An Algorithm for the Accurate Reliability Evaluation of Triple Modular Redundancy Networks

JACOB A. ABRAHAM, STUDENT MEMBER, IEEE, AND DANIEL P. SIEWiorek, MEMBER, IEEE

Abstract—There are several instances where the classical method of triple-modular redundancy (TMR) reliability modeling may provide predictions which are inadequate. It is shown that for even simple networks such as those exhibiting fan-in and fan-out, classical methods may predict a reliability that is higher than or lower than the actual reliability. Furthermore, the classical method gives no hint as to whether the predicted number is high or low. As a solution to this problem, a method of partitioning an arbitrary network into cells such that faults in a cell are independent of faults in other cells is proposed. An algorithm is then given to calculate the reliability of any such cell, by considering only the structure of the interconnections within the cells. The value of the reliability found is exact if TMR is assumed to be a coherent system. An approximation to the algorithm is also described; this can be used to find a lower bound to the reliability without extensive calculation.

Index Terms—Coherent system, N-tuple modular redundancy, reliability modeling, serial cell, triple modular redundancy (TMR).

I. INTRODUCTION

The rapidly increasing application of computers to areas where the loss of real-time computing power could be catastrophic has brought with it the need for very high reliability. Even if computers are constructed from highly reliable components these components will still have a finite probability of failure. Thus highly reliable operation necessitates the use of some form of redundancy. Redundancy has been defined as the existence of more than one means of performing a function [1]. This could be brought about by providing extra time to perform the function, or by extra hardware within the computer, or by both.

Whereas accelerated life tests on many copies of a component may be feasible to experimentally determine the reliability of a component as a function of time, computer systems are too complex and often too expensive to subject to such tests. Thus to evaluate and compare various redundant system designs, a reliability modeling technique is required. With such a model it becomes possible to predict system behavior and, in particular, determine whether the proposed system meets the design specifications.

If the reliability model is accurate, then insights may be gained as to how the system reliability changes as a function of the design parameters. This requires knowledge of the model's predictive properties under all possible system designs, i.e., is it an upper bound, a lower bound, or simply a "good guess?" The following discussion will illustrate some common network configurations where the reliability modeling techniques in the literature for triple modular redundancy (TMR) are sometimes inadequate predictors of system reliability.

II. RELIABILITY MODELING OF TMR NETWORKS

A. Introduction

With the introduction of the restoring organ by von Neumann in 1956 [2] the groundwork was laid for the TMR technique. Briefly, TMR consists of dividing a non-redundant circuit into several modules, triplicating the modules, and inserting a majority gate (sometimes referred to as a voter) between the triplicated modules. This can be generalized to an N-tuple modular redundant system [(NMR) [3]] having $N = 2t + 1$ modules and voters, each voter being a $t + 1$ or more out of $N$ voter. In Fig. 1, for example, a nonredundant network divided into modules is shown in Fig. 1(a). If the modules are triplicated and connected to triplicated voters the TMR version of the network results in Fig. 1(b).

Fig. 2 shows TMR in its simplest configuration—triplicated modules followed by triplicated voters. Networks whose nonredundant form may be represented by a serial cascade of modules will be referred to as serial TMR. Fig. 2 outlines a serial TMR cell. Normally the input and output lines for a module represent buses. Thus the voter symbol as well as the module symbol should be thought of as operations on vectors rather than on single bits of information.

Historically and practically TMR is an important redundancy technique to study. TMR augmented by standby spares is a prime candidate for hard cores in self-repairing computers [3]. It has been used on the Saturn V
launch vehicle computer [4]. TMR is easy for a designer to apply and has several good features [5]. Also, TMR serves as a benchmark against which other redundancy schemes are often compared. Thus a thorough understanding of TMR reliability modeling is necessary.

B. Previous Work

Several investigators have addressed the problem of modeling the reliability of TMR or multiple-line networks [5]-[15]. There have been two basic approaches. The first approach was to approximate the system by a serial TMR system, i.e., modeling the network as a cascade of single-input single-output modules, adding extra voters if required. This was the essence of the procedures developed by Brown [8], Teoste [9], Rhodes [10], Longden [11], Lyons [15], and Gurzi [7].

