Selection of the Optimal Memory Configuration in a System Affected by Soft Errors

Juan Antonio Maestro, *Member, IEEE*, and Pedro Reviriego, *Member, IEEE*

***Abstract*—The selection of the best memory configuration is a challenge for designers when systems are affected by soft errors. When memory reliability is an issue and scrubbing is not recommendable, multibit protection codes are one of the available options. In this paper, the reliability of memories protected with those codes is studied. First, a method to analytically approximate the mean number of events to failure of memories protected with error correction codes capable of handling multiple bit errors is presented and validated through simulation experiments. Then, the selection of the optimal protection code for a given memory configuration in terms of memory size, word length, and target reliability level is analyzed. This selection process is illustrated using a practical case study. The study is then extended to determine, for a given error correcting code, the maximum memory size that would meet a target reliability level. Finally, the effect of the memory word size on the system reliability is considered by comparing different options. In summary, the proposed methods can be useful for designers when choosing the memory configuration at the system level for critical applications in which reliability is a major concern.**

***Index Terms*—Memory, multibit error correction codes, reliability, single-event upsets (SEUs), soft errors.**

I. Introduction

RELIABILITY is a critical factor for memories [1]–[4] when they operate in environments where there are many sources of error [5], [6], for example, space [6], [7]. Aside from the higher probabilities of errors, these environments may present other difficulties [8]–[10] like distance, which makes systems physically unreachable, preventing the possibility of performing maintenance operations *in situ*. Due to these reasons, it is mandatory that the correct design decisions are taken, so that the reliability level is high enough in order to guarantee the correct operation of the system. However, if the event arrival rate is too high, there will be a large number of soft errors, for example, single-event upsets (SEUs) [11]–[14]. In this case, reliability may decrease a lot, which would make the system nonoperative for many applications. SEUs will be considered the main source of soft errors in the rest of this paper.

Manuscript received February 16, 2009; revised April 17, 2009. First published May 15, 2009; current version published September 2, 2009. This work was supported in part by the Spanish Ministry of Science and Education under Grant ESP-2006-04163, by the Regional Government of Madrid, and by the European Union FEDER programme.

The authors are with the Universidad Antonio de Nebrija, 28040 Madrid, Spain (e-mail: jmaestro@nebrija.es; previrie@nebrija.es).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TDMR.2009.2023081

Traditionally, memories have been protected with single error correction (SEC) codes [15], [16] that can correct up to one error per memory word. To avoid the accumulation of errors in the memory, which would eventually cause a failure when a second SEU affects a bit of a word already affected by a previous one, scrubbing is used [17]. The scrubbing process periodically reads the memory words and corrects the errors, so that a failure occurs only if two errors arrive in the same scrubbing period. This is usually a valid solution, but sometimes, it may not be feasible or desirable. For example, considering a memory, if power saving is an issue, then scrubbing would not be the most convenient option since it would increase power consumption or, if the memory is very frequently accessed due to hard time constraints, there may be no idle cycles to perform the scrubbing process.

There are other options based on redundancy, like adding extra memory chips and voting for the majority value whenever data are accessed. This usually produces a high cost overhead. Another solution would be to add more powerful error detection and correction codes to the memory. As discussed before, it is usual that memories implement SEC codes, but although this increases reliability significantly compared to an unprotected memory, sometimes, it is not enough.

Introducing codes that are able to correct a number of errors higher than one is feasible [17]–[19], but these codes need to be carefully chosen as they usually introduce some drawbacks. The first one is cost, since multibit protection implies additional redundancy. This cost includes the extra bits added to each word plus the correction and protection logic, which suffers an increment of complexity. Moreover, performance may be affected, because the mentioned complexity would eventually increase the data codification and decodification time.

In this paper, methods to help the designer choose the most suitable multibit protection code and memory word size for a system suffering soft errors will be offered. These methods will allow the selection of the optimal memory configuration for a given application in the early stages of the design process.

The rest of this paper is organized as follows: The problem assumptions and reliability estimates are introduced in Section II; a methodology to select the optimal protection level in a memory of a given word size is presented in Section III; then, in Section IV, the selection of an optimal word size is considered; and finally, some conclusions will be shared in Section V.

II. Problem Assumptions and Reliability Estimates

The following assumptions will be used for this paper.

1) Memories are protected using an L-bit protection code. This implies that each word has some redundant bits c, which are able to correct up to L simultaneous errors in the same word.
2) Memories do not implement scrubbing. Errors accumulate in the memory until $L+1$ hit the same word. At that point, the protection codes cannot handle the situation, which leads to a memory failure. The effect that some words may be rewritten by the application, thus eliminating the errors already accumulated on them, is not considered, which will lead to a conservative approach in the following sections.
3) Events are supposed to arrive following a Poisson distribution.

In order to characterize the quality of the protection codes, the following parameters will be used:

1) M: the number of words per memory;
2) N: the number of data bits per word (excluding protection bits);
3) c: number of redundant bits per word introduced by the protection code;
4) mean time to failure (MTTF): the factor that measures reliability in this approach. It represents how much time the memory should work on average until it fails;
5) event arrival rate per word (λ_{word}): number of events (SEUs) that are produced in the memory per word and unit of time. On the other hand, the event arrival rate per memory will depend both on the previous parameter and the number of words in the memory: $\lambda_{\text{memory}} = \lambda_{\text{word}} \cdot M$. Both parameters λ_{memory} and λ_{word} depend on the memory width. The higher the number of redundant bits c, the more likely to suffer a soft error. Therefore, the event arrival rate implicitly depends on L (since selecting a certain L-bit protection code would determine c);
6) mean number of events to failure (METF): this represents how many events (SEUs) the memory has suffered on average until reaching failure (during the time determined by the MTTF). Following a well-known relation for Poisson processes (see [20])

$$MTTF = METF/\lambda_{\text{memory}} = METF/(\lambda_{\text{word}} \cdot M). \tag{1}$$

This relation will be used in the techniques presented in this paper.

