DYNAMIC REROUTING TAG SCHEMES FOR THE AUGMENTED DATA MANIPULATOR NETWORK

ROBERT J. McMILLEN and HOWARD JAY SIEGEL

School of Electrical Engineering
Purdue University
West Lafayette, Indiana

ABSTRACT

The augmented data manipulator (ADM) is a multistage interconnection network designed for large-scale, parallel processing systems. This paper is an extension of an earlier work in which the use of the inverse ADM (IADM) network in an MIMD environment was investigated. Dynamically rerouting messages to avoid busy or faulty links is explored for both the ADM and IADM networks. Several schemes are presented. In some cases, there is no increase in tag overhead, but the switching elements are more complex. In other cases, the size of the routing tag is increased by one bit, but the switching elements are not as complex. A new broadcasting capability is developed that allows one processor to send a message to any number of other processors (with some restriction on the destination addresses). Finally, a scheme for dynamically rerouting a broadcast message is presented.

INTRODUCTION

Two basic classes of parallelism are SIMD and MIMD. The SIMD (single instruction stream–multiple data stream)[1] organization consists of a control unit, which broadcasts instructions to all processors. Each processor accesses data from its own local memory. The MIMD (multiple instruction stream–multiple data stream)[1] organization is such that all processors access both instructions and data from local memories. Both organizations require an interconnection network with which processors can communicate. The demands placed on the network by the two organizations are quite different. In an SIMD environment, permutation connections are generally required such that each input is connected to a unique output. In an MIMD environment, many random one-to-one and one-to-many (broadcast) connections are required.

PASM, a partitionable SIMD/MIMD system, is a proposed multimicroprocessor machine that can be partitioned to operate as one or more independent SIMD or MIMD machines of varying sizes.[2, 3] The PASM interconnection network must be able to operate in either SIMD or MIMD mode in addition to being partitionable. In choosing an interconnection network for PASM, two different types of multistage networks proposed in the literature are being investigated.

One type of network being considered is the cube. Networks having cube-type topologies include the generalized cube,[4] omega,[5] the indirect binary n-cube,[6] shuffle-exchange,[7] baseline,[8] SW-banyan ($S = F = 2$),[9] and the STARAN flip[10] (the equivalence of these networks was shown in Siegel and Smith[4] and Wu and Feng[11]). In cube-type networks, there is only one path from a given input port to a given output port. Although this property makes distributed control routing tag schemes simple, there is no possiblility of establishing a connection or of routing a message around a faulty node or link.

In both an SIMD environment and an MIMD environment, additional nodes and links in the network improve its ability to handle a large number of simultaneous requests. Thus, the cube-type networks do not have the same throughput potential as other networks with additional hardware (and additional cost), such as the Benes network[12] and the augmented data manipulator.[4]

An N-input Benes network consists of $2\log_2 N - 1$ stages of $N/2$ switching elements. It is capable of performing all $N!$ permutation connections. The major drawback is that $O(N\log_2 N)$ time is required to compute the switch settings.[13] This lengthy setup time makes it ill-suited to use in parallel processing systems.

The *data manipulator* multistage interconnection network was proposed for SIMD operations by Feng.[14] The *augmented data manipulator* (ADM) network[4] is a modified version of the data manipulator in which the individual switching elements can be controlled independently (Figure 1). The *inverse ADM* (IADM) network[15] is identical to the ADM network except that the stages are traversed in reverse order. Using these networks for SIMD processing has been considered,[4,15-18] as has using the IADM network in an

0149-7111/81/0000/0505$00.75 © 1981 IEEE

MIMD environment.[19] The purpose of this paper is to explore further the possibility of using either network in an MIMD environment. Being able to do so requires controlling the networks in a distributed way, because each switching element is to be set independently. Routing tags are used to distribute control of the network among the processors connected to it.

In a paper by McMillen and Siegel,[19] a distributed routing tag scheme for the IADM network was developed and some of its capabilities were explored. Also, a method for broadcasting to a power of 2 destinations was developed. In this paper, several schemes that dynamically reroute around busy or faulty switches for both the ADM and IADM networks are presented. Several schemes are provided to give the network implementer flexibility in design and the reader insight into the logical structure of the ADM and IADM networks. The general broadcasting scheme (for destinations that are a power of 2 in number) is described as well as a method for finding an alternate route for general broadcasts. The general broadcast scheme is modified so that a message can be broadcast to an arbitrary number of destinations (with some restrictions on their addresses). Finally, a procedure is presented that allows broadcast messages to be dynamically rerouted.

NETWORK DEFINITION

The ADM network is shown in Figure 1 for $N = 8$ in which $\underline{N} = 2^n$ is the number of processors in the system. Recall that the IADM network is identical, except that the input and output sides are reversed. The IADM, ADM, and data manipulator networks are based on the PM2I (plus-minus 2^i) interconnection functions.[20,21] It is assumed that each microprocessor in the MIMD machine is paired with its own memory module. The processors are connected to the network so that processor j is connected to input port j and output port j, $0 \leq j < N$.

