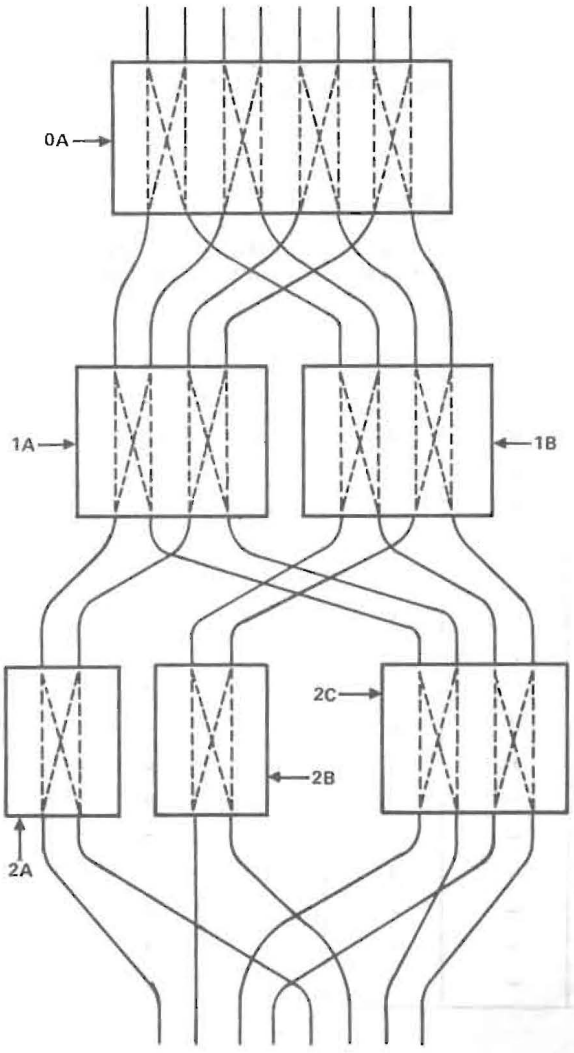

Figure 7. An 8-Item Network for Flip and Shift Permutations

For flip-permutations, 0A is driven by the least-significant flip-control bit; 1A and 1B are driven by the middle flip-control bit; and 2A, 2B, and 2C are driven by the most-significant flip-control bit.

To allow a combination of a flip-permutation with a shift-permutation in one pass through the network, each control signal is driven from an exclusive-OR gate. The shift-permutation control is fed to one exclusive-OR input and the flip-permutation control is fed to the other input. The resultant permutation is the same as the flip-permutation followed by the shift-permutation.

To shift data in a negative direction, one can mirror the data with a flip-permutation (11...11), shift the mirrored data in a positive direction, and then remirror the data with another (11...11) flip. The mirroring and remirroring can be combined with the shifts. For example, a shift of -31 places can be performed in two passes: A mirror with a shift of 32 followed by a mirror with a shift of 1.

Data Manipulations

General

The flip network can permute data on memory-to-PE transfers, PE-to-PE transfers, and PE-to-memory transfers. The permutations are useful in many applications to manipulate or route data between PE's. Bauer (Ref. 2) illustrates a number of these manipulations. Some manipulations require only one pass through the network; several require $\log_2 N$ passes for N items. One class of functions (irregular compress and expand) required about N passes for N items. Here, we show how these irregular functions can be accomplished in about $\log_2$ passes.

Irregular Spreading

Spreading (expanding, replicating) takes the output of a contiguous set of PE's and spreads it across a larger set of PE's, replicating some items but preserving their relative order. As an example, if we let

abcde (1)

represent the outputs of the first five PE's in order, then irregular spreading can create the following pattern of 19 items:

aaaa b ccccc ddddddd e (2)

in the first 19 PE's.

Spreading arises in a number of problems. To magnify a digitized image, new picture elements (pixels) are created on a finer grid; the old pixels must be spread and then interpolated to create the new image. This spreading is irregular if the image is being warped. Another

example is solving partial differential equations on multi-mesh regions; data computed on a coarse mesh must be spread and interpolated when moved across a boundary to a finer mesh.

In STARAN, spreading is accomplished with shift-permutations in the flip-network combined with appropriate PE masks. It will be illustrated with the example of spreading pattern (1) to obtain pattern (2). Figure 8 shows the state of the first 19 PE's at different steps of the process.

Initially, the five data items (a, b, c, d, and e) are stored in the first five PE's (0, 1, 2, 3, and 4, respectively). Each PE is to receive one of these items. The second column of Figure 8 shows the initial location of the item (e.g., PE's 5 through 9 are to receive item c, which initially is in PE 2).