The MTTF will be used in the rest of this paper as the estimate of the reliability since it is simple enough and gives an intuitive measure of the expected time to failure of the memories. However, to make the results independent of the event arrival rate λ_{word}, the METF will be used in the following, since the calculation of the MTTF having the METF is straightforward using (1). For the calculation of METF, an approximation has been proposed [20], [21] for the case $L = 1$ and $M \gg 1$

$$METF|_{L=1} \cong \sqrt{\frac{\pi \cdot M}{2}}. \tag{2}$$

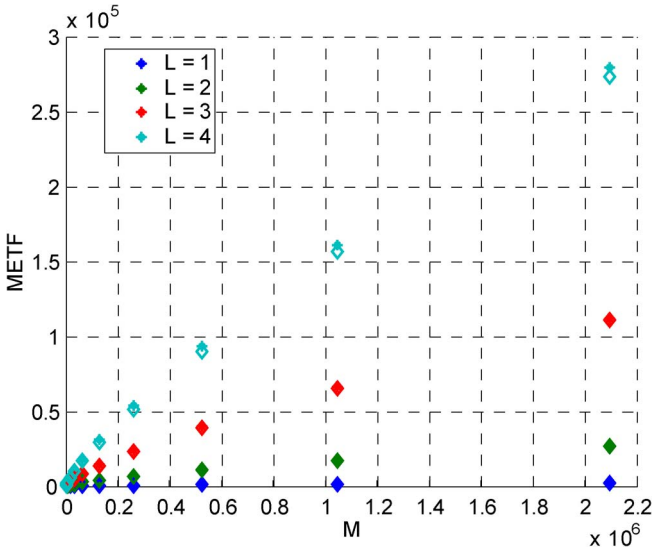

Fig. 1. Evolution of the METF obtained by (stars) simulation and (diamonds) approximation for different memory sizes (M) and several multibit protection codes (L).

However, to the best of the authors' knowledge, no analytical expression for the MTTF in the case of multibit protection codes $(L > 1)$ has been proposed.

This problem can be tackled by noticing that the memory failure model is equivalent to the classical "birthday surprise" problem [22] that has been extensively studied in the area of statistics [23]. In this case, we could see the multibit protection codes as a generalized problem, where at least $L+1$ people share the same birthday [24], and therefore, previous results from statistics can be applied [25]–[27].

As a conclusion, we propose the following approximation (where $\Gamma(x)$ is the gamma function), which can be used when M is large (see [26]). In fact, it can be seen that (2) is a particular case of (3) for $L = 1$

$$METF|_L \cong \sqrt[L+1]{(L+1)!} \cdot \Gamma\left(1 + \frac{1}{L+1}\right) \cdot M^{\frac{L}{L+1}}. \tag{3}$$

In order to check the accuracy of the approximations for the problem under study, a set of simulations has been conducted. The simulations have been designed to recreate several memories, with different sizes and protection codes. The selected values of L and M cover the practical range of L and the lower range of the values of M, where the approximations will be less accurate. For those values, events have been induced and the number of events to failure measured for a large number of experiments. Then, the average of all the experiments has been calculated and used as an estimate of the METF. The results are shown in Fig. 1, where the simulated values are plotted with a star and the values predicted by the approximations are plotted with diamonds. It can be seen that, for most memory sizes, the approximation given by (3) is accurate. This is more clearly observed in Fig. 2, where the ratio of the METF obtained by simulation to the METF given by approximation (3) is shown. The difference between simulation and approximation is below 5% for most memory sizes. In summary, approximation (3) can be used to accurately estimate the METF.

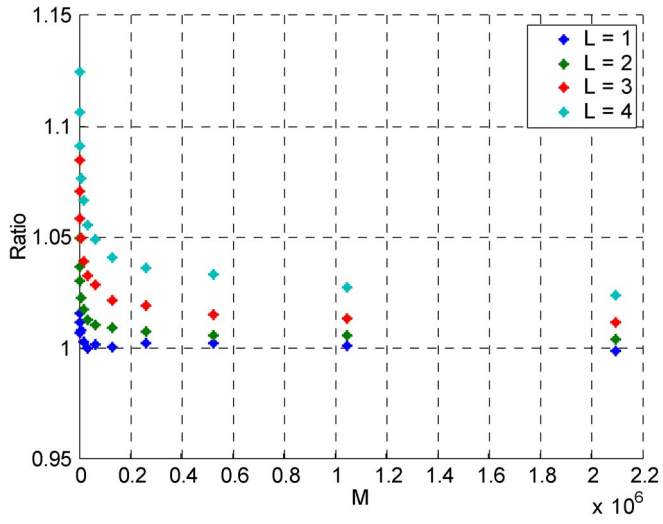

Fig. 2. Ratio of the METF obtained by simulation to the METF given by approximation (3) for different memory sizes (M) and several multibit protection codes (L).

From the results, it can also be observed that, given a certain memory size M, the METF of the system increases with L, as expected. Moreover, the METF increases when M grows, as explained before. The growth with M is given by (3) and increases with L. For large values of L, it can be seen that the growth of METF tends to be linear with M

$$O(METF)|_{L\gg 1} \rightarrow O(M). \tag{4}$$

For smaller values of L, which is the usual case, the growth of METF with M will be sublinear. Moreover, the denominator of (1) $\lambda_{\text{word}} \cdot M$ is obviously linear with M. Then, it is clear that both facts make the MTTF decrease with M.

III. Selection of the Protection Level: Optimal MTTF Versus Cost

In this section, a methodology to help the designer determine the most suitable configuration for a given system memory will be described. Suitability implies finding the minimal L which makes reliability (MTTF) high enough to meet the application requirement. It is important that L is minimal, because area cost and complexity grow as the protection level is increased. Formally, the problem can be stated as follows:

> "*Given a memory in a system suffering soft errors, characterized as* $\{M, \lambda_{\text{word}}, MTTF_{\text{Target}}\}$*, where* M *is the size in words,* λ_{word} *is the event arrival rate per word due to radiation, and* $MTTF_{\text{Target}}$ *is the target reliability, find the minimal* L *such that the reliability is met and the cost* c *is optimal*."