In the following, a *stage* consists of N nodes (switching elements) and the $3N$ data paths that are connected to the inputs of a succeeding stage. At stage i of the ADM network, $0 \leq i < n$ ($\underline{n} = \log_2 N$), the first output of node j is connected to the input of node $(j - 2^i) \bmod N$ of the next stage; the second output is connected to the input of node j; and the third output is connected to the input of node $(j + 2^i) \bmod N$ (Figure 1). Because $(j - 2^{n-1})$ equals $(j + 2^{n-1}) \bmod N$, there are actually only two distinct data paths instead of three from each node in stage $(n - 1)$. There is an additional set of N nodes at the output stage.

ROUTING TAG SCHEME

In this section, the routing tag scheme[19] is summarized. Each routing tag requires only $(n + 1)$ bits. Although all possible paths cannot be represented, a tag can be found to route a message between any source/destination pair. This is because there is more than one route between most source/destination pairs (dynamic rerouting schemes will be developed in the next section).

Let $S = s_{n-1} \ldots s_1 s_0$ denote the source address, and $D = d_{n-1} \ldots d_1 d_0$ denote the destination address in which s_i and d_i are the ith bits of the respective addresses. An $(n + 1)$ bit routing tag is formed by computing the *signed magnitude* difference between the destination and the source:

$$T = t_n t_{n-1} \ldots t_1 t_0 = D - S.$$

The *sign bit* is t_n in which $t_n = 0$ indicates positive and $t_n = 1$ indicates negative. Bits $t_{n-1} \ldots t_1 t_0$ equal the absolute value of $D - S$; that is, the *magnitude* of the difference. Complementing the sign bit provides a tag from D to S (a return tag).

For example, if N equals 16, S equals 1011, and D equals 0100, then T equals 10111. To route a message through the network, stage i need only examine bits t_n and t_i in the routing tag. If $t_i = 0$, the straight connection is used regardless of the value of t_n. If $t_n = 0$ and $t_i = 1$, the $+2^i$ link is used; if $t_n = 1$ and $t_i = 1$, the -2^i link is used. If T equals 10111, and if a message enters the ADM network at stage 3, then the sequence of connections traversed from processor 11 to 4 is straight, -2^2, -2^1, -2^0. A route consisting of only straight or $+2^i$ connections is called *positive dominant*; a route consisting of only straight or -2^i connections is called *negative dominant*.

Given a source address S and a routing tag $T = t_n t_{n-1} \ldots t_0$, the value of the destination address D is calculated as[19]:

$$D = [S + (-1)^{t_n}(t_{n-1}2^{n-1} + \ldots + t_0 2^0)] \bmod N.$$

Two tags, T_1 and T_2, are equivalent if and only if they route a message from the same source address to the same destination address; that is, given

$T_1(S) \rightarrow D_1$, and $T_2(S) \rightarrow D_2$,

$T_1 \sim T_2$ if and only if D_1 equals D_2.

THEOREM 1. *Let A' denote the twos complement of A and assume that a processor never sends a message to itself; that is, $T \neq 0$. Then $T' \sim T$.*[19]

Theorem 1 allows conversion from a positive dominant tag to a negative dominant tag, and vice versa.

REROUTING

In the last section, it was shown that both a positive dominant and a negative dominant route can be found between any pairs of processors (except where S equals D; that is, T equals 0). Other paths exist that use both positive and negative links (as well as straight). For example, for $N = 16$, to route a message from processor 0 to processor 5 in the ADM network, the positive dominant route is straight, $+2^2$, straight, $+2^0$. The negative dominant route is -2^3, straight, -2^1, -2^0. A few of the mixed sign routes are:

1. Straight, $+2^2$, $+2^1$, -2^0
2. -2^3, -2^2, straight, $+2^0$
3. -2^3, -2^2, $+\hat{2}^1$, -2^0

To take advantage of this physical property, several rerouting schemes for the ADM and IADM networks are proposed: theorems 2, 3 and 4 and corollaries 1 and 2 are for the ADM network; theorems 5, 6 and 7 and corollaries 3, 4 and 5 are for the IADM network.

THEOREM 2. *In the ADM network, if a message becomes blocked by a straight link at stage i, $0 < i < n$, it can be sent on the nonstraight link whose sign is the same as that of the routing tag, if the low order i bits of the tag are not all 0. To compensate, the routing tag is then replaced by the equivalent tag of opposite dominance.*

Proof. Assume the routing tag is

$$T = t_n t_{n-1} \ldots t_i \ldots t_0, \text{ that } t_i = 0,$$

and that the straight link requested in stage i is unavailable; the message is sent on the $(-1)^{t_n}2^i$ link (the link whose sign is the same as the tag). The distance between where the message was sent and the desired destination is

$$2^i - (t_{i-1}2^{i-1} + \ldots + t_0 2^0).$$

Thus, the message needs to move this distance in the opposite direction. By definition of twos complement, the distance is $(t_{i-1} \ldots t_1 t_0)'$. Because the sign bit needs to be complemented and the high-order $(n - i)$ bits have already been interpreted by the network, the twos complement of the whole routing tag can be taken. With this method, no routing information is lost, so a return tag can still be formed by complementing the sign bit. It also allows each node to perform the same operation when rerouting a message instead of twos complementing some subset of the bits based on the node's stage address.

COROLLARY 1. *The rerouting procedure of theorem 2 can be repeated as often as conditions for rerouting are met.*

Proof. The changes made to the routing tag convert it to a valid equivalent tag.