A variation of this first approach, due to Rubin [12], models networks as serial cells and inserts fictitious module trios where required to make all the cells serial cells. Then he alters the standard serial voter-module reliability formula to approximate the effect of these added fictitious modules.

The second basic approach is to develop a bound on the system reliability by treating TMR as a coherent system. Esary and Proschan [16] defined the concept of coherent systems. A coherent system is one in which, having once failed, the system or component cannot work properly again. A network cut is defined to be a set of components whose failure causes system failure. A minimal cut is a cut from which no members can be deleted without the set losing the property of being a network cut. The probability obtained by taking the product, over all minimal cuts, of the probability that the cut does not occur is a lower bound on coherent network reliability [16]. Jensen [13] uses matrix manipulation to establish the minimal cuts of a network. However, if there are \( n \) modules in the nonredundant network, Jensen's method in the worst case requires on the order of \( n^3 \) operations and on the order of \( n^2 \) storage locations just to set up the matrices for determining the minimal cuts. The algorithm to be presented...

Fig. 1. (a) Nonredundant network. (b) TMR version of Fig. 1(a).
here will calculate the exact classical reliability of TMR networks (i.e., the reliability of a coherent system as defined by Esary and Proschan [16]).

III. CALCULATING THE EXACT COHERENT RELIABILITY OF A TMR NETWORK

A. Introduction

The algorithm we present will calculate the exact reliability assuming TMR is a coherent system. The basic assumptions when treating TMR as a coherent system are as follows.

Assumption 1: Once a module or voter has failed it will always give an incorrect output.

Assumption 2: Once a module has a failed input its output is also failed.

It should be noted that TMR is not a coherent system when considering failure modes other than complete failure, as for example with one input to a voter stuck-at-1 and another stuck-at-0. This is because compensating failures such as these permit three states for the system and its components, while a coherent system can have only two states: functioning and failed. These compensating failures can be incorporated into the reliability model at the expense of more computation time [19]. The coherent system reliability calculated is thus a lower bound on actual system reliability.

For the remainder of this discussion the reliability of a system (module) will mean the conditional probability that the system (module) will be capable of performing its specified function at time \( t \) given that all system (module) components are functioning properly at time \( t = 0 \).

In the subsequent formulas time is an implicit variable. To calculate system reliability at time \( t \) the module reliability must be evaluated at time \( t \).

Our approach is to partition an arbitrary TMR network into independent “cells” so that a failure in one cell cannot combine with a failure in another cell to cause system failure. The reliability of each cell is found and the reliability of the whole network is found from the cell reliabilities. This is much simpler than finding the reliability of the whole network at one time. If there are \( N \) modules in a network which can be partitioned into \( n \) independent cells of \( m \) modules each, where \( N = m \cdot n \), and if the complexity of the reliability evaluation algorithm is a function \( \psi \) of the number of modules, it is easily seen that \( n \cdot \psi(m) \ll \psi(m \cdot n) \), especially when \( \psi \) is exponential, as is usually the case. Also, this method is a specialized one for TMR, and takes advantage of the known properties of TMR.

Reconsider Fig. 1 where the nonredundant network [Fig. 1(a)] and its TMR counterpart [Fig. 1(b)] are depicted. Each of the triplicated modules or voters will be referred to as a module or voter trio, and each module or voter in a trio is said to occupy a particular position in the trio. It is to be noted that the modules need not be a single-output module, and that there need not be voters after every module trio. System failure in a TMR system occurs when there are two or more errors in any of the (triplicated) output lines. Under Assumption 1, system failure will occur if any of the module or voter trios have more than one failed module or voter. Assumption 2 implies that system failure can also occur if more than one module or voter in a trio has a wrong input or if one module in a trio is failed and another has a failed input. The reliability of a network is then the probability that system does not have one of these failure modes.

