can be rewritten as
\[ \Phi^* = d_1(d_2 + d_3 + d_4) + d_5(ad + b + cde + ace). \]

Noting that the desired function is to be realized on node 1 of the circuit, the CF of the specification can be obtained [2], [3] as

\[ \Phi_1 = d_1 + d_2 + d_3 + d_4 + \text{DON'T CARE TERMS} \]

\[ - d_5(ad + b + cde + ace) \]

\[ + d_5(ad + b + cde + ace + bcd) \]

\[ + abcde. \]

From [2] and [3], a circuit realizes its specification if \( \Phi^* \leq \Phi_1 \). In other words, \( \Phi^* - \Phi_1 = 0 \). That the latter relation is satisfied can be easily verified by inspection, since \( \Phi_1 \) can be expressed as

\[ \Phi_1 = d_1 + d_2 + d_3 + d_4 + d_5(ad + b + cde + ace + bcd) \]

\[ - (ab + cde). \]

VIII. CONCLUSIONS

We have shown how multiple valued characteristic functions can be used to extract functional descriptions of CSA networks from their structure and external signal constraints.

The algebraic manipulations can be programmed on a computer, for instance, by representing the CF's using cubical complexes [2], [15] extended for multiple values of the constituent variables. The technique described here can be used for formal verification, since it transforms a network into a series of logic functions. The analysis can be aided by applying the properties of Boolean equations [2]-[5], [16]. Our current work is in this direction [13], [14].

REFERENCES


failure, been describing, the time until a memory system failure occurs will

tolerance. Eventually, however, we expect that so many chip failures

will have occurred that some one of the individual

memory chips) guarantees that no chip failure, however catastrophic,

will have suffered two or more errors. When this happens, we declare

function

measures of this random variable, the

a

types are illustrated in Fig. 3. (The letters A, B,

column

cell on a chip. Such errors are caused by stray alpha particles which

can cause two errors in any

memory chips) guarantees that no chip failure, however catastrophic,

will have occurred in the ith row of chips at time

The organization of memory chips.

Fig. 1. The organization of memory chips.

situation in which one or more of the bits written on a chip cannot be

reliably read. These failures are traditionally classified as either

hard, meaning that the affected memory cells are permanently

damaged, or soft, meaning that the damage is only temporary.

Laboratory observation of real memories [1], [12], [13] shows that by far the most common type of chip failure is a soft error of a single
cell on a chip. Such errors are caused by stray alpha particles which

can, under certain circumstances, change a stored "1" to a "0" without otherwise damaging the chip [14]. However, several kinds of

hard failure have been observed. A single-cell failure, for example, can also occur as a hard error. There are also several kinds of hard

failures which cause multiple cell errors. A row failure, which can be caused by a failure of one of the chip’s row drivers, causes all

cells in one row of the affected chip to fail. Similarly, a column failure, which can be caused by a failure of one of the chip’s column

sense amplifiers or column decoders, causes all l rows of memory.

A short circuit at a memory cell can cause a row-
column failure. Finally, a catastrophic whole chip failure may

occur, in which all the cells of a chip fail. All five of these failure

types are illustrated in Fig. 3. (The letters A, B, C, D, and F will be

referred to in Section II.)

The organization of the SEC-DED code (assuming "by-one"

memory chips) guarantees that no chip failure, however catastrophic,
can cause two errors in any n bit codeword, and so the memory

system will survive any single-chip failure. In fact there are many

possible combinations of multiple chip failures that can also be

tolerated. Eventually, however, we expect that so many chip failures

will have occurred that some one of the individual n bit codewords

will have suffered two or more errors. When this happens, we declare

a memory system failure.

If we start with a brand new memory system of the kind we have been
describing, the time until a memory system failure occurs will be

a random variable. In this paper we will derive accurate and easily

evaluated estimates for two of the most important quantitative measures of this random variable, the system reliability function

and the mean time to failure (MTTF). The reliability function

represents the probability that the system will not have failed after

hours, and the MTTF represents the average length of

time the system will function before a memory system failure occurs.

In the next section (Section II) we will describe the probabilistic model which is commonly used to describe the occurrences of the

various types of chip failures, and see how it leads, via a Poisson

approximation, to a reasonably simple formula for

and

MTTF. In Section III we will give some useful approximations to the

exact formula found in Section II. In Section IV we will make

numerical comparisons of the predictions of our formulas to the

results of computer simulation. The analytic predictions will be seen
to be in very close agreement with the simulations, thereby justifying

our confidence in the accuracy of the Poisson approximation made in

Section II. Finally, in Section V, we will compare our results to those which have already appeared in the literature.