THEOREM 3. *In the ADM network, if a message becomes blocked by a straight link at stage i, $0 < i < n$, it can be sent on the nonstraight link whose sign is the same as that of the routing tag, if the low order i bits of the tag are not all 0. To compensate for the reroute, t_{n+1}, an additional bit in the tag (initially 0), is set to 1 to indicate a reroute occurred. Whenever $t_{n+1} = 1$, t_j $(0 < j \leq i)$, is given a new interpretation; if $t_j = 1$, route the message on the $-(-1)^{t_n}2^j$ link and reset the reroute bit to 0; if $t_j = 0$, route the message on the $-(-1)^{t_n}2^j$ link and leave the reroute bit set to 1.*

Proof. Given a routing tag

$$T = t_{n+1} t_n t_{n-1} \ldots t_i \ldots t_1 t_0,$$

suppose t_i equals 0. Assume the requested straight link in stage i is unavailable. According to the theorem statement, t_{n+1} is set to 1, and the message is sent on the $(-1)^{t_n}2^i$ link. To see that the new tag interpretation is correct, the action taken in stage $i - 1$ is considered:

- *Case 1.* $t_{i-1} = 1$. The distance of the message from the source after being rerouted is:

$$(-1)^{t_n}(t_{n-1}2^{n-1} + \ldots + t_{i+1}2^{i+1} + 2^i).$$

After the compensation is applied in stage $i - 1$, the distance from the source is:

$$(-1)^{t_n}(t_{n-1}2^{n-1} + \ldots + t_{i+1}2^{i+1} + 2^i - 2^{i-1}) =$$

$$(-1)^{t_n}(t_{n-1}2^{n-1} + \ldots + t_{i+1}2^{i+1} + 2^{i-1}).$$

Since $t_i t_{i-1} = 01$, the message is now where it would have been if the reroute had not occurred.

- *Case 2.* $t_{i-1} = 0$ ($t_{i-2} \ldots t_0 \neq 0$). The message is sent on the $-(-1)^{t_n}2^{i-1}$ link in this case as well; thus, its distance from the source is also

$$(-1)^{t_n}(t_{n-1}2^{n-1} + \ldots + t_{i+1}2^{i+1} + 2^{i-1}).$$

Since $t_{i-1} = 0$, the net result is as though the message was rerouted in stage $i - 1$; thus, the reroute bit remains set to 1.

COROLLARY 2. *The rerouting procedure of theorem 3 can be repeated as often as conditions for rerouting are met.*

Proof. According to theorem 3, while rerouting is in progress, regardless of the value of the magnitude bits, nonstraight links are used. Since straight links must be requested for rerouting to occur, no further rerouting occurs while an earlier reroute is in progress. Rerouting terminates when the first 1 bit in the tag is encountered. When rerouting terminates, the message is back on the original path specified by the routing tag. Because there is no evidence of a previous reroute in the tag (reroute bit is reset to 0), rerouting can be repeated.

THEOREM 4. *Assume a message was rerouted in stage i of the ADM network using theorem 3. If the reroute bit, t_{n+1}, is still set when this message arrives in stage k, $i > k > 0$, then if $t_k = 1$ (and the remaining low order bits are not all 0), the message can be sent straight and reroute bit left set to 1.*

Proof. The theorem is proved by induction on the number of times the action in the theorem statement is applied between the setting and resetting of t_{n+1}. Let the period between setting and resetting of the reroute bit be called a *reroute cycle.* If there is just one 1 in the magnitude portion of the routing tag ($t_{n-1} \ldots t_1 t_0$), the conditions of theorem 4 cannot be met, so only theorem 3 is applied. For more than one 1, the goal in the following is to show that the rerouting process terminates prior to the message's exit from the network, despite being prolonged by the rerouting procedure of theorem 4.

1. Basis: The action of the theorem statement occurs once. This implies there are two 1's encountered in the magnitude of the routing tag during a reroute cycle, t_j and t_k, $n > i > k > j \geq 0$. Recall that $t_i = 0$ and that the message was rerouted in stage i. When the message arrives in stage k, because $t_k = 1$, theorem 4 can be applied, and the message is routed straight. The reroute bit, t_{n+1}, remains set to 1. The effects of this action are to postpone the completion of compensation. The distance travelled from stage i to stage k is:

$$d_{i,k} = (-1)^{t_n}(2^i - 2^{i-1} - \ldots - 2^{k+1}) = (-1)^{t_n}2^{k+1}.$$

Since $t_{k+1} = 0$ and $t_k = 1$, the distance travelled thus far does not yet correspond to the distance specified by the routing tag. To complete compensation for the reroute, 2^k must be subtracted from the distance travelled thus far. If theorem 3 were to be applied, the $-(-1)^{t_n}2^k$ link would be specified, and compensation would be applied immediately. In this case, however, the message is routed straight, so the distance travelled to stage j is:

$$d_{i,j} = (-1)^{t_n}(2^{k+1} - 2^{k-1} - \ldots - 2^{j+1}) =$$
$$(-1)^{t_n}(2^k + 2^{j+1}).$$

Note that since the reroute bit was left set to 1 and $t_r = 0$, $k > r > j$, the message was routed on the $-(-1)^{t_n}2^r$ links (theorem 3). Because t_j is the least significant 1 in the cycle, theorem 3 must be applied (the $-(-1)^{t_n}2^j$ link would be specified, the reroute bit reset to 0, and compensation completed).