There are other costs associated to the protection codes apart from the redundant bits c, for example, the encoding and decoding logic. Their impact on the overall cost should be negligible in most cases, and therefore, they are not considered in this paper.

The technique to find out L is divided into two consecutive steps.

A. Step 1: Determination of the Required METF

Although the imposed constraint is to meet a certain MTTF, it is better to reduce the problem to work with the METF, as discussed before. The reason of this is that METF is independent of λ_{word} (contrary to MTTF), which is more convenient to make the study more general.

Therefore, the initial step would be to determine the mean number of events to produce a failure associated to a given MTTF. This is straightforward, since there is a relation that works with both magnitudes (see Section II). Just making $METF = MTTF \cdot (\lambda_{\text{word}} \cdot M)$, the right value would be obtained. However, a side effect introduced by the L-bit protection codes has to be taken into account. Since these codes add some redundant bits, the size of the memory is effectively increased, and therefore, the probability of receiving more events also grows for a given memory technology and radiation environment. In this way, the error event arrival rate is directly proportional to the size of the memory in bits.

If, as defined before, the number of redundant bits per word is c, then the area would be incremented by a factor $f = (N + c)/N$, where N is the original number of data bits per word. As the event arrival rate per memory $(\lambda_{\text{word}} \cdot M)$ is linear with the memory size, the same increment applies here. Therefore, the required $METF_{\text{Target}}$ would be calculated as

$$METF_{\text{Target}} = MTTF_{\text{Target}} \cdot (\lambda_{\text{word}} \cdot M) \cdot f. \tag{5}$$

An important thing to notice is that factor f depends directly on the protection code type. At this point, the decision on which that protection code should be has not been made yet. Therefore, a collection of METFs should be calculated, one for each value of L that will be considered

$$METF_{\text{Target}}|_L = MTTF_{\text{Target}} \cdot (\lambda_{\text{word}} \cdot M) \cdot f_L \tag{6}$$

$$f_L = \frac{N + c_L}{N} \tag{7}$$

where c_L is a function of L. The values that have been considered for L in the experiments are $L = 1$, 2, 3, and 4, which are the most typical values when protecting memories. This set can be expanded with other values if a given application requires it.

With this set of $\{METF_{\text{Target}}|_L\}$, the second step will be to find out the most appropriate protection code to meet the reliability constraint.

B. Step 2: Calculation of the Optimal L

There are several considerations that have to be made about multibit protection codes.

1) The area cost grows with L, but this varies from code to code, and also with the word size (N) as given by the factor (7). This is particular to each specific protection code, and therefore, it should be studied for each case.
2) METF and MTTF also grow with L. With this, the reliability can be increased until the constraint is met.
3) MTTF decreases with memory size. In other words, the larger the memory is, the more likely it will fail after

a certain time. However, METF increases with M, because as the memory size grows, it will be less likely that multiple events hit the very same word. This may seem contradictory if (1) is analyzed, because MTTF and METF are directly proportional. The reason for this is that $(\lambda_{\text{word}} \cdot M)$ also increases with M, and that increment grows quicker than METF, as can be seen in (3). Therefore, $METF/(\lambda_{\text{word}} \cdot M)$ will decrease with M, as discussed before.

There are three magnitudes that need to be considered when analyzing the memory: $\{METF_{\text{Target}}|_L\}$, M, and L. Notice that λ_{word} is not relevant at this point, because the working parameter is METF and not MTTF.

1) Optimal L to Meet Reliability Requirements for a Memory Size M: In the following, the algorithm to find out the most suitable L is presented.

1) The first value of $METF_{\text{target}}|_L$ with $L = 1$ would be considered.
2) Using approximation (3), the $METF|_L$ for that value of M would be calculated.
3) If that value is larger than $METF_{\text{target}}|_L$, then an L-bit protection code would be enough to meet the MTTF constraint.
4) If the value is smaller than $METF_{\text{target}}|_L$, then that protection is not enough. The process should be repeated from 1), with $L \leftarrow L + 1$ until the protection level is enough.

With this, the optimal protection for the given memory would be obtained, minimizing area cost while meeting the reliability constraints.

2) Optimization of M for the Selected Protection Level L: Once the most appropriate value of L has been chosen, a range of M is delimited in which the reliability constraints are met. This means that, although the protection level has been selected based on an initial memory size estimated by the designer, there may be larger sizes that are potentially protected by that value of L. Let us define M_{initial} as the initial size estimated by the designer, based on the application and the system requirements. If, now, a protection for L bits is added, the value of $M_{\max}$ will be calculated in order to determine how much the memory size could increase while meeting the reliability constraints. Then, the designer will have the option of reconsidering the initial memory size M_{initial} with an M_{final} in the interval $[M_{\text{initial}}, M_{\max}]$. In other words, assuming the cost of implementing L, more memory could be used, if an analysis of the system advises so. This also determines the upgrade capability for future expansions of the system.

Using (1) and (3), the value $M_{\max}$ can be obtained as

$$M_{\max} \cong \left(\frac{\sqrt[L+1]{(L+1)!} \cdot \Gamma\left(1 + \frac{1}{L+1}\right)}{\lambda_{\text{word}} \cdot MTTF_{\text{target}}} \right)^{L+1} = (L+1)! \cdot \left(\frac{\Gamma\left(1 + \frac{1}{L+1}\right)}{\lambda_{\text{word}} \cdot MTTF_{\text{target}}} \right)^{L+1} \quad (8)$$

where λ_{word} is the arrival rate per memory word, which is a function of L (since each protection code will add a different number of redundant bits to each word, as commented before).

In the next section, a case study will be discussed, where real values are used to illustrate the presented methodology.