B. Partitioning a Network into Cells

We will now discuss the partitioning of an arbitrary network which simplifies the task of reliability evaluation.
In a TMR network a voter or module trio \( p \) is defined to be directly connected to a voter or module trio \( q \) if a single fault in a particular position of \( p \) allows only the single fault in the corresponding position of \( q \) without causing system failure. In Fig. 3 for example, if a single fault occurs in voter trio \( p \)—say the voter marked \( X \) has failed—then only one of the three modules in module trio \( r \) (the one marked \( X \)) can fail without causing system failure. Therefore, \( p \) is directly connected to \( r \). Similarly, \( q \) is directly connected to \( r \), and \( p \) is directly connected to \( q \). On the other hand, neither \( p \), \( q \), or \( r \) is directly connected to \( s \). We denote the relation "is directly connected to" by \( D \). Clearly, \( D \) is a symmetric relation. Further, we define that every trio is directly connected to itself, i.e., \( D \) is reflexive.

For any set of trios in a TMR network, two trios \( p \) and \( r \) are defined to be connected if there exists a sequence of trios in the set (possibly a null sequence) \( q_1, q_2, \ldots, q_n \) such that

\[ p \ D \ q_1 \ D \ q_2 \ \cdots \ D \ q_n \ D \ r. \]

Let \( C \) be the relation "is connected to." It is obvious that \( C \) is an equivalence relation.

Therefore, an arbitrary TMR network can be partitioned into equivalence classes using the relation \( C \). We call these equivalence classes cells. As an example, Fig. 4 shows a TMR network with the cells enclosed in dotted lines. The trios within a cell which feed trios in other cells or network outputs are known as cell output trios.

Any single error at an output trio of a cell will be corrected by the input (voter) trio of the next cell, while two or more errors will result in system failure. Therefore the cell reliability is defined as the probability of at most one error at each output trio of the cell. The network reliability is then the product of all the cell reliabilities.

C. Assumptions and Definitions Used in the Algorithm

To simplify the explanation of the algorithm, only net works with single output modules and voters following all the modules will be used. For the present, we will also assume that all the modules in a cell have the same reliability. The algorithm can be readily extended to include more complex cases, as will be discussed briefly later, and as described in [19].

The cell shown in Fig. 5(a) will be used as the example to illustrate the algorithm. Let \( N_e \) and \( N_m \) be the number of voter and module trios in a cell, respectively. In the example, \( N_e = 4 \) and \( N_m = 3 \).

The structure matrix, \( S \), of a cell is defined as follows. This matrix can be written down from inspection of the cell, and indicates which voter trios of the cell have paths to which module trios. Each of the voter and module trios is numbered arbitrarily, the voter trios from 1 to \( N_e \), and the module trios from 1 to \( N_m \). In Fig. 5(a), the voter trios are numbered from 1 to 4, and the module trios from 1 to 3. The structure matrix \( S \) is then defined to be an \( N_e \times N_m \) matrix such that

\[ S(i,j) = 1, \quad \text{if there is a path from voter trio } i \text{ to module trio } j \]

\[ = 0, \quad \text{otherwise.} \]

The structure matrix of the cell of the example is thus obtained in Fig. 5(b). For example, there is a path from voter trio 1 to module trio 1, but no path from voter trio 1 to module trios 2 and 3. Therefore, \( S(1,1) = 1 \) but \( S(1,2) = S(1,3) = 0 \). The other rows are obtained in a similar manner.

The fault matrix, \( F \), of a cell is defined as an \((N_e + 1) \times (N_m + 1)\) matrix, where \( F(i,j) \) is the number of exactly \( i \) voter faults and \( j \) module faults that the cell can have and yet remain reliable, i.e., produce at most one error at each output trio. If \( F \) can be obtained, then calculating the reliability of a cell is a simple matter, since \( F \) enumerates all possible fault patterns that the cell can tolerate.

Given a set with \( N \) elements, a combination of \( i \) elements is defined as one of the \( \binom{N}{i} \) subsets of \( N \) with \( i \) elements. A combination of trios can be further partitioned into equivalence classes generated by \( C \), and these are called groups.

For a combination of \( i \) voter trios in a cell, \( G_v \) is defined as the number of ways in which \( i \) voter failures (one from each trio) can occur without causing system failure. Suppose these \( i \) voter trios can be partitioned into \( n \) groups. Each voter in a group is connected to the other voters in the group, and so the voters in a group can fail in only three ways. Then, for this particular combination, \( G_v = 3^i \), since the groups are elements of a partition. From the cell of Fig. 5(a), consider the combination of three voter trios, \((1,3,4)\). There are two groups, \(((1),(3,4))\), and \( G_v = 3^3 = 9 \).