2. Induction step: Assume that the action of the theorem statement can be applied $p - 1$ times. That implies that if there are p 1's encountered in the magnitude of the tag during a reroute cycle, the message will be routed properly. Suppose the action is applied p times and, thus, there are $p + 1$, 1's to be considered. Let t_k be the most significant 1 in the cycle and t_l be the next most significant 1, $n > i > k > l > 0$. Calculate:

$$d_{i,k} = (-1)^{t_n}(2^i - 2^{i-1} - \ldots - 2^{k+1}) = (-1)^{t_n}2^{k+1}.$$

As before, $d_{i,l} = (-1)^{t_n}(2^k + 2^{l+1})$.

Suppose for a moment that no reroutes occurred until stage r, $k > r > l$. The distance, $d_{i,l}$, would be $(-1)^{t_n}(2^k + 2^{l+1})$. Thus, from the viewpoint of a node in stage l, a tag with p 1's in the cycle is

being processed. By the induction hypothesis, this case will be handled properly.

THEOREM 5. *In the IADM network, if a message becomes blocked by a $\pm 2^i$ link, $0 \leq i \leq n - 2$, it can be sent on the oppositely signed link if the routing tag is replaced by the equivalent tag of opposite dominance.*

Proof. Assume that routing tag

$$T = t_n t_{n-1} \cdots t_i \cdots t_0,$$

that $t_i = 1$, and that the $(-1)^{t_n}2^i$ link is blocked; the message is sent on the $(-1)^{\bar{t}_n}2^i$ link. The distance the message travels after being rerouted (from stage 0 to stage $i + 1$) is:

$$(-1)^{t_n}(t_{i-1}2^{i-1} + \cdots + t_0 2^0) + (-1)^{\bar{t}_n}2^i =$$

$$(-1)^{\bar{t}_n}(t_{i-1} \cdots t_0)'.$$

The message arrives at the same node in the $(i + 1)$st stage to which it would have been sent if the equivalent tag of opposite dominance had been used. Having switched paths due to the reroute, the tag should be converted to its twos complement equivalent form to route the message to the desired destination correctly. As in theorem 2, no information is lost. The $n - 2$ bound on i is based on the assumption that the $+2^{n-1}$ and -2^{n-1} links use the same physical connection.

COROLLARY 3. *The rerouting procedure of theorem 5 can be repeated as often as conditions for rerouting are met.*

Proof. Same reasoning as corollary 1.

THEOREM 6. *If at stage i in the IADM network, a message is blocked by a $\pm 2^i$ link, $0 \leq i \leq n - 1$, it can be sent on the oppositely signed link if 2^{i+1} is added to the magnitude portion of the routing tag (mod N).*[19]

COROLLARY 4. *The rerouting procedure of theorem 6 can be applied as many times as the conditions for rerouting are met.*[19]

THEOREM 7. *The rerouting scheme of theorem 6 can be implemented without modifying the routing tag if one additional bit, t_{n+1}, is appended to the tag and initialized to 0. To perform a reroute at stage i, send the message on the oppositely signed link and set t_{n+1} to 1. If t_{n+1} is 1 when a message is received at a node in stage j, $i < j < n$, then:*

- *If $t_j = 1$, the message is routed straight, and t_{n+1} remains set.*
- *If $t_j = 0$, the message is routed on to the $(-1)^{t_n}2^j$ link, and t_{n+1} is reset to 0.*[19]

COROLLARY 5. *The rerouting procedure of theorem 7 can be applied as often as the conditions for rerouting are met. If a nonstraight link is required, but the carry bit, t_{n+1}, has already been set by a previous reroute, the message may still be rerouted by using the nonstraight link of the opposite sign and leaving t_{n+1} set.*[19]

Theorems 2 through 7 demonstrate the flexibility of the routing tag scheme and rerouting properties of the ADM and IADM networks that cube-type networks discussed in the literature do not have. The different schemes give a network implementer some freedom in designing the nodes of the network. Each scheme is equally capable of using the multiple paths between arbitrary source/destination pairs. The differences between the schemes affect routing tag overhead and node complexity.

The primary advantage to the scheme of theorems 2 (ADM) and 5 (IADM) is that it is self-contained and consequently does not contribute to routing tag overhead. The compensation that is applied to a routing tag because of a reroute is the twos complement operation. The operation is performed by the node that reroutes the message, and no other node is involved in the procedure. The drawback is that each node must contain enough extra hardware to perform the twos complement operation (such as $[n + 1]$ inverters and an $[n + 1]$ bit incrementer). This has more impact on a discrete logic design than on an LSI chip design. If the network is bidirectional, theorems 2 and 5 can be combined so that the same compensation operation is performed regardless of the message's direction of travel.

The scheme of theorem 6 (IADM) involves adding a power of 2 to the routing tag to compensate for rerouting. This also has the advantage of being self-contained, but the route information contained in the tag (that allows a receiver to determine the source address) is destroyed. To implement the scheme requires an n bit full adder for modifying the routing tag.

The theorem 7 (IADM) scheme is a modification of that developed in theorem 6. The schemes of both theorem 7 and theorem 3 (ADM) have the advantage that the only modification made to the tag is to set or clear one bit. Although the remaining bits of the routing tag are interpreted differently if the reroute bit is set, the new interpretation can be implemented with a small amount of combinational logic. In these two schemes, several nodes can be involved in the reroute compensation process; more than one node needs to make a different routing choice because the reroute bit is set. The disadvantage of these schemes is the routing tag overhead: an extra tag bit needs to be transmitted.