II. MODELS: FORMULAS FOR R(t) AND MTTF

Our basic quantitative assumption about individual chip failures is that they are exponentially distributed. This means that the

reliability of a given chip, i.e., the probability that it has not failed

after

hours is equal to

where

is a constant that must be found experimentally [11]. We will need to distinguish between the five types of chip errors depicted in Fig. 3, and so for future reference, we will use the following notation.

A: row failure

B: column failure

C: single-cell failure

D: row-column failure

F: whole chip failure.

We assume that, if a given chip fails, the conditional probabilities that the failure will be of type A, B, C, D, or F are

respectively. The probabilities

and

also have to be determined experimentally. We further assume that failures on one chip are independent from failures on all other chips.

Given all these assumptions, it is in principle possible to calculate the

row reliability function

defined as follows.

for

not yet occurred in the ith row of chips at time

For example, if the only kind of chip failures were whole chip failures, we would have

and

which is just the probability that the given row has suffered either

zero or one whole chip failure after

hours [10]. Since the

rows are assumed to fail independently, the reliability of the entire system of

chips is

The MTTF of system whose reliability function is

is well known to be given by the formula

and so for a computer memory system of the kind we are considering,

Thus, everything we are interested in depends in a simple manner on the

row-reliability function

Unfortunately, however, an exact formula for

proves to be extremely complicated. (For example, Mikhail et al. [15] give a recursive method for computing it when errors of types

and

are present.) Thus, difficulty has led us to make the following simplifying assumption. We no longer view a row of chips as consisting of

separate chips, but "end on" so that the failures on all

chips are superimposed onto a single "protochip." We also make an

important assumption about how protochip failures are distributed,
In each protochip, the failures of types A, B, C, D, and F form independent Poisson processes of intensities \( an \lambda, bn \lambda, cn \lambda, dn \lambda, \) and \( fn \lambda, \) respectively.

Under this assumption, the row reliability function \( R(t) \) is just the probability that at time \( t \) no cell on the protochip has suffered two or more errors. As we will see, this assumption greatly simplifies the formulas for \( R(t) \) without introducing significant inaccuracies. For example, if we again consider a situation in which only whole chip failures occur, then under the Poisson assumption the number of whole chip failures in a given row is a Poisson process of intensity \( \lambda n, \) and so the row-reliability function, i.e., the probability of zero or one whole chip failures after \( n \) hours is given by

\[
R(t) = e^{-\lambda n t}(1 + \lambda nt).
\]

Formulas (6) and (2) are very similar, so if we use the Poisson assumption via (6) and (4) we get instead

\[
\text{MTTF} = \frac{1}{\lambda} \left( \frac{1}{n} + \frac{1}{n-1} \right),
\]

whereas, if we use the Poisson assumption via (6) and (4) we get instead

\[
\text{MTTF} = \frac{1}{\lambda} \left( \frac{2}{n} \right).
\]

In Section IV we will give further comparisons between exact MTTF's and those obtained by the Poisson protochip method.

Using the Poisson protochip, we can now derive a formula for the row-reliability function \( R(t) \), the probability that the memory system is still working after \( t \) hours. It will be convenient to classify the various tolerable combinations of protochip failures into the following four categories.

- \( E_i \): only row and single-cell failures
- \( E_2 \): only column and single-cell failures
- \( E_3 \): one row-column failure and single-cell failures
- \( E_4 \): one whole chip failure

These 'tolerable failure configurations' are shown in Fig. 4.

We note that the configurations \( E_1 \) and \( E_2 \) are not disjoint, so that we also introduce \( E_{12} \), defined as

\[
E_{12} = E_1 \cap E_2: \text{only single-cell failures}.
\]

Then the row-reliability function defined by (1) is

\[
R(t) = \Pr \{ E_1 \cup E_2 \cup E_3 \cup E_4 \} = \Pr \{ E_1 \} + \Pr \{ E_2 \} - \Pr \{ E_{12} \} + \Pr \{ E_3 \} + \Pr \{ E_4 \}.
\]

We now proceed to calculate the five probabilities in (7).