For a combination of voter trios, \( L \) is defined as an \( N_m \)
For the combination of voter trios (1,2) for example, \( L = 110 \), since there is no path from voter trios 1 and 2 to module trio 3.

For a combination of \( i \) voter and \( j \) module trios, \( G_m \) is defined as the number of ways in which \( j \) module failures (one from each module trio) can occur, given that \( i \) voter failures have occurred, without causing system failure. All the modules to which the \( i \) voter trios have paths can fail in only one way, while each of the module trios in the set of \( j \) module trios which are not connected to the \( i \) voter trios can fail in three ways. If the number of such module trios in the second set is \( m \), then, \( G_m = 3^m \). From the definition of \( L \) it can be seen that if we take the \( L \) corresponding to the combination of \( i \) voter trios, \( m \) is the number of zeros in the positions of \( L \) corresponding to the \( j \) module trios. For the voter trio combination (1,2) which has an \( L = 110 \) and a module trio combination (2,3) for example, the number of zeros in positions 2 and 3 of \( L \) is 1, thus \( m = 1 \).

C. Algorithm to Calculate the Reliability of a Cell

The algorithm to be described generates the fault matrix directly from the structure matrix of the cell. Table I gives the development of the algorithm for the cell of Fig. 5(a) and Table II is the fault matrix of the cell.

If no voters fail, the modules can fail independently, one module from each trio. The number of ways in which \( j \) modules can fail is then given by the number of ways of choosing \( j \) out of \( N_m \) trios, multiplied by the number of ways \( j \) modules can be chosen from the \( j \) trios, so that,

\[
F(0,j) = \binom{N_m}{j} 3^j, \quad j \geq 0.
\]

This gives the first row of the fault matrix in Table II.

Consider \( F(i,0) \), \( i > 0 \), which is the total number of ways in which \( i \) voters and 0 modules can fail. If we take any combination of \( i \) voter trios, the number of ways in which \( i \) voter failures can occur is given by \( G_i \). Therefore the total number of ways in which \( i \) voter failures and 0 module failures can occur is the sum of \( G_i \) over all the possible \((^N_i)\) combinations.

For each combination, the partition into groups can be made in many ways, but one way quite attractive for programming on a digital computer is the following. If two voter trios \( i_1, i_2 \) are directly connected, then the rows of the structure matrix corresponding to them (rows \( i_1, i_2 \)) will both have a 1 in the same position, and the \( A \) and of the two rows will not be the 0 vector. (A logical binary operation on two vectors is carried out by performing the binary operation on corresponding bits of the two vectors.) They then belong to the same group. The \( OR \) of the two
To find \( F(2,0) \) in the example we have to take the six possible combinations and sum the value of \( G \), for these, which gives 30. For the combination (3,4), the vector \( L \) is the OR of rows 3 and 4 of \( S \), and is equal to 011.

Now consider \( F(i,j) \), \( i,j > 0 \), which is the total number of ways in which \( i \) voters and \( j \) modules can fail without causing system failure. Given a combination of \( i \) voter trios, \( G_m \) is the number of ways in which \( j \) module failures can occur in a combination of \( j \) module trios. The number of ways in which \( i \) voters and \( j \) modules can fail for any given combination of \( i \) voter and \( j \) module trios is then \( G_r \cdot G_m \), and the total number of ways in which \( i \) voters and \( j \) modules can fail is the sum of \( G_r \cdot G_m \) over all such combinations of \( i \) voter and \( j \) module trios. Thus for every combination of voter trios, we take every possible combination of \( j = 1,2,\ldots,N_m \) bits of \( L \) and for each of these, \( G_m = 3^m \) where \( m \) is the number of zeros in that combination of bits of \( L \). This is shown in Table 1.

Taking the example again, consider the vector \( L \) of voter combination (1,2), which is 110. For module combination (1,3) the number of zeros in the positions 1 and 3 of \( L \) is 1, and \( G_m \) for this combination is \( 3^1 = 3 \), but for module combination (1,2) there are no zeros in those positions, and \( G_m \) for that combination is \( 3^0 = 1 \). \( F(1,3) \) is given by \( 3 \cdot 9 + 3 \cdot 3 + 3 \cdot 3 + 3 \cdot 3 = 54 \).