Theorem 4 (ADM) adds to the scheme of theorem 3, making it more flexible. The additional cost in hardware is small (again, just a few gates).

According to partitioning theory,[4, 17] if a size $N = 2^n$ ADM or IADM is partitioned into $Q = 2^q$ independent subnetworks of N/Q input/output ports each, the addresses of all input/output ports in the same partition must agree in the $(n - q)$ low-order bits. This implies that all routing tags used for communication among processors in the same partition will be 0 in their $(n - q)$ low-order bits. Under such an assignment, the partitioning is enforced by setting all switching elements in the $(n - q)$ low-order numbered stages of the network to straight. In all rerouting schemes presented, tags containing consecutive low-order 0's are never rerouted in the corresponding stages of either network; these stages remain set to straight. This is completely compatible with the partitioning rules. A message sent between an input port and an output port in the same subnetwork may be rerouted using any of the schemes described and will still remain within its subnetwork.

BROADCASTING

Allowing a source to send data to a power of 2 destinations simultaneously is discussed in this section. The general broadcasting scheme described applies to both the ADM and IADM networks. A method is developed for modifying that scheme so that broadcasts to a number of destinations that is not a power of 2 can be performed. The details of that method presented here are specific to the IADM network; however, a similar method exists for the ADM network. The IADM method was chosen for presentation because it is more intuitive in nature.

The general broadcasting scheme is given in theorem 8. Theorem 9 shows how to compute an alternate route for a general broadcast routing tag. The method for broadcasting to a nonpower of 2 destinations is then developed. Finally, theorems 10 and 11 specify how to reroute a broadcast in the ADM and IADM networks, respectively.

Broadcasting will be discussed in terms of sending a broadcast tag from the source to its destinations. Whether this tag is a header on a packet or is used to establish a circuit is irrelevant.

A node in the network is capable of performing two kinds of broadcasting or branching. A branch using the straight and $+2^i$ links or the straight and -2^i links is called a *2^i-type branch*; a branch using the -2^i and $+2^i$ links is called a *2^{i+1}-type branch*.[19] Only 2^i-type broadcasts are considered here.

A *broadcast tag* is denoted by $\{R, B\}$, in which R is an $(n + 1)$ bit *routing control* word, and B is an n bit *broadcast mask*.[19] R corresponds to the routing tag defined earlier (T). A broadcast tag is interpreted as follows:

- If the ith bit of B, b_i, is 0, no broadcast is performed in the ith stage, and R is treated exactly as T would be (interpreted as a normal route).
- If b_i is 1, the ith bit of R is ignored, and a broadcast is performed.
- The sign bit of R, r_n, determines which nonstraight link to use.
- For $C = c_{n-1} \ldots c_1 c_0$, the notation $c_{a/b}$ will indicate bits c_a through c_b of C, $a \geq b$.

THEOREM 8. *Let $X = x_{n-1} \ldots x_1 x_0$. Source processor S can broadcast to the 2^j destination processors whose addresses are:*

$$x_{n-1/i+j}0 \ldots 00x_{i-1/0},$$
$$x_{n-1/i+j}0 \ldots 01x_{i-1/0},$$
$$\vdots$$
$$x_{n-1/i+j}1 \ldots 11x_{i-1/0},$$

if the broadcast tag $\{R, B\}$ is such that $R = X - S$ and bits i through $(i + j - 1)$ of B are set to 1 (all others are set to 0).[19]

An example of a broadcast to four destinations is shown in Figure 2 for S equals 10110 and X equals 01000.

DEFINITION 3. *Two broadcast tags, $\{R, B\}$ and $\{R^1, B^1\}$, are equivalent ($\{R, B\} \sim \{R^1, B^1\}$) if and only if, given the same source address, a message routed by the tags arrives at the same set of processors.*

THEOREM 9. *Given a broadcast tag $\{R, B\}$, let $R = r_n \ldots r_0$ and $B = b_{n-1} \ldots b_0$. If $r_{i-1/0} \neq 0$ and b_i through b_{i+j-1} in which $i \leq (i + j - 1) < n$ are 1, and all other bits b_k are 0, then a new tag $\{R^1, B\}$, such that $\{R^1, B\} \sim \{R, B\}$, is constructed as follows in which $R^1 = \rho_n \ldots \rho_0$:*

$$\rho_n = \bar{r}_n,\ \rho_{n-1/i+j} = \bar{r}_{n-1/i+j}$$

$$\rho_{i+j-1/i} = r_{i+j-1/i},\ \rho_{i-1/0} = (r_{i-1/0})'.^{19}$$

In the following, a scheme is presented that modifies the broadcast routing tag scheme to allow broadcasting to an arbitrary number of destinations. This is accomplished by reducing the number of destinations in a set of 2^i, specified by the general broadcast theorem (theorem 8). The restrictions on valid destination addresses implied by theorem 8 still apply. No additional bits have to be added to the broadcast routing tag. More processing is required of the nodes, however, which increases their complexity. Initially, only broadcast tags whose routing control word is positive dominant are discussed. (The extension to negative dominant routing control words follows.)