C. Case Study

In this case study, the details of an experiment that will be used to illustrate the proposed technique are presented. The problem will consist in deciding the most suitable protection code (L) for a memory in a system under radiation, working in a space application.

The initial memory size is $M_{\text{initial}} = 2^{20}$ words $(\approx 10^6)$, the memory width is $N = 16$, and the radiation source will be characterized with an event arrival rate per bit of $(\lambda_{\text{word}}/16) = \lambda_{\text{bit}} = 2 \cdot 10^{-8}$ events per day. This would correspond to a typical equatorial orbit (> 3000 km), as reported in [17].

The reliability requirement of the system has been chosen as five years of nonfailure work, and therefore, a very conservative constraint of 200 years will be used to calculate the protection (if a not-so-hard constraint is chosen, and since there is no scrubbing, failures could statistically appear before the five-year time frame).

The following three multibit protection alternatives have been proposed as candidates for this system:

1) $L = 1$, with a cost per word of $c_1 = 6$;
2) $L = 2$, with a cost per word of $c_2 = 11$;
3) $L = 3$, with a cost per word of $c_3 = 16$.

The codes for $L = 1$ and $L = 2$ have been analyzed in [18], while the code for $L = 3$ uses the same cost overhead as the Golay triple code reported in [17].

The first step consists in calculating the required METF for each of the proposed codes, using (6).

The values of f_1, f_2, and f_3 are 1.38, 1.69, and 2, respectively, and using the λ_{bit} stated at the beginning of this section, the following set of required METFs is obtained:

1) $METF_{\text{target}}|_1 = 3.3680 \cdot 10^4$;
2) $METF_{\text{target}}|_2 = 4.1335 \cdot 10^4$;
3) $METF_{\text{target}}|_3 = 4.8989 \cdot 10^4$.

The second step is to determine the minimum L that meets the obtained METFs using approximation (3). The results are shown in Fig. 3, where the values of the required METFs for the memory size are plotted.

It can be observed that the required METF for $L = 1$ is over the value provided by the protection, and therefore, this code is not enough. In other words, the selected code for $L = 1$ would fail with a lower number of events than required. The same happens for $L = 2$, whose required METF is also over the associated graph.

However, the METF required for $L = 3$ is below the value offered by that protection code. In other words, that code will need more time to fail than the imposed constraint and therefore would be valid for this application. This would then be the optimal code.

Once L has been chosen, the next step will be to determine the maximum size M that supports the reliability constraints.

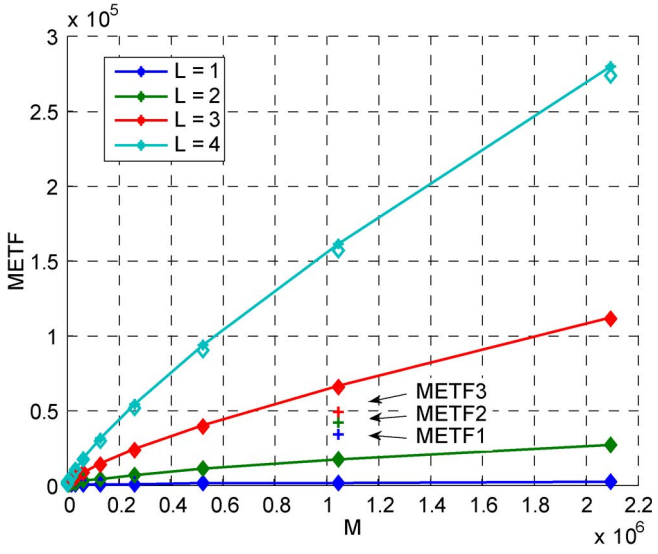

Fig. 3. METF versus M and L for the proposed case study.

Using (8), the following values are obtained (although the case $L = 3$ has been chosen, values for $L = 1$ and 2 are also provided for illustrative purposes):

1) $M_{\max}|_{L=1} = 1522$;
2) $M_{\max}|_{L=2} = 69\,747$;
3) $M_{\max}|_{L=3} = 3\,400\,041 \approx 3.4 \cdot 10^6$.

It can be seen that the selected protection value of $L = 3$ is able to support a higher memory size than the 10^6 words of the initial memory system. Therefore, this protection level is enough for this particular application. Moreover, this implies that a larger memory of up to $3.4 \cdot 10^6$ words could be used, since the reliability offered by the system would meet the imposed requirements.

This information could be used by the designer, not only to find out the best L but also to determine the maximum memory size able to work with that protection level.

D. Study of the Evolution of $M_{\max}$ With L

It has been proven in the previous section that increasing L not only produces an increment on the protection level but also in the memory size able to support that level. Since the choice of the memory size is an important decision for the system at the designer level, it is interesting to study how $M_{\max}$ evolves with L.

If the following expression is met:

$$\frac{\Gamma\left(1+\frac{1}{L+1}\right)}{\lambda_{\text{word}}|^L \cdot MTTF_{\text{target}}} > 1 \tag{9}$$

it is clear that the value of $M_{\max}$ will grow with L. This is the case in most situations, as $\lambda_{\text{word}}|^L \cdot MTTF_{\text{target}}$ is the average number of errors per word when a failure occurs, which is normally less than one. In other words, a large memory would normally fail before having an error in all its words. In Fig. 4, the values of $M_{\max}$ for different MTTF values and codes (L) are shown for a per-bit arrival rate of $\lambda_{\text{bit}} = 2 \cdot 10^{-8}$ and a memory word size of 16 data bits plus the corresponding redundant bits for each L (the codes are those used in the previous case study). The growth of $M_{\max}$ with L can be clearly observed. The dependence of $M_{\max}$ with the required MTTF value is also clear.