In parallel, each PE computes a shift value, which is simply the difference between its own index and the initial location. This shift value is shown in the third column in binary notation. The maximum shift value is 14, which is less than 2^4; thus, four passes through the flip network are required to spread the data.

The first pass is a PE-to-PE transfer with a shift-permutation of 8 places. The bit-slice with weight 8 of the shift value is used as a mask; where the bit is 0, the PE retains its stored value and where the bit is 1 the PE accepts data from the flip network. The fifth column of Figure 8 shows the values stored in each PE after this pass. PE's 0 through 10 are masked off and do not change state; PE's 11 through 18 accept data from PE's 3 through 10, respectively.

The second pass is a shift permutation of 4 places with the weight 4 bit-slice of the shift value used as a mask. PE's 6 through 10 and 15 through 18 accept data from PE's 2 through 6 and 11 through 14, respectively. The sixth column of Figure 8 shows the result.

Similarly, two more passes are executed with shifts of 2 places and 1 place, respectively, and with the weight 2 and weight 1 bit-slices of the shift value as masks, respectively. The last column of Figure 8 shows the result; this is pattern (2).

As long as the shift value bit-slices are treated in the correct order (most-significant bit-slice first), spreading can be performed

			DATA VALUE				
PE INDEX	INITIAL LOCATION OF DATA	SHIFT VALUE 8 4 2 1	INITIALLY	AFTER 8 SHIFT	AFTER 4 SHIFT	AFTER 2 SHIFT	AFTER 1 SHIFT
0	0	0 0 0 0	a	a	a	a	a
1	0	0 0 0 1	b	b	b	b	a
2	0	0 0 1 0	c	c	c	a	a
3	0	0 0 1 1	d	d	d	b	a
4	1	0 0 1 1	e	e	e	c	b
5	2	0 0 1 1	–	–	–	d	c
6	2	0 1 0 0	–	–	c	c	c
7	2	0 1 0 1	–	–	d	d	c
8	2	0 1 1 0	–	–	e	c	c
9	2	0 1 1 1	–	–	–	d	c
10	3	0 1 1 1	–	–	–	e	d
11	3	1 0 0 0	–	d	d	d	d
12	3	1 0 0 1	–	e	e	e	d
13	3	1 0 1 0	–	–	–	d	d
14	3	1 0 1 1	–	–	–	e	d
15	3	1 1 0 0	–	–	d	d	d
16	3	1 1 0 1	–	–	e	e	d
17	3	1 1 1 0	–	–	–	d	d
18	4	1 1 1 0	–	–	–	e	e

Figure 8. Irregular Spread Example

without collisions. Data can be spread into 2^n PE's with n passes or less if all shift values are non-negative.

Spreads with negative shift values require a modified method. First, all shift values are biased by a positive constant so that they are all non-negative. Then, certain bit-slices of the shift value field are complemented (the bit-slices corresponding to 1 bits in the bias constant). The result is a shift value where some bit-slices have negative weights and some have positive weights. The spread algorithm is then followed except that negative shifts are performed whenever negative-weight bit-slices are used as masks. The negative shifts are done with mirrors (with mirrored PE masks). If the bias constant is odd, the least-significant shift-value bit-slice has a negative weight and then an extra pass through the flip network is required to remirror the data into normal order. Data can be spread into 2^n PE's with n + 1 passes at most.

Irregular Compressing

Compressing (closing) takes data items from a scattered set of PE's and packs them into a contiguous set of PE's while preserving their relative order. It is the inverse operation of spreading and can be performed by reversing the steps of a spread.

Conclusions

The flip network scrambles and unscrambles data for the MDA memory. It also can perform the PE-to-PE routing required for many problems.

There is close connection between the flip network and the perfect shuffle. One can implement any flip network permutation with a few passes through a shuffle-exchange network. In many applications like the fast-Fourier-transform, a shuffle is used to pair up certain items. One pass through a flip network will also pair up the same items; the pairs may be ordered differently, however.

Irregular spreading and compressing can be performed in a few passes through the network. These operations are useful in image warping, rotation, magnification, and resampling.

References

1. K. E. Batcher, "The Multi-Dimensional-Access Memory in STARAN." 1975 Sagamore Computer Conference on Parallel Processing, p. 167; also submitted for publication in the IEEETC Special Issue on Parallel Processing.

2. L. H. Bauer, "Implementation of Data Manipulating Functions on the STARAN Associative Processor." 1974 Sagamore Computer Conference, pp 209-227.

3. H. S. Stone, "Parallel Processing with the Perfect Shuffle." IEETC Vol. C-20, pp. 153-161 (February 1971).