Thus the rest of the entries of the fault matrix are,

\[
F(i,j) = \sum_{\text{all combinations of } i \text{ rows of } S; \text{ all combinations of } j \text{ digits of } L} G_r \cdot G_m, \quad i > 0, j > 0.
\]

If all the modules of a cell have the same reliability, we do not need the separate entries of \( G_m \) but only the sum. In that case, if \( L \) has \( m \) zeros in it, \( \sum G_m \) for \( j \) module combinations is

\[
G_m = \sum_{j \text{ module trios}} j \binom{m}{j-k} \binom{N_m-m}{k} \cdot 3^{j-k}.
\]

It is assumed here that when \( k \) is negative or greater than \( n \), the binomial coefficient \( \binom{n}{k} \) is zero. The above expression is obtained by considering a particular combination of \( j \) digits of \( L \). If it has \( k \) 1's and \( j-k \) 0's in it, these can be arranged in \( \binom{m}{j-k} \cdot \binom{N_m-m}{k} \) ways, and for each of the arrangements, the value of \( G_m \) is \( 3^{j-k} \). We then sum over all possible values of \( k \).

The reliability of the cell is then given by,

\[
R_{\text{cell}} = \sum_{i=0}^{N_v} \sum_{j=0}^{N_m} F(i,j) \cdot R_v^{2N_v-i} \cdot (1-R_v)^i \cdot R_m^{2N_m-j} \cdot (1-R_m)^j
\]

where \( R_v \) and \( R_m \) are the reliabilities of a single voter and a single module, respectively.
TABLE I
Development of the Algorithm

<table>
<thead>
<tr>
<th>Voter combinations</th>
<th>$G_v$</th>
<th>$L$</th>
<th>Module combinations - $G_m$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 module</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(1)</td>
</tr>
<tr>
<td>1 voter</td>
<td>3</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>(2)</td>
<td>110</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>(3)</td>
<td>3</td>
<td>111</td>
<td>1</td>
</tr>
<tr>
<td>2 voters</td>
<td>(1,2)</td>
<td>3</td>
<td>110</td>
</tr>
<tr>
<td>(1,3)</td>
<td>9</td>
<td>111</td>
<td>1</td>
</tr>
<tr>
<td>(2,3)</td>
<td>3</td>
<td>111</td>
<td>1</td>
</tr>
<tr>
<td>(1,4)</td>
<td>3</td>
<td>111</td>
<td>1</td>
</tr>
<tr>
<td>(2,4)</td>
<td>3</td>
<td>111</td>
<td>1</td>
</tr>
<tr>
<td>(1,2,3)</td>
<td>3</td>
<td>111</td>
<td>1</td>
</tr>
<tr>
<td>3 voters</td>
<td>(1,2,4)</td>
<td>3</td>
<td>111</td>
</tr>
<tr>
<td>(1,3,4)</td>
<td>3</td>
<td>111</td>
<td>1</td>
</tr>
<tr>
<td>(2,3,4)</td>
<td>3</td>
<td>111</td>
<td>1</td>
</tr>
<tr>
<td>(1,2,3,4)</td>
<td>3</td>
<td>111</td>
<td>1</td>
</tr>
<tr>
<td>4 voters</td>
<td>(1,2,3,4)</td>
<td>3</td>
<td>111</td>
</tr>
</tbody>
</table>

TABLE II
Fault Matrix of the Cell in Fig. 5

<table>
<thead>
<tr>
<th>modules</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>9</td>
<td>27</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>66</td>
<td>108</td>
<td>54</td>
</tr>
<tr>
<td>voters</td>
<td>2</td>
<td>102</td>
<td>114</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>54</td>
<td>54</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>9</td>
<td>9</td>
<td>3</td>
</tr>
</tbody>
</table>

D. Approximation to the Algorithm to Reduce the Computational Complexity

The algorithm described in the previous pages provides a means of finding the exact reliability of a coherent TMR network. The algorithm does not take much storage space, since each combination of rows of $S$ is generated one at a time, and the $G_m$ and $G_v$ values found for that combination. There is no need to remember the combination from one row of the table to the next. What is sacrificed is execution time, since for $n$ voters and $m$ modules in a cell, on the order of $2^{n+m}$ operations is required, because we have to take all possible combinations of voters and modules. We are in effect trading time for accuracy. The entries in the fault matrix are the possible fault patterns for voters and modules which do not cause system failure. A method will now be described to obtain approximate values for some of the entries so that the total execution time is reduced; the reliability estimate is, nevertheless, very close to that which would have been obtained by using the exact method.

