Lifetime analyses of error-control coded semiconductor RAM systems

R.M.F. Goodman, B.Sc., Ph.D., C.Eng., and Prof. R.J. McEliece, B.Sc., Ph.D.

Indexing terms: Codes, Computer applications, Memory systems

Abstract: The paper is concerned with developing quantitative results on the lifetime of coded random-access semiconductor memory systems. Although individual RAM chips are highly reliable, when large numbers of chips are combined to form a large memory system, the reliability may not be sufficiently high for the given application. In this case, error-correction coding is used to improve the reliability and hence the lifetime of the system. Formulas are developed which will enable the system designer to calculate the improvement in lifetime (over an uncoded system) for any particular coding scheme and size of memory. This will enable the designer to see if a particular memory system gives the required reliability, in terms of hours of lifetime, for the particular application. In addition, the designer will be able to calculate the percentage of identical systems that will, on average, last a given length of time.

List of principal symbols

λ = chip failure rate
R = chip reliability, the probability of correct operation of a chip
Q = probability of chip failure
R_R = row reliability
R_S = system reliability
α = probability of system operating correctly, equal to R_S
n = number of chips in a coded row
k = number of data-carrying chips in a row
r = error correction power of code
m = number of chip rows in memory system
$T_r(\alpha)$ = system lifetime to probability level α, with r bit error correction
$T_r(1/2)$ = median time to failure (mTTF)
$\mu_r(\gamma)$ = a solution of the Poisson distribution
$C_r(\alpha)$ = coding gain

1 Introduction

The continued decrease in cost of semiconductor random access memory (RAM) chips makes the construction of very large memory arrays an economic possibility. Not only can such arrays be used to form the basic core memory of large computers, but also small size microcomputers can benefit from large memory arrays for applications such as speech processing, picture processing, intelligent terminals and data bases etc. If large memory systems such as these are to be increasingly used, it is essential that they should be reliable, and not require frequent servicing. Although an individual RAM chip may have a quoted reliability of better than 10^{-6} failures/h, when large numbers of these chips are combined to form a total system the reliability of the system becomes exponentially worse. In these cases, it is essential that some form of error correction coding (ECC) be used to protect against data loss.

LSI dynamic RAMs generally have a low failure rate of the order of 10^{-7} to 10^{-6} failures/h. In addition, there is a 'learning curve' for devices as they appear on the market, which means that the larger RAMs have a lower reliability than the smaller devices. For example, the current industry standard 16k-by-1 bit dynamic RAM has a quoted failure rate of approximately 3×10^{-7} failures/h [1], whereas the 64k RAMs just appearing will initially have a failure rate much worse than this. Thus, although system reliability is increased by using larger RAMs, the need for ECC remains.

The need for improved system reliability is of particular concern if a large memory system is to be mass produced. For example, if a memory system has a 10% probability of failure after one year, we may turn the argument 'around' and say that on average 10% of the manufacturer's memory systems are only going to last one year. This would clearly be unacceptable in many applications.

In this paper, we assess the improvement in memory-system lifetime that can be obtained by using error-control coding. We are not concerned here with the particular form of coding used, as these are dealt with in the literature [2–6]. We derive general results that will enable a system designer to get an accurate impression of what coding will do for any particular memory system. Firstly, we consider chip failures and the need for ECC; next, system lifetime is defined and a general formula for the improvement in lifetime of a coded system is defined. This is then developed into a formula for the median time to failure (mTTF). We then consider asymptotic results for the case of large memory systems. Finally, we develop a formula for calculating the time at which any given probability of system failure exists.

2 Chip failures and need for ECC

The predominant failure mode within a RAM chip is a 'stuck-at' fault. In this mode, either an individual cell, or a whole row or column within the X, Y memory array, appears to be stuck at a particular value 0, or 1, on read. Alternatively, the chip can catastrophically fail, and every location appears 'stuck'. Thus errors are stationary in time and chips do not 'repair themseves' in the sense that the error condition does not pass.