To modify the general broadcasting scheme, the binary tree representing the paths used in a broadcast in the IADM network shown in Figure 3 is used. To reduce the number of destinations, it is necessary to "prune" an appropriate number of lower branches (represented by broken lines) from the tree. A *pruning node* is defined to be any node in the last stage of those stages involved in performing a broadcast that participates in the decision process of whether or not to prune a branch. To perform the pruning, a mechanism is needed to *instruct* the nodes that would route data to those branches. The method used is to include with the broadcast routing tag a counter that is decremented in accordance with the path followed. The positive links in stage i are given weights of 2^i. A message routed on the straight link is left with its counter unaltered. A message routed onto a positive link has its counter reduced by 2^i. A counter may not become negative, and so it is reduced to 0 if the amount to be subtracted is larger than the count. It is the counter value of 0 that instructs a pruning node to abort its broadcast and route a message only on the straight link.

To be initialized, the counter should contain the number of pruning nodes that are to participate in the broadcast. To calculate the initial value of the counter, the difference between the number of processors to receive the broadcast and the largest power of 2 less than that number is found. For example, to broadcast to 12 processors, the counter should be initialized to $12 - 2^3 = 4$.

To include a counter in the broadcast routing tag without adding any extra bits requires two observations:

1. Recall from theorem 8 that wherever the broadcast mask contains 1's, the corresponding bits of the routing tag have no significance and are thus unused. Under the scheme presented there, if there are $M = 2^m$ destinations, m bits of the routing tag are unused.

2. If P is the number of destinations to receive a broadcast under the new scheme, a tree with 2^p leaves is to be pruned in which $p = \lceil \log_2 P \rceil$. There are 2^{p-1} pruning nodes involved in deciding whether or not to perform a broadcast. If $M = 2^p$, there will always be enough unused bits in the routing tag R to accommodate the counter.

To be compatible with the alternate route scheme of theorem 9 and with the broadcast rerouting schemes to be presented in theorems 10 and 11, broadcast tags with negative dominant routing control words are handled differently. A pruning node receiving a negative broadcast tag whose counter is 0 will send the message on the negative link only. Also, the straight links are given a weight of 2^i in stage i, and the negative links a weight of 0. The different treatment of negative broadcast tags is required to guarantee that the same set of destinations is generated by positive or negative broadcast tags.

The algorithm shown in Figure 4 is used by each node to interpret a broadcast routing tag. In theorem 8, the broadcast mask bits equal to 1 were assumed to be adjacent. The same assumption is made in the procedure to be used by a node in stage i. It is assumed that stage j contains the pruning nodes. Also note that

(R AND B) = 0

implies the counter is 0. In Figure 3, the value of the counter for a broadcast to 12 processors is shown at each node as it is received.

The additional broadcast capability just described for the IADM network increases the complexity of nodes in the network, but does not require any extra connections between nodes. If the nodes are implemented as LSI chips, adding this feature is strongly recommended. The functional complexity of the chip is increased without the addition of any pins.

As with the IADM scheme, the ADM scheme is such that no additional bits are required in the broadcast tag (a similar counter is inserted into the unused bits of the routing control portion of the tag). Because of the difference in the way branching occurs in the ADM network, broadcasting to nonpowers of 2 destinations requires a different pruning procedure.

Two theorems remain to be proved, and they deal with rerouting broadcast messages. Theorem 10 is an extension of theorem 9 and handles rerouting in the ADM network. The method presented is appropriate for both general broadcasts as specified by theorem 8 and those to nonpowers of 2 as just described. Theorem 11 handles rerouting for the IADM network.

One property of twos complement numbers that will be referred to as P1 can be stated as follows: Let T be an n-bit number represented by $t_{n-1} \ldots t_1 t_0$. If t_i is the least significant 1 bit in T, then $t'_{n-1/i+1} = \overline{t}_{n-1/i+1}$, and $t'_{i/0} = t_{i/0}$. This property is used in the following theorems:

THEOREM 10. *The ADM network can reroute a broadcast message as long as the node that performs the reroute is not also required to broadcast the message.*

Proof: As long as a given bit of the broadcast mask is 0, the routing tag portion is interpreted such that theorem 2 can be applied. The only modification required is that the bits of the routing tag that correspond to 1's in the broadcast mask are left unmodified. If conditions for rerouting are met, the broadcast routing tag is converted to its alternate form according to theorem 9. Because of property P1, the conversion specified in theorem 9 is nearly identical to the conversion for routing tags (nonbroadcast) specified in theorem 1. The difference is that bits containing the broadcast counter are left intact. In the ADM network, after the broadcast message has been completely replicated, any of the copies can be rerouted according to theorems 2, 3, or 4.

THEOREM 11. *The IADM network can reroute a broadcast message as long as the node that performs the reroute is not also required to broadcast the message.*

Proof: The reasoning is similar to theorem 10 and follows from theorems 5 and 9 and property P1. In the IADM network, after the broadcast message has been completely replicated, the various copies can be rerouted according to any of theorems 5, 6, or 7.

CONCLUSIONS

Data routing schemes for the ADM and IADM networks operating in an MIMD environment were presented. The schemes provide the user much flexibility. They take advantage of the structure of the network to offer features not found in other networks. If a message is blocked because of a busy node or a faulty node or link, it can often automatically be rerouted. This capability provides fault tolerance as well as improved throughput. Three schemes each were presented for the ADM and IADM networks. Each scheme is equally capable of using the multiple paths that exist between sources and destinations. The differences, however, give the network designer flexibility.