We begin by calculating \( \Pr \{ E_1 \} \), which is the probability that the protochip has suffered no errors of type \( B, D, \) or \( F \), and that the errors of type \( A \) and \( C \) are 'tolerable,' i.e., that no cell on the protochip has suffered two or more errors. To do this, we focus on a single row of cells on the protochip, say row \( i \). Since each protochip has \( l \) rows, our Poisson assumption implies that the row failures in this particular row form a Poisson process of intensity \( an \lambda / l \). Therefore, the probability that there have been no row failures in row \( i \) at time \( t \) is

\[
\exp \left( -\frac{an \lambda t}{l} \right),
\]

and the probability that there has been exactly one row failure in this row is

\[
\exp \left( -\frac{an \lambda t}{l} \right) \left( \frac{an \lambda t}{l} \right).
\]

To simplify the notation, from now on we will use the parameter \( x \) defined by

\[
x = \lambda nt,
\]

so that the probability of zero and one failures in row \( i \) at time \( t \) are given by

\[
e^{-ax/2} + e^{-ax/2}(cx/l),
\]

respectively. Next we focus on a particular cell within the \( i \)th row, say cell \( (i, j) \). Since each chip contains \( l \) cells, our Poisson assumption implies that the single-cell failures in this particular cell form a Poisson process with intensity \( cnl/l^2 \). Therefore, the probabilities that the \( (i, j) \)th cell has suffered zero or one single-cell errors at time \( t \) are

\[
e^{-cx/2} + e^{-cx/2}(cx/l^2),
\]

respectively. It follows that the probability that at time \( t \) no cell in row \( i \) has suffered a single-cell error is

\[
(e^{-cx/2})^I = e^{-cx/I},
\]

while the probability that no cell in row \( i \) has suffered more than one single-cell error is

\[
\left( e^{-cx/2} \left( 1 + \frac{cx}{l^2} \right) l \right)^I.
\]

We now wish to calculate the probability that no cell within the \( i \)th row has suffered two or more errors of type \( A \) or \( C \). This probability is the sum of the following two probabilities:

- The row has not failed and there is at most one single-cell error in each of the \( l \) cells.
- The row has failed and there are no single-cell errors in any of the \( l \) cells.

This probability is, therefore,

\[
e^{-ax/2} \left( 1 + \frac{cx}{l^2} l \right)^I + e^{-ax/2}(cx/l)(e^{-cx/I}).
\]

Finally, the probability \( \Pr \{ E_{12} \} \) is the probability that the type \( A \) and \( C \) errors in all \( l \) rows of the protochip will be tolerable, which is just the \( l \)th power of (9), multiplied by the probability that there are no errors of type \( B, D, \) or \( F \), which is \( \exp \left( -(b + d + f)x \right). \) After some algebra we find that this product is

\[
\Pr \{ E_{12} \} = e^{-x} \left( 1 + \frac{cx}{l^2} l \right)^I \frac{ax}{l}. \]

By replacing "rows" with "columns" in the preceding argument, we find that

\[
\Pr \{ E_2 \} = e^{-x} \left( 1 + \frac{bx}{l^2} l \right)^I \frac{ax}{l}. \]

To compute \( \Pr \{ E_{12} \} \), we note that this case corresponds to no errors of types \( A, B, D, \) or \( F \), and at most one error in each cell of the
Thus, there can be no further failures of any kind. Therefore, devoted to exploring its consequences.

To calculate \( \text{Pr} \{ E_2 \} \), note that if there is a row-column failure and if there are no failures of types \( A, B, \) or \( F \), we can tolerate zero or single-cell failures in the unaffected \((l-1)^2 \) cells but no errors in the \( 2l-1 \) cells affected by the row-column failure. Hence,

\[
\text{Pr} \{ E_2 \} = e^{-a+b+c} \cdot e^{-\alpha x} (dx) \cdot (e^{-\alpha x})^{2l-1} \\
\cdot \left( e^{-\alpha x/2} + \frac{a x}{2} \right)^{2l-1} \\
\cdot \left( 1 + \frac{a x}{2} \right)^{2l-1} \\
e^{-\alpha x} \left( 1 + \frac{a x}{2} \right)^{2l-1} \\
= e^{-\alpha x} \left( 1 + \frac{a x}{2} \right)^{2l-1}.
\]

(13)

To calculate \( \text{Pr} \{ E_3 \} \), we note that if there is a whole chip failure, there can be no further failures of any kind. Therefore,

\[
\text{Pr} \{ E_3 \} = e^{-a+b+c} \cdot e^{-\alpha x} (dx) \\
= e^{-\alpha x}.
\]

(14)

Finally, to compute the row-reliability function \( R(t) \), we combine (7) with (10), (11), (12), (13), and (14), and obtain the following somewhat intimidating expression.

\[
R(t) = e^{-a} \left[ e^{-a/2} (1 + a/2)^{2l-1} \\
+ e^{-\alpha x} (1 + a x/2)^{2l-1} + dx \left( 1 + a x/2 \right)^{2l-1} \right].
\]

(15)

This expression is our main result. The rest of the paper will be devoted to exploring its consequences.