Let us assume that the chip failure rate is given by λ (e.g. $\lambda = 10^{-6}$ failures/h), where a failure is any 'stuck-at' fault which prevents the chip operating correctly. If $t = 0$ is a point in time at which the chip is functioning correctly, then, assuming constant failure rates [2], the probability of correct operation at time t is given by

$$R = e^{-\lambda t} \tag{1}$$

where R is called the 'chip reliability'. The probability of chip failure is given by

$$Q = 1 - R = 1 - e^{-\lambda t} \tag{2}$$

If we now consider an uncoded memory with a total of D chips, then the probability of correct operation of the system

Paper 1633E, first received 16th March and in final form 26th August 1981

Dr. Goodman is with the Department of Electronic Engineering, University of Hull, Hull, England. Prof. McEliece is with the Coordinated Science Laboratory, University of Illinois, Urbana-Champain, USA

is R^D, and the probability of system failure at time t is $(1-R^D)$.

If we assume that memory chips fail independently at random, then the system mean time to failure [3] (MTTF) is

$$(MTTF) = 1/\lambda D \tag{3}$$

Let us insert some numbers to get accustomed to these equations. Consider a 4-megaword RAM system operating with a 16-bit microcomputer. Assuming that the memory is built out of industry standard 4116 type 16k-by-1 bit dynamic RAMs, then the system takes the form of an array with 16 columns and $m = 4096/16 = 256$ chip rows, and $D = 4096$ devices. Given a device failure rate of $\lambda = 3 \times 10^{-7}$, then from eqn. 3, the system MTTF is 813 h or approximately one month. Alternatively, from eqn. 2 we find that the probability of system failure at the 48 h point is 6%. This is clearly unacceptable for many applications, and implies that, on average, 6% of these systems will function correctly for only 48 h.

3 System lifetimes

There are several ways to assess the improvement in 'lifetime' of a memory system due to coding. First, it is possible to calculate the mean time to failure (MTTF) for both coded and uncoded systems. We have done this, but the calculation is lengthy, and we prefer to omit it, as not much insight into the problem is gained. Alternatively, mean time between failures (MTBF) can be calculated if renewal times can be assumed.

In this paper, however, we consider the lifetime $T_r(\alpha)$ of the system to be the time at which the probability of system failure equals some value $(1-\alpha)$. A special case is the time at which the probability of system failure is 1/2, and this time is the median time to failure (mTTF).

4 General analysis of system lifetime

Consider a memory system to be composed of an array of n columns by m rows of RAM chips, as shown in Fig. 1. The RAM chips are assumed to be 1-bit wide, so that the failure of a single chip only affects 1 bit in a horizontal word. Furthermore, we assume that the first k columns contain the data bits, where k is equal to the computer word size. The remaining $(n-k)$ columns contain the parity check bits.

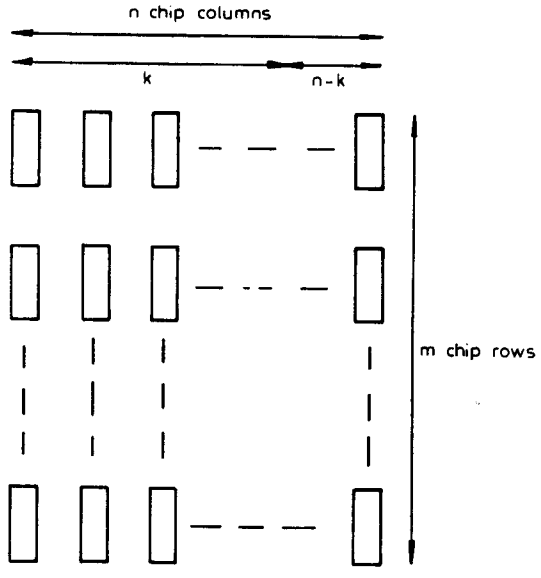

Fig. 1 *RAM organisation*

The efficiency (or rate) of the coded system is therefore k/n, and its redundancy is $(n-k)/n$. Note that an uncoded memory has $(n-k) = 0$.

Let us assume that each chip row of the above memory is coded by using a binary block code of length n bits, which is capable of correcting any combination of r errors amongst the n bits. The probability of correct operation of the row (or row reliability) is given by

$$R_R = \sum_{i=0}^{r} \binom{n}{k} (1-R)^i (R)^{n-i} \tag{4}$$

If the memory system is composed of m rows, then the probability of correct operation of the system (system reliability) is the probability that all m rows operate correctly. Conversely, the probability of system failure is the probability that one or more chip rows fails to operate correctly, because $r+1$ or more errors have occurred in a single row. The system reliability is therefore given by

$$R_S = (R_R)^m \tag{5}$$

The problem we wish to consider is that of inverting the equation

$$R_S = \alpha \tag{6}$$

i.e. inverting the equation

$$R_R = (\alpha)^{1/m} \tag{7}$$

to produce the solution $t = T_r(\alpha)$, which is the lifetime for the given level of performance α.