A capability was developed that removes the restriction that the number of broadcast destinations be a power of 2. This capability was added without increasing the number of bits in a broadcast tag.

All schemes presented in this paper are dependent only on the logical structure of the network. They are equally well suited to different physical realizations. For example, in a circuit-switched implementation, the routing tags are used to establish paths through the network. In a packet-switched implementation, the routing tags are used to guide messages from stage to stage through the network. Many of the schemes discussed in this paper have been simulated and verified.

The study presented here is intended to aid system designers who must choose an interconnection network to meet their particular needs. By exploring the properties of the network in a way that is independent of a specific implementation, the information finds applicability in many different scenarios.

ACKNOWLEDGMENT

This work was supported by the Air Force Office of Scientific Research Air Force Systems Command, USAF, Grant No. AFOSR-78-3581. The United States Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

REFERENCES

[1]M. J. Flynn. Very high-speed computer systems, *Proceedings of IEEE*, 54 (Dec. 1966), 1901-1909.

[2]H. J. Siegel, P. T. Mueller, Jr., and H. E. Smalley, Jr. Control of a partitionable multimicroprocessor system, *1978 International Conference on Parallel Processing* (Aug. 1978), 9-17.

[3]H. J. Siegel et al. PASM: A partitionable multimicroprocessor SIMD/MIMD system for image processing and pattern recognition (Report TR-EE 79-40, Electrical Engineering), Purdue University, 1979.

[4]H. J. Siegel and S. D. Smith. Study of multistage SIMD interconnection networks, *5th Symposium on Computer Architecture* (April 1978), 223-229.

[5]D. H. Lawrie. Access and alignment of data in an array processor, *IEEE Transactions on Computers*, C-24 (Dec. 1975), 1145-1155.

[6]M. C. Pease. The indirect binary n-cube microprocessor array, *IEEE Transactions on Computers*, C-26 (May 1977), 458-473.

[7]T. Lang and H. S. Stone. A shuffle-exchange network with simplified control, *IEEE Transactions on Computers*, C-25 (Jan. 1976), 55-65.

[8]C-L. Wu and T-Y. Feng. Routing techniques for a class of multistage interconnection networks, *1978 International Conference on Parallel Processing* (Aug. 1978), 197-205.

[9]L. R. Goke and G. J. Lipovski. Banyan networks for partitioning multiprocessor systems, *1st Symposium on Computer Architecture* (Dec. 1973), 21-28.

[10]K. E. Batcher. The flip network in STARAN, *1976 International Conference on Parallel Processing* (Aug. 1976), 65-71.

[11]C-L. Wu and T-Y. Feng. On a class of multistage interconnection networks, *IEEE Transactions on Computers*, C-29 (Aug. 1980), 694-702.

[12]V. E. Benes. *Mathematical Theory of Connecting Networks and Telephone Traffic* (New York: Academic Press, 1965).

[13]D. C. Opferman and N. T. Tsaa-Wu. On a class of rearrangeable switching networks, *Bell Systems Technical Journal*, 50 (May-June 1971).

[14]T. Feng. Data manipulating functions in parallel processors and their implementations, *IEEE Transactions on Computers*, C-23 (Mar. 1974), 309-318.

[15]S. D. Smith et al. Use of the augmented data manipulator multistage network for SIMD machines, *1980 International Conference on Parallel Processing* (Aug. 1980), 75-78.

[16]H. J. Siegel. Interconnection networks for SIMD machines, *Computer*, 12 (June 1979), 57-65.

[17]_____. The theory underlying the partitioning of permutation networks, *IEEE Transactions on Computers*, C-29 (Sept. 1980), 791-801.

[18]H. J. Siegel and R. J. McMillen. Using the augmented data manipulator network in PASM, *Computer* 14 (Feb. 1981), 25-33.

[19]R. J. McMillen and H. J. Siegel. MIMD machine communication using the augmented data manipulator network, *7th Symposium on Computer Architecture* (May 1980), 51-58.

[20]H. J. Siegel. Analysis techniques for SIMD machine interconnection networks and the effects of processor address masks, *IEEE Transactions on Computers*, C-26 (Feb. 1977), 153-161.

[21]_____. A model of SIMD machines and a comparison of various interconnection networks, *IEEE Transactions on Computers*, C-28 (Dec. 1979), 907-917.

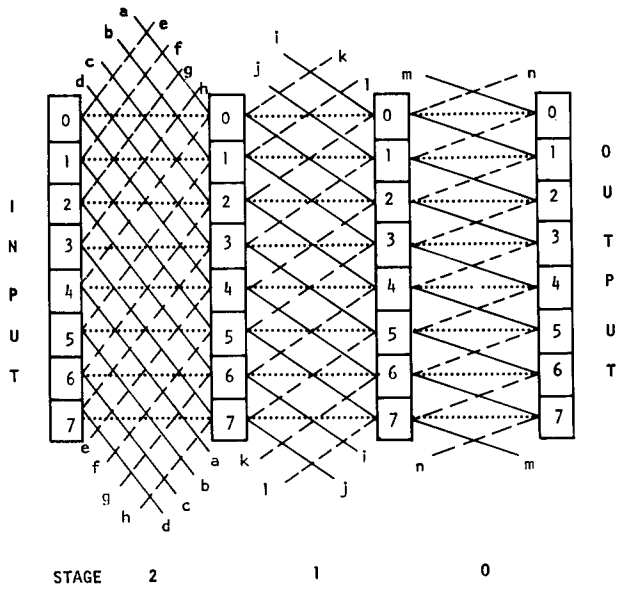

Figure 1. Augmented Data Manipulator Network for N = 8 (Lowercase letters represent end-around connections.)