III. ASYMPTOTIC APPROXIMATIONS AND SPECIAL CASES

Although the expression (15) for \( R(t) \) is simple enough for numerical work, it is possible to give approximations to it that will yield additional insight into the problem. For example, in most modern chips the number \( I \) (the number of storage cells per row or column) is quite large, and this suggests that the limiting behavior of (15) as \( l \to \infty \) may be interesting to consider. In fact, this limit is given by the formula

\[
R_\infty (t) = e^{-a} \left[ e^{-a/2} (1 + a/2)^{2l-1} \\
+ e^{-\alpha x} (1 + a x/2)^{2l-1} + dx \left( 1 + a x/2 \right)^{2l-1} \right].
\]

(16)

This formula is simple enough to integrate explicitly, and so by (5) we find that the MTTF for one row of chips is approximately

\[
\int_0^\infty R_\infty (t) \, dt = \frac{1}{\lambda M} \left( \frac{1}{1 - a - c} + \frac{1}{1 - b - c} - \frac{1}{1 - c} \right) + \frac{d}{(1 - c)^2}.
\]

(17)

Experimentation with (16) and (17) indicates that these approximations are quite accurate if \( a + c \) is not too close to 1. Such a restriction is understandable, since if, e.g., \( c = 1 \), an \( l = \infty \) chip would have an infinite MTTF, since the probability of any given cell position being hit twice would be zero.

Another interesting case to consider is the case of a large number of rows. (A CRAY-1, for example, has \( M = 1024 \) rows, each consisting of 72 1K ECL RAM chips.) In this case we can exploit the classic theory of asymptotic analysis [3] and find an asymptotic formula for the integral in (5). Omitting the details, we find that the result is of the form

\[
\text{MTTF} = \frac{1}{\lambda M} \sqrt{\frac{\pi}{M \cdot K_1 + K_2}},
\]

(18)

In (18) the number \( 1/\lambda M \) represents the mean time between chip failures (there are \( n M \) chips and each one has an MTTF of \( 1/\lambda \)). The other term, viz., \( \sqrt{M \cdot K_1 + K_2} \) represents the asymptotic value of the mean number of events to failure (MTTF), which is the average number of chip failures which occur before a system failure occurs. The constants \( K_1 \) and \( K_2 \) in (18) are determined as follows. If we call the bracketed term in (15) \( r(x) \) and expand it as a polynomial in \( x \) up to terms of degree 3, we find that

\[
r(x) = 1 + x + r_2 x^2 + r_3 x^3 + \cdots,
\]

(19)

where

\[
r_2 = \frac{c^2}{2} \cdot \frac{(l^2 - 1)}{l^2} \cdot \frac{(l - 1)}{l} \cdot \frac{(a^2 + b^2)}{2} \\
+ \frac{(a^2 + b^2)(l^2 - 3l + 2)}{6} \cdot \frac{l^2}{l^2} \cdot \frac{l^2}{l^2}.
\]

The constants \( K_1 \) and \( K_2 \) in (18) are then given by

\[
K_1 = \frac{\pi}{2(1 - 2r_2)}
\]

(20a)

\[
K_2 = \frac{2(r_2 - r_3)^2}{3(1 - 2r_2)}.
\]

(20b)

The difference between the asymptotic formula (18) and the true value (5) is guaranteed to go to zero, if \( a, b, c, d, f, \) and \( l \) are fixed and \( M \to \infty \). However, (18) usually gives remarkably accurate answers for small values of \( M \), often even for \( M = 1 \), as we shall see in the next section.