We now use an approximation to eqn. 4 to give an approximate solution to eqn. 7. Consider eqn. 4; if the quantities r/n and $r(1-R)$ are smaller than 1, as is certainly the case in this application, then the row reliability is well approximated by the Poisson distribution

$$R_R \simeq e^{-\mu} \sum_{i=0}^{r} \frac{\mu^i}{i!} \qquad \text{where} \quad \mu = n(1-R) \tag{8}$$

Let us define $\mu_r(\gamma)$ to be the solution to the equation

$$e^{-\mu} \sum_{i=0}^{r} \frac{\mu^i}{i!} = \gamma \tag{9}$$

Then the solution to eqn. 7 is well approximated by

$$(1-R) \simeq \frac{1}{n} \mu_r(\alpha^{1/m}) \tag{10}$$

Thus, from eqns. 2 and 10,

$$1 - e^{-\lambda t} \simeq \frac{\mu_r(\alpha^{1/m})}{n}$$

That is, the system lifetime is the solution

$$t = T_r(\alpha) \simeq \frac{1}{\lambda} \log \left\{ 1 - \frac{\mu_r(\alpha^{1/m})}{n} \right\} \tag{11}$$

Now, in all cases of interest, the ratio $\mu_r(\alpha^{1/m})/n$ will be small enough for the approximation

$$\log \left(1 - \frac{\mu}{n}\right) \simeq -\frac{\mu}{n}$$

to be very accurate. Hence, eqn. 11 becomes

$$T_r(\alpha) \simeq \frac{1}{\lambda n}\,\mu_r(\alpha^{1/m}) \tag{12}$$

For an uncoded memory, $n = k, r = 0$, and we may define the uncoded system lifetime as $T_0(\alpha)$, where, from eqn. 12:

$$T_0(\alpha) \simeq \frac{1}{\lambda k}\,\mu_0(\alpha^{1/m}) = \frac{1}{\lambda km}\log(\alpha^{-1}) \tag{13}$$

In this case, the solution $\mu_0(\alpha^{1/m}) = \log(\alpha^{-1/m})$ is exact and can be derived directly.

In order to compare equivalent coded and uncoded systems, let us define the 'coding gain' of the coded system $C_r(\alpha)$ to be the ratio of the system lifetime with coding, to that of the system without coding. Thus, from eqns. 12 and 13, we have:

$$C_r(\alpha) \simeq \frac{\mu_r(\alpha^{1/m})}{\mu_0(\alpha^{1/m})}\left(\frac{k}{n}\right) \tag{14}$$

From now on, the use of the $\simeq$ sign is discontinued and it must be inferred that our lifetimes and coding gains are approximations. It can be shown by direct numerical solution, however, that such approximations are very accurate for memory systems of practical interest.

5 Median time to failure (mTTF)

We may consider the median time to failure (mTTF) to be truly representative of the system lifetime in the general sense. In this case, $\alpha = 1/2$, and eqns. 12–14 become

$$(mTTF)_r = T_r(1/2) = \frac{1}{\lambda n}\,\mu_r(2^{-1/m}) \tag{15}$$

$$(mTTF)_0 = T_0(1/2) = \frac{1}{\lambda k}\,\mu_0(2^{-1/m}) \tag{16}$$

$$C_r(1/2) = \frac{\mu_r(2^{-1/m})}{\mu_0(2^{-1/m})}\left(\frac{k}{n}\right) \tag{17}$$

Let us now consider some special cases.

5.1 Uncoded memory

First, the uncoded case, i.e. $r = 0$. Clearly then, $\mu_0(2^{-1/m}) = (1/m)\log 2$ from eqn. 13, and eqn. 16 becomes

$$(mTTF)_0 = T_0(1/2) = \frac{1}{\lambda km}\log 2 = \frac{0.693}{\lambda km} \tag{18}$$

This is, in fact, the exact value of mTTF for this case, as can be verified directly.