For an arbitrary cell, if we assume that every voter trio feeds every module trio, i.e., the $S$ matrix consists of all 1's, we get a lower bound on the entries of the $F$ matrix. This is because the assumption restricts the number of failure patterns. The number of ways in which voters and modules can fail increases if some voters or modules can fail independently of others. In the given case, the $S$ matrix consisting of all 1's), no voter or module can fail independently of another. Then $G_v$ for every combination of voter trios is 3, and $G_m$ for every combination of module trios is 1, and the entries of the fault matrix are given by,

$$F(i,j) = 3 \cdot \binom{N_i}{i} \cdot \binom{N_m}{j}, \quad i > 0.$$  

When we take combinations of $i$ voter trios, they represent cases where $i$ voter failures occur. If the voters are made of single gates (as in threshold voters), or are single integrated circuit chips, they will usually have a very high reliability. Therefore, for $i$ voter failures, the $(1 - R_v)^i$ term in the reliability equation becomes very small as $i$ becomes larger. Hence for large $i$, we are justified in using the lower bound given above.

One way to use the approximate method to save time without sacrificing too much accuracy is to use the exact method for $i = 0, 1, \cdots, i'$, and then use the approximate algorithm for $i = i' + 1, \cdots, N_v$. The choice of $i'$ is dictated by the time available, and the accuracy required; the accuracy depends on the voter and module reliabilities. If an accuracy and a time limit are specified, the reliability can be calculated as described above, and then, $i'$ can be increased by 1, and the reliability again calculated. If the difference in the two reliabilities is less than the accuracy required, we can stop. If not, and there is more time available, the iteration can be continued. If we run out of time, the accuracy to be expected can be returned by the program.

For $n$ voters and $m$ modules in a cell, the approximate method requires the order of $n \cdot m$ operations. Therefore, this method used in conjunction with the exact method can significantly reduce the time required for the reliability calculation [19].

E. Modifications of the Algorithm

Certain restrictions were placed on the type of cells, in the discussion above, in order to simplify the algorithm. We will indicate here briefly how they can be removed.

If the reliabilities of the different modules in a cell are different, the entries of the fault matrix must be split up.
in order to reflect the different failure modes of the different module trios. This information is readily available when the algorithm is developed. An example will show the procedure necessary. Suppose the three module trios in the cell of Fig. 5(a) have modules with reliabilities $R_m1$, $R_m2$, and $R_m3$ instead of the same $R_m$. Consider the combination of voter trios (1,2) for which $G_v = 3$ (from Table I) and the combination of module trios corresponding to these, (2,3) for which $G_m = 3$. Then the term in the reliability of the cell corresponding to these failures is

$$3 \cdot R_v^{10}(1 - R_v)^2R_m^3(1 - R_m2)R_m^3(1 - R_m3).$$

Thus we do not find $\sum G_v \cdot G_m$ but consider each $G_v \cdot G_m$ product as above. Therefore, with only a slight modification to the algorithm, the fact that different modules have different reliabilities can be taken into account.

If every module trio is not followed by a voter trio, more information must be maintained in the algorithm for now modules are directly connected to other modules. This problem can be solved by having additional rows in the structure matrix, corresponding to the modules in the cell. Then with a few other modifications to the algorithm, the entries of the $F$ matrix can be calculated. The algorithm with some refinements can also be extended to find the reliability of an NMR network [19].

IV. COMPARISON TO OTHER WORK

The reliability model developed in the previous section will now be compared to two approaches appearing in the literature: 1) the serial cell approach and 2) the minimal cut set approach. The basis of comparison will be a fan-out and a fan-in network.