The growth of $M_{\max}$ with L can be analyzed in more details by looking at the ratio produced when incrementing L

$$Ratio|_{M_{\max}} = \frac{M_{\max}|^{L+1}}{M_{\max}|^L} \cong \frac{(L+2)}{MTTF_{\text{target}}} \cdot \frac{\left(\lambda_{\text{word}}|^L\right)^{L+1}}{\left(\lambda_{\text{word}}|^{L+1}\right)^{L+2}} \cdot \frac{\Gamma\left(1+\frac{1}{L+2}\right)^{L+2}}{\Gamma\left(1+\frac{1}{L+1}\right)^{L+1}} \tag{10}$$

where the fact that the per-word arrival rate increases with L has been taken into account (a linear growth of λ_{word} and L has been assumed in the following; see Table I in the next section). The reason of this increment is that higher values of L would introduce more redundant bits per word, making λ_{word} also higher. This increase reduces the growth of the ratio (10) with L, as shown in Fig. 5. The ratio has been illustrated for different values of L for the previous configuration ($\lambda_{\text{bit}} = 2 \cdot 10^{-8}$; $N = 16$) and a required MTTF value of 200 years. This means that, for large enough values of L, an increment in this magnitude would produce a constant ratio in $M_{\max}$. In other words, the growth of $M_{\max}$ with respect to L tends to be linear (since the increment ratio of $M_{\max}$, which, in fact, is the slope of this function, tends to be constant, as can be seen in Fig. 5). This may help the designer determine how much memory size under protection would get each time L is increased.

IV. Memory Word-Size Selecion

Another important decision for the designer is not only to decide the size of the memory but also its organization. In this way, for a given memory size, parameters such as word width and the total number of words are important to determine the reliability of the system.

It is well known that the relative cost of error protection decreases with the word size [18], as shown in Table I for $L = 1$ (SEC) and $L = 2$ (double error correction codes).

In this section, the effect of the organization in a memory of $M \cdot N$ bits will be studied. Initially, the memory will be organized with M words of N bits, and later, a different configuration of $M/2$ words and $2 \cdot N$ bits will be used. This will leave the memory size unchanged (therefore, this effect will not influence reliability) but with a different memory organization.

In terms of protection, if we assume that the first configuration is protected with a code capable of correcting L bits, two different options for the second configuration can be proposed.

1) Use also a code that can correct L bits.
2) Increase protection with a code that can correct $L + 1$ bits.

A. Protecting the Memory With a Code of L Bits

In this scenario, L is the same as in the initial configuration, and therefore, this factor will not influence the reliability of the

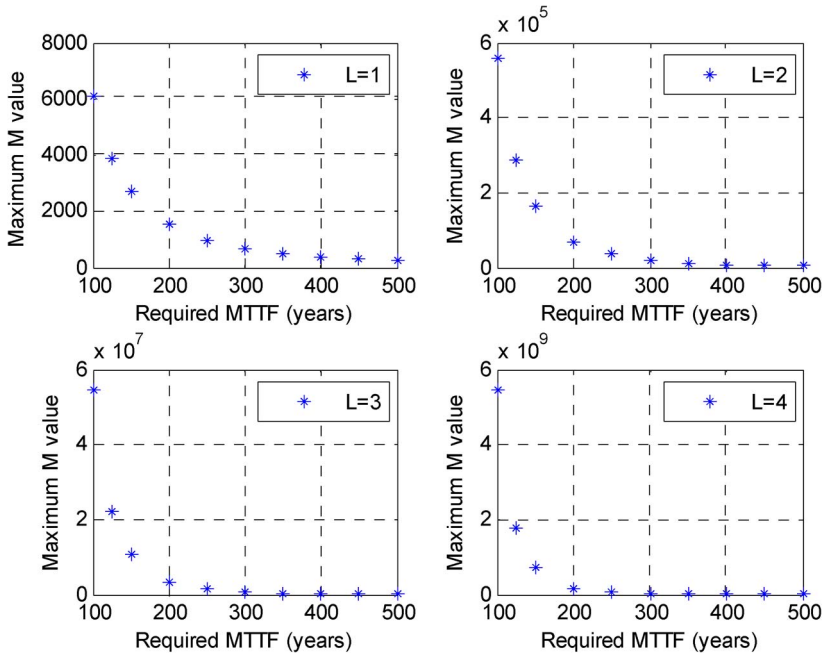

Fig. 4. Maximum memory size as a function of the required MTTF for different values of L.

TABLE I
CHECK BITS VERSUS WORD SIZE

Word Size	L=1	L=2
8	5	9
16	6	11
32	7	13
64	8	15

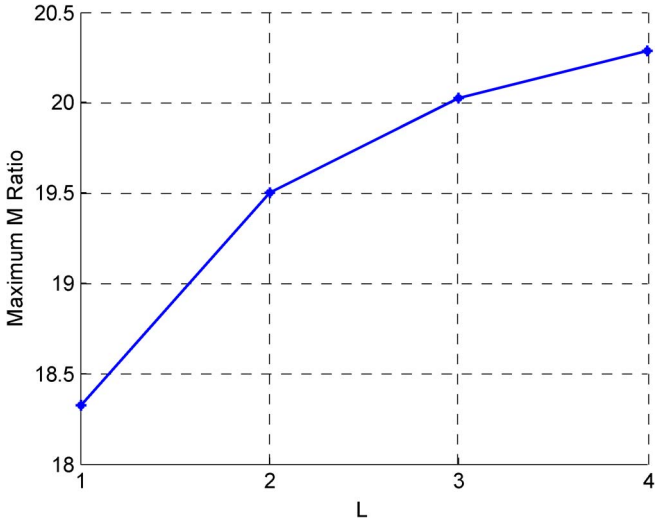

Fig. 5. Ratio of the maximum memory sizes for a required MTTF of a code that can correct $L+1$ bits and a code that can correct L bits.

system when changing the organization to the second configuration (double memory width).

Since L has not changed and, now, the memory width has been doubled, the protection overhead will be relatively lower than that in the initial configuration. In other words, the number of redundant bits per data bit decreases with the word size.