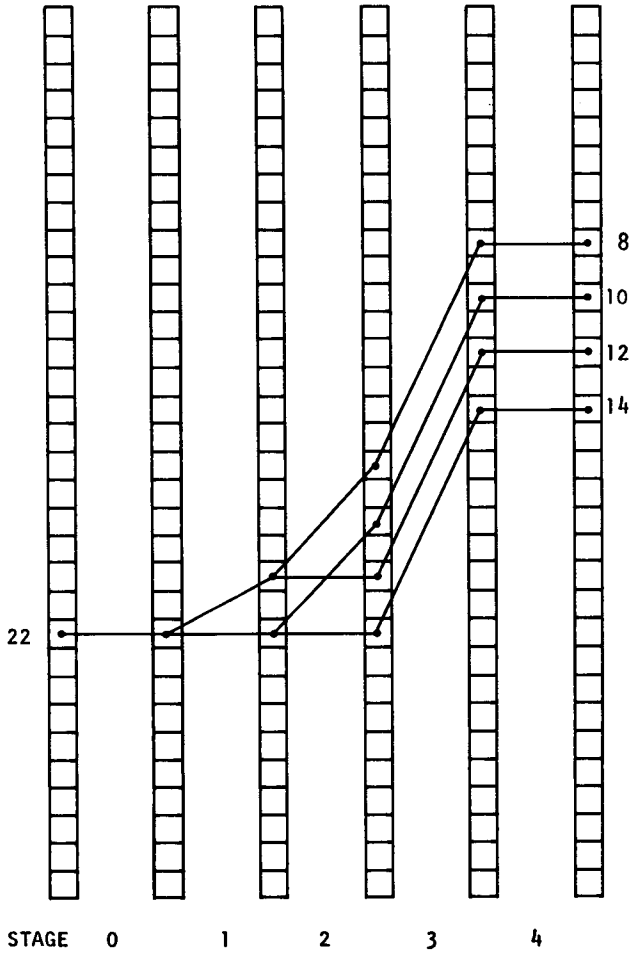

Figure 2. Broadcast to Four Processors (R = 101110, B = 000110; N = 32)

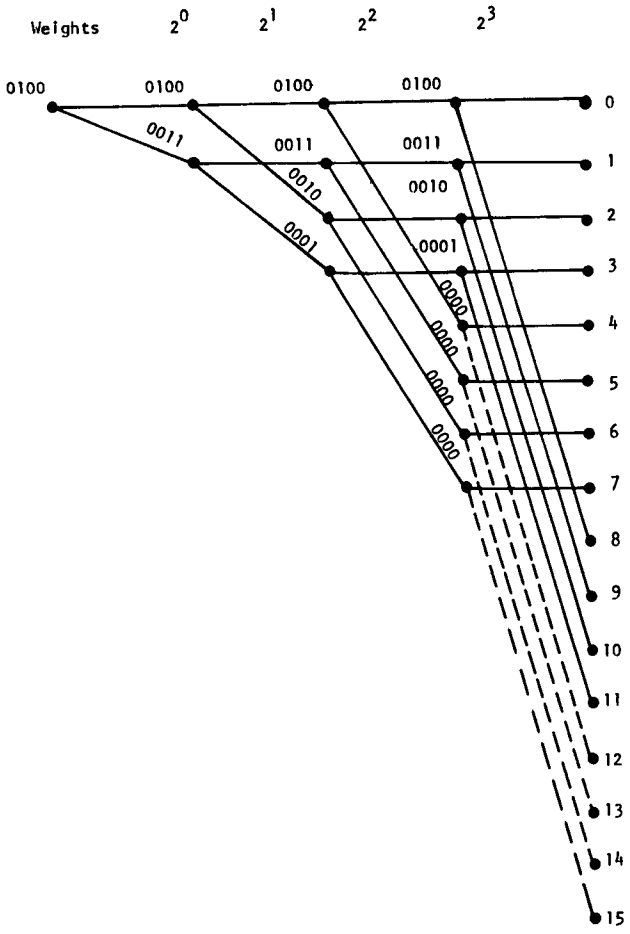

Figure 3. Broadcast to 12 Destinations Showing Value of Broadcast Counter as It Enters each Node (A node does not replicate message if counter is 0.)

```
if b_i = 1
  then
    if r_n = 0
      then
        if (R AND B) = 0 and i = j
          then send message straight only
          else subtract 2^i from R AND B setting
               the counter bits to 0  if the
               result is negative.  If not,
               subtract 2^i from R, then send
               this copy +2^i.  Send an unmodi-
               fied copy straight.
      else
        if (R AND B) = 0 and i = j
          then send message -2^i only
          else subtract 2^i from R AND B setting
               the counter bits to 0 if the
               result is negative.  If not,
               subtract 2^i from R, then send
               this copy straight.  Send an un-
               modified copy -2^i.
```

Figure 4. Algorithm Used by Node in Stage *i* of IADM Network to Interpret a Broadcast Tag (Tag can specify arbitrary number of destinations, and pruning nodes are in stage *j*.)