Consider a 16 register multiplexed data bus system where the contents of a data register can be supplied to any one of 16 general purpose registers. Fig. 6 shows a TMR configuration of the data register to register transfer. The serial cell and minimal cut set models are developed in the Appendix.
The system reliability for the three approaches for the network of Fig. 6 is plotted as a function of module reliability in Fig. 7.

Now consider a case of 16:1 fan-in such as an arithmetic and logic unit (ALU) multiplexer which takes data from one of 16 registers as an input to an ALU. The three models for this fan-in network are also depicted in Fig. 7. The minimal cut lower bound is a rather poor predictor of system reliability while the serial cell approach predicts the same system reliability for both fan-in and fan-out networks.

Mission time improvement $I$ [18] is an interesting parameter for comparing redundant system designs. $I$ is the ratio of the times at which two redundant systems have the same reliability. For completeness, the concept of mission time improvement is developed in the Appendix. This parameter can also be used to compare reliability models. Fig. 8 shows a plot of mission time improvement
when \( I \) is the ratio of the exact model to the serial cell model. It can be seen that a mission time improvement of 50 percent for the 1-16 fan-out network can be obtained by using the more accurate reliability model. Another way of looking at the parameter \( I \) is that if the serial cell model is used then the resultant system is overdesigned by 50 percent since it could meet its mission time specification with less reliable components. Alternatively it could use the same component reliability and contain 50 percent more components and still meet the specifications. In the case of 16-1 fan-in the system only has 50 percent of designed mission time.

**CONCLUSIONS**

An algorithm has been described which finds the exact reliability of a TMR network, assuming that TMR is a coherent network. It thus gives a lower bound on the actual reliability of TMR [20]. Dividing the network into cells and considering each cell separately simplifies the reliability calculation significantly. An approximate method has also been described which takes much less time than the exact method and also gives a lower bound on the reliability.

**APPENDIX**

**Serial Cell Reliability Model**

The serial cell technique attempts to model all networks as a cascade of serial cells. Consider the serial cell of Fig. 2. Assume that the voters never fail. Under the coherent system assumption, there exists four states or module failure patterns for which the system still realizes its design function. They are 1) no module failures, and 2) three states, each of which has a single module failure (the two remaining working modules will realize the design function and form a majority regardless of the behavior of the failed module). Thus the cell reliability derived by summing over the working states is given by:

\[
R_{cell} = R_m^3 + 3R_m^2(1-R_m) = 3R_m^2 - 2R_m^3
\]  

(1)

where \( R_m \) is the module reliability. If a wrong input to a module is assumed to yield a wrong output then a voter failure has the same effect as a module failure. Replacing \( R_m \) by \( R_mR_v \), in (1) yields:

\[
R_{cell} = 3(R_mR_v)^2 - 2(R_mR_v)^3
\]  

(2)

where \( R_v \) is the voter reliability.

Now consider network configurations more complex than a cascade of serial cells, such as a network exhibiting fan-out such as the network of Fig. 6. One approach to handle fan-in/out in the serial cell reliability model is to assign the voters to the modules they drive [17] since a voter failure affects only the module it drives. Thus cell 2 of Fig. 6 shows one way to assign voters to the driven modules. Now the serial cell reliability model for the network of Fig. 6 can be developed.

The reliability of a module end cell such as cell 1 can be derived from (2) by letting \( R_v = 1 \). Similarly setting \( R_m = 1 \) in (2) yields the reliability of voter end cells such as cell 3. Next assume \( R_v = R_v = R \), i.e., \( R_m = R_v \). This simplification is not crucial and similar results are obtainable when \( R_v \) and \( R_m \) retain their separate identities. The end cell reliability is thus \( 3R^2 - 2R^3 \). The serial cell reliability model for the system of Fig. 3 would consist of 17 end cells (16 voter and one module) and 16 serial cells, like cell 2, each of which share the one voter trio. The system reliability is thus modeled by:

\[
R_{serial} = (3R^2 - 2R^3)^{17}(3R^4 - 2R^6)^{16}. \tag{3}
\]

For the case of fan-in there are still 17 end cells (16 module and one voter). The fan-in portion would consist of 16 overlapping serial cells. Thus (3) represents the serial cell model for both fan-in and fan-out.