To compare the reliability (MTTF) of both memory configurations, an analysis of METF and $(\lambda_{\text{word}} \cdot M)$ will be performed, according to (1).

The METF for the first configuration is given by (3)

$$METF|_{L,M} \cong \sqrt[L+1]{(L+1)!} \cdot \Gamma\left(1+\frac{1}{L+1}\right) \cdot M^{\frac{L}{L+1}}. \quad (11)$$

In the same way, the METF for the second configuration is given by

$$METF|_{L,M/2} \cong \sqrt[L+1]{(L+1)!} \cdot \Gamma\left(1+\frac{1}{L+1}\right) \cdot \left(\frac{M}{2}\right)^{\frac{L}{L+1}}. \quad (12)$$

Analyzing the METF ratio of both configurations

$$Ratio_{\text{METF}} = \frac{METF|_{L,M/2}}{METF|_{L,M}} \cong \left(\frac{1}{2}\right)^{\frac{L}{L+1}}. \quad (13)$$

It can be seen that the new METF has decreased after changing the memory organization. However, since the protection overhead is lower for the second configuration (in relative terms), $(\lambda_{\text{word}} \cdot M)$ would also be lower, which will make the actual difference in terms of MTTF smaller than the ratio given by (13).

Thus, the main conclusion is that doubling the word size reduces the area of the memory (less redundant bits for the same number of data bits), at the expense of a reduction in the MTTF. This can be useful from the designer point of view, when area is an issue and has to be minimized, but reliability is not a problem: even if it decreases after the word size is changed, the imposed constraints can still be met. In other words, if the MTTF is substantially larger than the required MTTF, then

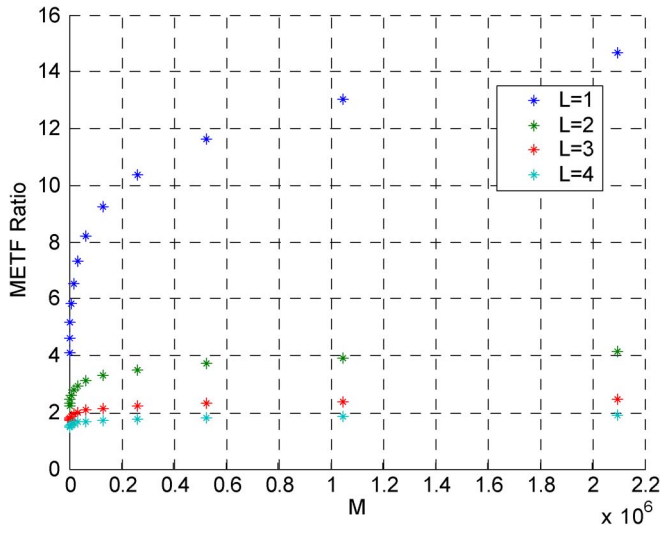

Fig. 6. METF ratio (15) versus M and L.

doubling the word size would reduce the area of the memory while still meeting the reliability requirements.

B. Protecting the Memory With a Code of $L + 1$ Bits

A different scenario is proposed now, in which the second memory configuration is protected up to $L + 1$ bits. In this case, the cost would be similar if the codes proposed in [18] (whose costs are shown in Table I) are used (or similar ones).

Having a similar protection cost in both configurations, let us study the effect of doubling the word size on the reliability.

The METF for the first configuration was given by (11).

Now, for the second configuration, this time protecting $L + 1$ bits, the METF would be

$$METF|_{L+1,M/2} \cong \sqrt[L+2]{(L+2)!} \cdot \Gamma\left(1+\frac{1}{L+2}\right) \cdot \left(\frac{M}{2}\right)^{\frac{L+1}{L+2}}. \tag{14}$$

The ratio of both parameters will be given by

$$Ratio_{\text{METF}} = \frac{METF|_{L+1,M/2}}{METF|_{L,M}} \cong \frac{\sqrt[L+2]{(L+2)!} \cdot \Gamma\left(1+\frac{1}{L+1}\right) \cdot M^{\frac{1}{(L+1)\cdot(L+2)}}}{\sqrt[L+1]{(L+1)!} \cdot \Gamma\left(1+\frac{1}{L+1}\right) \cdot 2^{\frac{L+1}{L+2}}}. \tag{15}$$

In this case, since the protection overheads are comparable (at least for $L = 1$; see Table I), $(\lambda_{\text{word}} \cdot M)$ will be very similar in both memory configurations, and therefore, the ratio of the METFs is a good approximation for the MTTF ratio. This ratio grows with M, as can be seen in Fig. 6. The first conclusion is that, if the memory size remains unchanged, increasing L will produce an increment on the METF and, therefore, in reliability.

This scenario is rather opposite to the one studied in the previous section. It will be appropriate when reliability needs to be increased, which can be achieved by using a higher order protection code. Thanks to the memory reorganization (less memory words and larger word size), this increment in the protection has been attained without increasing the protection cost (thus leaving the actual memory size unchanged).

The last analysis will be devoted to study; under which circumstances, an increment in L will produce a reliability improvement.

In most cases, the memory size M will be a power of 2 ($M = 2^K$), so that the dependence of (15) with the memory size can be expressed as

$$M^{\frac{1}{(L+1)\cdot(L+2)}} = 2^{\frac{K}{(L+1)\cdot(L+2)}}. \tag{16}$$

In this case, when $K \gg (L + 1) \cdot (L + 2)$, the ratio (15) will be larger than one. This provides a simple way to evaluate the benefits of increasing the word size for a given L and memory size. The larger the L is, the larger the memory needs to be to get a substantial benefit in the MTTF.

C. Case Study

As an example of the previous sections, let us consider a situation in which a configuration with $N = 8$ and $L = 1$ provides an MTTF that exceeds the required constraint by a factor f_{ex} (therefore, there is room to optimize the memory size).