5.2 Single-row, single-error correction

Next consider $r = 1$, $m = 1$, i.e. a single-chip row with single-error correction. One can verify numerically that $\mu_1(1/2) = 1.678$, so that by eqn. 15:

$$(mTTF)_1 = T_1(1/2) = \frac{1.678}{\lambda n} \tag{19}$$

and the coding gain is

$$C_1(1/2) = \frac{1.678}{\log 2}\left(\frac{k}{n}\right) = 2.42\left(\frac{k}{n}\right) \tag{20}$$

Let us apply eqn. 20 to the case of a $k = 16$-bit computer word encoded with the $n = 21$ Hamming single-error-correcting code. The rate of this code is $k/n = 0.762$ and eqn. 20 becomes

$$C_1(1/2) = 1.84 \tag{21}$$

Quantitatively, this means that a single-chip row of memory will have a lifetime of approximately 1.8 times that of the equivalent uncoded memory.

5.3 Single-row, multiple-error correction

Consider now $m = 1$, and increasing values of r. Table 1 shows r against $\mu_r(1/2)$ and the coding gain $C_r(1/2)$. From Table 1,

Table 1: Coding gain against r

r	$\mu_r(1/2)$	$C_r(1/2)$
0	0.6931	1 × (k/n)
1	1.678	2.4 × (k/n)
2	2.674	3.9 × (k/n)
3	3.672	5.3 × (k/n)
4	4.671	6.7 × (k/n)
5	5.6702	8.2 × (k/n)
6	6.6696	9.6 × (k/n)
7	7.66925	11.1 × (k/n)
8	8.66895	12.5 × (k/n)
9	9.668715	13.9 × (k/n)
10	10.66852	15.4 × (k/n)

it seems clear that $\mu_r(1/2) \simeq r + 2/3$, and in fact it can be shown (Appendix 10) that

$$\mu_r(1/2) = r + 2/3 + \frac{8}{405}r^{-1} - \frac{64}{5103}r^{-2} + \ldots \tag{22}$$

The coding gain for large r is therefore approximated by

$$C_r(1/2) = (1.44\,r + 0.196)\frac{k}{n} \tag{23}$$

It is interesting to note that the relative improvement in coding gain decreases rapidly with increasing r so that the benefits of coding are subject to rapidly 'diminishing returns'. For example, in Section 5.2 we saw that, for $k = 16$, a single-error-correcting code increases the lifetime of a single word by 1.84. If an $n = 26$ double-error-correcting code is used, the single-word lifetime is increased by a factor of $3.9 \times (16/26) = 2.4$ over uncoded, which is only a factor of 1.3 better than the single-error-correcting code.

6 Case of memory with many chip rows

It is clearly possible to extend the analysis of Section 5.3 to the case of multiple-chip rows, i.e. $m = 2$, 3 etc. However, we now consider the case of m large, i.e. a computer memory with many chip rows. In this case, $\mu_r(\alpha^{1/m})$ can be approximated by noting that

$$\alpha^{1/m} \simeq 1 + \frac{1}{m}\log\alpha \tag{24}$$

and that, for small values of μ,

$$e^{-\mu}\sum_{i=0}^{r}\frac{\mu^i}{i!} \simeq 1 - \frac{\mu^{(r+1)}}{(r+1)!} \tag{25}$$

Thus, for large m, we have the approximation:

$$\mu_r(\alpha^{1/m}) = (r+1)!^{1/(r+1)}m^{-1/(r+1)}(\log 1/\alpha)^{1/(r+1)} \tag{26}$$

The system lifetime is then given by eqn. 12 as

$$T_r(\alpha) = \frac{1}{\lambda n}\mu_r(\alpha^{1/m}) \tag{27}$$

The coding gain from eqns. 14 and 26 is then

$$C_r(\alpha) = \frac{k}{n}(r+1)!^{1/(r+1)} m^{r/(r+1)} (\log \alpha^{-1})^{-r/(r+1)} \tag{28}$$

For $r \gg 1$, eqn. 28 can be approximated via Stirling's formula for $n!$ to be:

$$C_r(\alpha) \simeq \frac{k}{n}\frac{(r+1)}{e} m \frac{-1}{\log \alpha} \tag{29}$$