We now consider two special cases, viz., \( f = 1 \) and \( c = 1 \). When \( f = 1 \) (all failures are whole chip failures), the reliability function \( R(t) \) in (15) becomes simply

\[
R(t) = e^{-a} (1 + x),
\]

which is exactly the same as the \( l = \infty \) approximation (16). Therefore, the MTTF for one row of chips is given exactly by (17), i.e.,

\[
\text{MTTF} = \frac{2}{\lambda M}.
\]

Since there are \( n \) chips, each with MTTF equal to \( 1/\lambda \), this formula simply reflects the fact that the system will fail as soon as two whole chip failures have occurred. (We already observed this in Section II.) More interesting is what happens when the number of rows \( M \) is large. In this case the system will fail as soon as one of the \( M \) chip rows has suffered two errors. If we interpret the occurrence of a chip
failure in the \(i\)th row as the arrival of a "person" whose "birthday" occurs on the \(i\)th day of an \(M\) day year, we see that the expected number of chip failures needed to cause a system failure is the same as the expected number of people we need to interview before we find two with the same birthday. It is known that this "birthday surprise" number is given by [8]

\[
B(M) = M \cdot \int_0^\infty (e^{-x} (1 + x))^M dx.
\]

(21)

This is in agreement with the theory developed in Section II, since with \(f = 1\) the formula (15) simplifies to \(R(t) = e^{-x(1 + x)}\), and so by (5) the MTTF is

\[
\text{MTTF} = \frac{1}{\lambda M} \cdot \int_0^\infty (e^{-x} (1 + x))^M dx
\]

\[
= \frac{1}{\lambda n M} \cdot B(M).
\]

Furthermore, since \(r(x) = 1 + x\) in this case, in the expansion (19) we have \(r_2 = r_3 = 0\), and so the asymptotic formula (18) becomes

\[
\text{MTTF} = \frac{1}{\lambda n M} \left( \sqrt{\frac{\pi M}{2}} + \frac{2}{3} \right).
\]

(22)

What this means is that the term \(\sqrt{\pi M/2} + 2/3\) is an approximation to the METF, which in this case is simply the mean number of whole chip failures before system failure, as well as the "birthday surprise" number \(B(M)\):

\[
B(M) = \text{METF} - \sqrt{\frac{\pi M}{2} + \frac{2}{3}}.
\]

(23)

The approximation given in (23) is very accurate, too, as the following table shows.

<table>
<thead>
<tr>
<th>(M)</th>
<th>Exact (B(M)) [from (21)]</th>
<th>Approx. (B(M)) [from (23)]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.000</td>
<td>1.920</td>
</tr>
<tr>
<td>2</td>
<td>2.500</td>
<td>2.439</td>
</tr>
<tr>
<td>4</td>
<td>3.219</td>
<td>3.173</td>
</tr>
<tr>
<td>8</td>
<td>4.245</td>
<td>4.212</td>
</tr>
<tr>
<td>16</td>
<td>5.704</td>
<td>5.680</td>
</tr>
<tr>
<td>32</td>
<td>7.774</td>
<td>7.756</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>365</td>
<td>24.616</td>
<td>24.611</td>
</tr>
</tbody>
</table>

The \(M = 365\) entry shows that on a planet with a 365-day year, one needs to interview between 24 and 25 people, on the average, before finding two with the same birthday. Alternatively, in a 365-row computer memory in which only whole chip failures occur, and in which each row is SEC-DED protected, the memory will tolerate between 24 and 25 chip failures, on the average, before failing.

A similar "birthday analysis" can be made for the case \(c = 1\). In this case only single-cell failures occur; it is as if there were \(fM\) independent rows of \(1 \times 1\) chips. The number of single-cell failures which can be tolerated before a system failure occurs is thus \(1/\lambda n M\) times the "birthday number" \(B(12M)\). This too is consistent with our theory, since with \(c = 1\), the formula (15) becomes

\[
R(t) = e^{-x} \left( \frac{1 + x}{f^2} \right)^{1/2},
\]

and so by (5)

\[
\text{MTTF} = \frac{1}{\lambda n} \cdot f^2 \cdot \int_0^\infty (e^{-x} (1 + x))^{12M} dx
\]

\[
= \frac{1}{\lambda n M} \cdot B(12M).
\]

The asymptotic formula (18) is even more accurate in this case, since with \(r(x) = 1 + x/f^2\) we have

\[
r_2 = \frac{c^2}{2} \cdot \frac{(f^2 - 1)}{f^2}
\]

\[
r_3 = \frac{c^2}{6} \cdot \frac{(f^4 - 3f^2 + 2)}{f^4}
\]

which means that the asymptotic formula (18) works out to be

\[
\text{MTTF} = \frac{1}{\lambda n M} \left( \sqrt{\frac{\pi M}{2}} + \frac{2}{3} \right).
\]

(24)

exactly the same as (22) except that \(M\) is replaced with \(f^2 M\). In this case it is typically quite difficult to integrate \(R(t)^M\) accurately, and the approximation (24) provides the only reliable way of obtaining accurate values for the MTTF.