If the word size is doubled, the MTTF for $L = 1$ can be obtained from the initial configuration by applying a correction factor to the METF and another one to the memory arrival rate $(\lambda_{\text{word}} \cdot M)$.

The ratio for the METFs would be given by (13)

$$Ratio_{\text{METF}} = \frac{METF|_{L=1,M/2}}{METF|_{L=1,M}} \cong \frac{1}{\sqrt{2}} = 0.707. \tag{17}$$

Considering that $(\lambda_{\text{word}} \cdot M)$ is proportional to the memory size, its ratio will be given by the area ratio of protecting the memory with $L = 1$ for word size N versus $2 \cdot N$ (see Table I)

$$Ratio_{\lambda_{\text{word}}\cdot M} = \frac{area|_{L=1,M/2}}{area|_{L=1,M}} = \frac{M(2N + c_{2N})}{2M \cdot (N + c_N)} = \frac{(16+6)}{2 \cdot (8+5)} = 0.8461 \tag{18}$$

where c_N and c_{2N} are the protection bits needed for word sizes N and $2N$, respectively. These values can be seen in Table I.

Therefore, the MTTF ratio would be [see (1)]

$$Ratio_{\text{MTTF}} = \frac{MTTF|_{L=1,M/2}}{MTTF|_{L=1,M}} = \frac{Ratio_{\text{METF}}}{Ratio_{\lambda_{\text{word}}\cdot M}} = 0.836. \tag{19}$$

Examining (18) and (19), two conclusions may be drawn. The first one is that, by modifying the memory configuration, the area has been reduced, since the new size is 84.61% of the initial one. The second conclusion is that the new reliability is 83.6% of the initial. If this reduction is acceptable (the initial excess value f_{ex} can support it), then the new configuration is

still feasible but with an important area optimization. This has been achieved by simply modifying the memory organization, as explained before.

Let us study the opposite situation, in which reliability needs to be increased. In order to achieve this, L will be increased to $L+1$, and the memory width will be doubled. If the initial values of $N=8$ and $L=1$ are assumed, then the increase of the MTTF would be substantial when $K \gg (L+1)\cdot(L+2)=6$, i.e., for memories much larger than $M=2^K=32$. This can be, in fact, observed in Fig. 6. It can also be proved that, in this scenario, each time that the memory size is doubled, an additional increment of the MTTF ratio (versus the initial configuration) of $2^{1/6}=1.122$ is achieved. This implies that, the larger the value of M is, the more reliability increment is achieved.

For larger values of L, like $L=4$, the increment in the MTTF would be substantial when $K \gg 30$, i.e., for memories much larger than $M=2^K=1\,073\,741\,824 \approx 10^9$. This can also be observed in Fig. 6. As a summary, increasing the protection when doubling the word size is more beneficial for small values of L and large values of M.

V. Conclusion

In this paper, the reliability of memories protected with multibit protection codes has been analyzed from the designer point of view. The idea behind this paper is to provide a set of analyses that can help in the decision making process when designing the optimal memory for a given system and application. Considerations like the best multibit protection code, the maximum memory size that can hold that protection, or the memory width are discussed and put in perspective. These analyses are based on previous results from statistics, where the same problem has been extensively studied from a different perspective. In this way, some approximations previously used in other problems have been shown to be valid in this case, in order to determine the optimal multibit protection codes. To achieve this, they have been validated through simulations to check their accuracy for the range of values of the different parameters that are of interest in the memory protection problem. Another contribution is the analysis of the implications of the memory word size on the reliability, for which different alternatives are proposed to select the optimal value in a given configuration. A number of case studies have also been presented to illustrate the multibit protection code and word-size selection processes that can help the designer in making accurate decisions at a system level.

About the future work, the application of the proposed methods for more complex codes and larger blocks of data, such as those used in magnetic storage and other systems, will be considered.

References

[1] T. Sasada, S. Ichikawa, and T. Kanai, "Measurement of single-event effects on a large number of commercial DRAMs," *IEEE Trans. Nucl. Sci.*, vol. 53, no. 4, pt. 1, pp. 1806–1812, Aug. 2006.

[2] R. C. Baumann, "Soft errors in advanced computer systems," *IEEE Des. Test Comput.*, vol. 22, no. 3, pp. 258–266, May/Jun. 2005.

[3] L. Schiano, M. Ottavi, and F. Lombardi, "Markov models of fault-tolerant memory systems under SEU," in *Rec. Int. Workshop Memory Technol., Des. Test.*, Aug. 2004, pp. 38–43. Issue 9/10.

[4] G. C. Cardarilli, A. Leandri, P. Marinucci, M. Ottavi, S. Pontarelli, M. Re, and A. Salsano, "Design of a fault tolerant solid state mass memory," *IEEE Trans. Rel.*, vol. 52, no. 4, pp. 476–491, Dec. 2003.

[5] J. F. Ziegler and W. A. Lanford, "The effect of sea level cosmic rays on electronic devices," *J. Appl. Phys.*, vol. 52, no. 6, pp. 4305–4318, Jun. 1981.

[6] C. A. Gossett, B. W. Hughlock, M. Katoozi, G. S. LaRue, and S. A. Wendler, "Single event phenomena in atmospheric neutron environments," *IEEE Trans. Nucl. Sci.*, vol. 40, no. 6, pp. 1845–1856, Dec. 1993.

[7] Y. Tosaka, H. Kanata, T. Itakura, and S. Satoh, "Simulation technologies for cosmic ray neutron-induced soft errors: Models and simulation systems," *IEEE Trans. Nucl. Sci.*, vol. 46, no. 3, pp. 774–779, Jun. 1999.

[8] J. E. Vinson, "Circuit reliability of memory cells with SEU protection (for space application)," *IEEE Trans. Nucl. Sci.*, vol. 39, no. 6, pp. 1671–1678, Dec. 1992.