6.1 Median time to failure with m large

The coding gain for m large can be found by putting $\alpha = 1/2$ into eqns. 28 and 29. Table 2 shows the coding gain against

Table 2: Coding gain against r

r	$C_r(1/2)$	
1	1.699	$\times \frac{k}{n} \times m^{1/2}$
2	2.32	$\times \frac{k}{n} \times m^{2/3}$
3	2.91	$\times \frac{k}{n} \times m^{3/4}$
4	3.49	$\times \frac{k}{n} \times m^{4/5}$
5	4.06	$\times \frac{k}{n} \times m^{5/6}$
≫ 1	$0.53(r+1) \times \frac{k}{n} \times m$	

r. Table 2 (or eqn. 28) can be used to give an accurate estimate of the median time to failure $T_r(1/2)$ for any given memory system.

For example, consider a 16-bit computer word, and a memory with 64 rows of chips. If the memory is coded with the $n = 21$ single-error-correcting Hamming code, and the chip failure rate is 10^{-6}, then eqn. 18 gives the uncoded mTTF as 677 h. From Table 2, the coding gain is 10.36, giving a coded mTTF of $10.36 \times 677 = 7011$ h. It is interesting to note that an exact solution to this example, obtained via eqn. 14, gives a coding gain of 10.87 and an mTTF of 7359 h. This shows that the above approximations are reasonably accurate.

6.2 Case of α close to 1

We may wish to know the lifetime of a coded or uncoded system to a point in time at which $\alpha = 0.9, 0.99, 0.999$ etc. That is, the time at which the probability of system failure is 10%, 1%, 0.1% etc. This result can be approximated by noting that $\log(\alpha^{-1}) \simeq (1-\alpha)$, for α close to 1. In this case, the uncoded memory has, from eqn. 13, a lifetime of

$$T_0(\alpha) = \frac{1}{\lambda km}(1-\alpha) \tag{30}$$

and eqn. 28 becomes

$$C_r(\alpha) = \frac{k}{n}(r+1)!^{1/(r+1)} m^{r/r+1} (1-\alpha)^{-r/r+1} \tag{31}$$

Let us use this formula in an example. Consider the 16-bit memory system defined in Section 6.1. Taking ($\alpha = 0.99$), we wish to find the time at which the system has a 1% failure probability. The uncoded system has a 1% failure probability at a time given by eqn. 30, as $T_0(0.99) = 1/\lambda km\,(0.01) = 9.76$ h.

From eqn. 31, the coding gain for the coded memory is $C_1(0.99) = 16/21\sqrt{2} \times 8\sqrt{100} = 86.2$, which gives a coded system lifetime of 841 h (or approximately one month) at the 1% point. Again, the argument may be turned around, to say that, on average, 1% of these systems will only last one month.

7 Conclusions

In this paper, formulas are developed which will enable a system designer to calculate the improvement in reliability that can be obtained by applying coding to a semiconductor memory system. The designer first calculates the lifetime of the uncoded memory system, and then simply multiplies this by the coding gain factor to yield the coded system lifetime.

In addition, the designer can use the formulas to calculate what percentage of a particular mass-produced memory system will last a given length of time.

By using these formulas, the system designer can assess whether or not a particular coding scheme (including no coding) will give his memory system its required reliability.

8 Acknowledgments

The authors would like to acknowledge partial financial support from the UK Science and Engineering Research Council and the Joint Services Electronics Program (USA), contract N00014-78-c-0424.