**Minimal Cut Set Reliability Model**

The lower bound on system reliability is given by [16]:

\[
\prod_{\forall c \in \text{cut}} (1 - \alpha_i)
\]

such that

\[
\text{if i is a minimal cut}
\]

where \( \alpha_i \) is the probability that the minimal cut does not occur (i.e., all the components composing the minimal cut do not fail). Consider Fig. 6. All minimal cuts consist of either two voters, two modules, or one voter and one module. There are 3 ways two modules can fail in the module end cell and 16-3 ways two voters can cause system failure in the voter end cells. In the fan-out portion there are 3 double voter failures, 3-16 double module failures, and 3-2-16 single voter single module failures (such as voter \( A \) and module \( B \)) whose failure would cause system failure. Hence the minimal cut reliability model for fan-out is:

\[
(1 - (1 - R)^3)^{18}. \tag{4}
\]

Now consider the case of fan-in. There are 16-3 ways two modules can cause system failure in the 16 module end cells and 3 ways for two voters in the voter end cell. In the fan-in portion there are 3 double module failures, 3-16-2 single voter and single module failures, 3-16 double voter failures in the same voter trio, and 3-2- \( \sum_{i=1}^{20} i \) or 720 ways two voter failures from different voter trios can interact to cause system failure. Thus the minimal cut reliability model for fan-in is:

\[
(1 - (1 - R)^3)^{918}. \tag{5}
\]

**Mission Time Improvement**

Mission time improvement can be used to compare two redundant systems or two reliability models. First the two reliability models are derived. Let \( R_m = \exp(-\lambda m) \) and \( R_m' = \exp(-\lambda m') \). A value for \( R_m' \) is assumed and sub-
stutized in one equation. Then an $R_{m}'$ is calculated such that the two system reliabilities are identical. If we represent $R_m$ by $R_m = R_{m}'$ then $\lambda_1 t_1 = \lambda_2 t_2$. Further if $\lambda_1 = \lambda_2$ then $t_1 = t_2$ and model one has the same system reliability at $t_1$ as model two does at time $t_2$. The mission time improvement is defined as $I$.

ACKNOWLEDGMENT

The authors wish to thank Prof. E. J. McCluskey of the Digital Systems Laboratory, Stanford University, Stanford, Calif., for his encouragement and many stimulating discussions.

REFERENCES


Jacob A. Abraham (S’71) was born in Kerala, India, on December 8, 1948. He received his B.Sc.(engineering) degree in electrical engineering from the University of Kerala, Kerala, India, in 1970 and the M.S. degree in electrical engineering from Stanford University, Stanford, Calif., in 1971. He is currently working towards the Ph.D. degree in electrical engineering at Stanford University.

Since 1971 he has held a Research Assistantship in the Digital Systems Laboratory, Stanford University, working on fault-tolerant computing. He is interested in reliability modeling of redundant systems, fault-tolerant computing, and computer architecture.

Mr. Abraham is a member of the IEEE Computer Society, the Association for Computing Machinery, and Sigma Xi.

Daniel P. Siewiorek (S’67–M’72) was born in Cleveland, Ohio, on June 2, 1946. He received the B.S. degree in electrical engineering (summa cum laude) from the University of Michigan, Ann Arbor, in 1968, and the M.S. and Ph.D. degrees from Stanford University, Stanford, Calif., in 1969 and 1972, respectively.

While completing his doctoral studies at Stanford University, he held a Research Assistantship in the Digital Systems Laboratory, working on fault-tolerant computing. Since the Spring of 1972 he has been an Assistant Professor of Electrical Engineering and Computer Science, Carnegie-Mellon University, Pittsburgh, Pa., where he has been teaching courses in switching theory, logic design, and computer organization. His current research interests include computer architecture, reliability modeling, fault-tolerant computing, modular design, and design automation. Dr. Siewiorek is serving as Co-Chairman of the Pittsburgh Chapter of the IEEE Computer Society and as Associate Editor of the Computer Systems Department of the Communications of the Association for Computing Machinery.

He is a member of Tau Beta Pi, Eta Kappa Nu, and Sigma Xi.