The relationship between MTTF's and "birthday surprises" was originally noted in [6] and [14].

IV. NUMERICAL EVALUATION OF THE MTTF FORMULAS

In this section we will illustrate our results numerically. The plan is to take three sets of values for the parameters \(a, b, c, d, f, l\), as reported in the literature, and for various values of \(M\) to compute the MTTF (which differs from the METF by the factor \(1/\lambda M n\), as we explained in the last section) using four methods. The first method is direct Monte Carlo simulation. This method is very slow, but it makes no use of our Poisson assumption and provides a valuable check of the accuracy of our other methods. The second method is direct integration of the expression (15):

\[
\text{METF} = M \cdot \int_0^\infty (R(x))^M dx.
\]

The third method is direct integration of the \(l = \infty\) approximation of the row-reliability function, given in (16):

\[
\text{METF} = M \cdot \int_0^\infty (R(x))^M dx.
\]

The final method is to use the two-term asymptotic approximation given by (18), viz.,

\[
\text{METF} = \sqrt{MK_1} + K_2.
\]

The sets of parameters are taken from [9], [11], and [15], as summarized in the following table.

<table>
<thead>
<tr>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
<th>(d)</th>
<th>(f)</th>
<th>(l)</th>
</tr>
</thead>
<tbody>
<tr>
<td>[9]</td>
<td>0.01646</td>
<td>0.01646</td>
<td>0.85234</td>
<td>0</td>
<td>0.11365</td>
</tr>
<tr>
<td>[11]</td>
<td>0.047</td>
<td>0.047</td>
<td>0.893</td>
<td>0.013</td>
<td>0</td>
</tr>
<tr>
<td>[15]</td>
<td>0.12</td>
<td>0.18</td>
<td>0.35</td>
<td>0</td>
<td>0.35</td>
</tr>
</tbody>
</table>

Our numerical results for the corresponding METF's are given in Figs. 5-7. Each of the numbers in the "simulation" columns represents the average number of (simulated) chip failures before (simulated) system failure for 40000 Monte Carlo trials, reported to the nearest tenth.

We see in very case that our "exact" expression (15) gives results that are indistinguishable from the simulation results. The \(l = \infty\)...
failure and F simultaneously. For example, pairs of row errors. The asymptotic estimate is always low, positive terms which we have neglected. This is because with I

earlier introduced. Indeed, our model is to our knowledge the only one that handles all five failure types A, B, C, D, and F simultaneously. We believe that the key innovation of our paper, however, is the introduction of the Poisson approximation. As we have seen, this approximation allows us to obtain simple formulas for the system reliability without sacrificing significant accuracy. And although our main formula (15) may seem excessively complex, when compared to the corresponding formulas in [9], [11], and [15], it is very simple indeed. As we have demonstrated in Section IV, it can be easily programmed to give fast and accurate reliability estimates that can be used by memory system designers.

REFERENCES
[4] C. L. Chen and Y. H. Hsiao, "Error-correcting codes for semicon-
[7] M. Y. Hsiao, "A class of optimal minimum odd-weight-column SEC-

V. A SURVEY OF RELATED WORK: CONCLUSION
The literature contains many papers devoted in whole or part to the subject of this paper, including two survey papers [9] and [15]). In this section we will attempt to describe how our work adds to what is already known.

The earliest work on ECC memory reliability [10] deals only with type F chip failures, i.e., whole chip failures. Later models, including those in [9] and [11], extended the types of failure modes to include types A, B, C, and D, but as pointed out in [16], it is impliedly assumed in these models that the failure types are "nested." That is, there is a hierarchy of failure types, such that each type is a subset of the previous type. For example, single-cell, row, and whole chip failures are nested, but no nested hierarchy can contain both row and column failures. Since one row and one column failure in a row of chips will cause a memory system failure, it is important to have a model that handles "crossed" failure types, e.g., failure types A and B simultaneously, as is done in [15]. However, [15] does not consider failute type D. Indeed, our model is to our knowledge the only one that handles all five failure types A, B, C, D, and F simultaneously.

FISHNET: A Distributed Architecture for High-Performance Local Computer Networks
YONG J. KANG, JAMES H. HERZOG, AND JOHN SPRAGINS

Abstract—FISHNET (Facilities Integrated in a Shared Habitat Network) addresses the problem of effectively integrating a wide variety of computers, terminals, memory devices, and computer peripherals in a local environment. High performance is achieved by effectively separating the high volume node-to-node data communications and the low-