9 References

1 EUZENT, B.: 'Intel 2116 W-channel silicon gate 16K dynamic RAM'. Reliability report RR-16, Intel Corporation, 1977
2 ALNETHER, J.: 'Error detecting and correcting codes part 1'. Applications note AP146, Intel Corporation, 1979
3 LEVINE, L., and MEYERS, W.: 'Semiconductor memory reliability with error detecting and correcting codes', *Computer*, 1976, 9, pp. 43–50
4 WALKER, W.K.S., SUNDBERG, C.W., and BLACK, C.J.: 'A reliable spaceborne memory with a single error and erasure correction scheme'. *IEEE Trans.*, 1979, C-28, pp. 493–500
5 CARTER, W.C., and McCARTHY, C.E.: 'Implementation of an experimental fault-tolerant memory system', *ibid.*, 1976, C-25, pp. 557–567
6 GOODMAN, R.M.F.: 'Error correction coding for VLSI memories'. *IEE Colloquium Digest 1980/41*, 1980, pp. 119–122
7 HOEL, P.G., PORT, S.C., and STONE, C.J.: 'Introduction to probability theory' (Houghton-Mifflin, Boston, 1971)
8 FISHER, R.A., and CORNISH, E.A.: 'The percentile points of distributions having known cumulants', *Technometrics*, 1960, 2, pp. 209–225
9 ABRAMOWITZ, C.M., and STEGUN, I.A., (Eds.) 'Handbook of mathematical functions' (Dover, New York, 1965)

10 Appendix: Asymptotic expression for $\mu_r(\gamma)$

The object is to obtain an asymptotic approximation for $\mu_r(\gamma)$, defined as the solution to the equation

$$e^{-\mu}\sum_{i=0}^{r}\frac{\mu^i}{i!} = \gamma$$

which is valid for fixed γ and large r and, in particular, to obtain the approximation (eqn. 22) for $\gamma = \frac{1}{2}$. Our result depends on the following well known facts from probability theory:

(*a*) If X is a Poisson random variable with mean μ, and if Y is a random variable with the gamma density function

$\Gamma(n)^{-1}e^{-t}t^{n-1}, t>0$, then $Pr\{X \leqslant n-1\} = Pr\{Y \geqslant \mu\}$.* Thus, if $F_n(y)$ denotes the cumulative distribution function of Y, $\mu_{n-1}(\gamma)$ is the solution to the equation $F_n(\mu) = 1-\gamma$.

(*b*) If $Y_1, Y_2, \ldots, Y_n$ are independent, identically distributed random variables, each with the geometric density function $e^{-t}, t>0$, the sum $Y = Y_1 + \ldots + Y_n$ has the gamma distribution cited above.†

(*c*) If $Y = Y_1 + Y_2 + \ldots + Y_n$ is a sum of independent, identically distributed random variables, and if $F_n(y)$ is its cumulative distribution function, there is an asymptotic expression (with respect to n) for the solution $y(\gamma, n)$ to the equation $F_n(y) = 1-\gamma$. This expression, the Cornish-Fisher expansion [8], can be viewed as a generalisation and inversion of the central limit theorem. It depends on the moments of the Y_i's and the solution x of the equation

$$\frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-t^2/2}dt = \gamma,$$

In the special case, where each Y_i has the exponential density e^{-t}, a straightforward application of the formulas given in Reference 8 yields the following expression for $y(\gamma, n)$, which is, by our preceding remarks, also equal to $\mu_{n-1}(\gamma)$:

$$y(\gamma, n) = \mu_{n-1}(\gamma) = n + n^{1/2}x + \frac{1}{3}(x^2-1) + n^{-1/2}\left(\frac{x^3-7x}{36}\right) - n^{-1}\left(\frac{3x^4+7x^2-16}{810}\right) + n^{-3/2}\left(\frac{9x^5+256x^3-433x}{38880}\right) + n^{-2}\left(\frac{12x^6-243x^4-923x^2+1472}{204120}\right) + \ldots$$

This formula gives very good results even for modest values of n. For example, with $n = 11$, $\gamma = 0.1$, we find (e.g. from Table 26.5 of Reference 9) that $x = 1.28155$, and the first four terms of the above expression give $\mu_{10}(0.1) = 15.40704$, whereas the actual value is 15.40664.

In the special case $\gamma = \frac{1}{2}$, we have $x = 0$, and the preceding asymptotic expansion gives

$$\mu_{n-1}(\tfrac{1}{2}) = n - \frac{1}{3} + \frac{8}{405}n^{-1} + \frac{184}{25515}n^{-2} + \ldots$$

On replacing n by $n+1$, we obtain the advertised expansion (eqn. 22), namely,

$$\mu_n(\tfrac{1}{2}) = n + \frac{2}{3} + \frac{8}{405}n^{-1} - \frac{64}{5103}n^{-2} + \ldots .$$

*op.cit., Reference 7, section 5.3.3.

†*Ibid.*, Chap. 6, theorem 5