[9] A. S. Brogna, F. Bigongiari, F. Bertuccelli, W. Errico, S. Giovannetti, E. Pescari, and R. Saletti, "SEU protected CPU for slow control on space vehicles," in *Proc. 2nd IEEE* Int. *Workshop* Electron. Des., *Test Appl.*, Jan. 28–30, 2004, pp. 422–424.

[10] Y. Chen, D. Nguyen, S. Guertin, J. Bernstein, M. White, R. Menke, and S. Kayali, "A reliability evaluation methodology for memory chips for space applications when sample size is small," in *Proc. IEEE Int. Integr. Rel. Workshop Final Report*, Oct. 20–23, 2003, pp. 91–94.

[11] R. D. Schrimpf and D. M. Fleetwood, *Radiation Effects and Soft Errors in Integrated Circuits and Electronic Devices*. Singapore: World Scientific, 2004.

[12] P. E. Dodd and L. L. Massengill, "Basic mechanisms and modeling of single-event upset in digital microelectronics," *IEEE Trans. Nucl. Sci.*, vol. 50, no. 3, pp. 583–602, Jun. 2003.

[13] M. Nicolaidis, "Design for soft error mitigation," *IEEE Trans. Device Mater. Rel.*, vol. 5, no. 3, pp. 405–418, Sep. 2005.

[14] T. C. May and M. H. Wood, "A new physical mechanism for soft errors in dynamic memories," in *Proc. 16th Annu. Int. Rel. Phys. Symp.*, 1978, pp. 33–40.

[15] A. M. Saleh, J. J. Serrano, and J. H. Patel, "Reliability of scrubbing recovery-techniques for memory systems," *IEEE Trans. Rel.*, vol. 39, no. 1, pp. 114–122, Apr. 1990.

[16] G. C. Yang, "Reliability of semiconductor RAMs with soft-error scrubbing techniques," *Proc. Inst. Elect. Eng.—Comput. Digit. Tech.*, vol. 142, no. 5, pp. 337–344, Sep. 1995.

[17] M. A. Bajura, Y. Boulghassoul, R. Naseer, S. DasGupta, A. F. Witulski, J. Sondeen, S. D. Stansberry, J. Draper, L. W. Massengill, and J. N. Damoulakis, "Models and algorithmic limits for an ECC-based approach to hardening sub-100-nm SRAMs," *IEEE Trans. Nucl. Sci.*, vol. 54, no. 4, pt. 2, pp. 935–945, Aug. 2007.

[18] C. L. Chen and M. Y. Hsiao, "Error-correcting codes for semiconductor memory applications: A state-of-the-art review," *IBM J. Res. Develop.*, vol. 28, no. 2, pp. 124–134, 1984.

[19] S. Lin and D. Costello, *Error Control Coding*, 2nd ed. Englewood Cliffs, NJ: Prentice–Hall, 2004.

[20] R. M. Goodman and M. Sayano, "The reliability of semiconductor RAM memories with on-chip error-correction coding," *IEEE Trans. Inf. Theory*, vol. 37, no. 3, pp. 884–896, May 1991.

[21] M. Blaum, R. Goodman, and R. McEliece, "The reliability of single-error protected computer memories," *IEEE Trans. Comput.*, vol. 37, no. 1, pp. 114–119, Jan. 1988.

[22] R. M. F. Goodman and R. J. McEliece, "Hamming codes, computer memories, and the birthday surprise," in *Proc. 20th Allerton Conf. Commun., Control Comput.*, 1982, pp. 672–679.

[23] W. Feller, *An Introduction to Probability Theory and Its Applications*. New York: Wiley, 1968.

[24] G. Yang and T. Fuja, "The reliability of systems with two levels of fault tolerance: The return of the "birthday surprise"," *IEEE Trans. Comput.*, vol. 41, no. 11, pp. 1490–1496, Nov. 1992.

[25] M. Dwass, "More birthday surprises," *J. Comb. Theory*, vol. 7, pp. 258–261, 1969.

[26] M. S. Klamkin and D. J. Newman, "Extensions to the birthday surprise," *J. Comb. Theory*, vol. 3, pp. 279–282, 1967.

[27] L. Holst, "On birthday, collectors', occupancy and other classical urn problems," *Int. Stat. Rev.*, vol. 54, no. 1, pp. 15–27, 1986.

Juan Antonio Maestro (M'07) received the M.Sc. degree in physics and the Ph.D. degree in computer science from the Universidad Complutense de Madrid, Madrid, Spain, in 1994 and 1999, respectively.

He has served both as a Lecturer and a Researcher at several universities, such as the Universidad Complutense de Madrid; the Universidad Nacional de Educación a Distancia (Open University), Madrid; Saint Louis University, Madrid; and the Universidad Antonio de Nebrija, Madrid, where he currently manages the Computer Architecture and Technology Group. His current activities are oriented to the space field, with several projects on reliability and radiation protection, as well as collaborations with the European Space Agency. He is the author of numerous technical publications, both in journals and international conferences. Aside from this, he has worked for several multinational companies, managing projects as a Project Management Professional and organizing support departments. His areas of interest include high level synthesis and cosynthesis, signal processing and real-time systems, fault tolerance, and reliability.

Pedro Reviriego (A'03–M'04) received the M.Sc. and Ph.D. degrees (with honors) in telecommunications engineering from the Technical University of Madrid, Madrid, Spain, in 1994 and 1997, respectively.

From 1997 to 2000, he was an R&D Engineer with Teldat, Madrid, working on router implementation. In 2000, he joined Massana to work on the development of 1000BaseT transceivers. During 2003, he was a Visiting Professor at the University Carlos III, Leganés, Spain. From 2004 to 2007, he was a Distinguished Member of the technical staff with LSI Corporation, working on the development of Ethernet transceivers. He is currently with the Universidad Antonio de Nebrija, Madrid. His research interests are fault tolerant systems, performance evaluation of communication networks, and the design of physical layer communication devices. He has authored numerous papers in international conferences and journals. He has also participated in the IEEE 802.3 standardization for 10GBaseT.