Conf. Proc. IEE: Practical Applications of Channel Coding Techniques,
London, U.K., 5pp, February 1983

CHANNEL CODES AND COMPUTER MEMORIES

R.M.F. Goodman B.Sc.,Ph.D., and P.J.C. Clare B.Sc.
Dept of Electronic Eng.
University of Hull
HULL HU6 7RX

Summary
This paper considers the application of channel coding techniques to VLSI computer memories. Techniques for both error detection and error correction are considered, at both the chip and the system level. In addition to simple Hamming code schemes we consider the use of spares switching and on-chip redundancy. Finally we present lifetime results for these coded systems, obtained by both theoretical analysis and computer simulation.

Introduction

All modern computer memories are built from VLSI RAM chips, and memory chips are appearing in larger and larger arrays in many end user products. Individually, RAM chips have a high reliability of the order of 10-7 failures/hour, and the semiconductor manufacturers are constantly striving to improve these rates. As larger memory chips appear, however, the "learning curve" dictates that reliability decreases before improving. More importantly, when many chips are combined to form a large memory array, the overall system reliability may be unsatisfactory. For these reasons, computer memories are protected from chip failures by some form of error-correction coding (ECC).

The objective of ECC is to increase system reliability without increasing system complexity unacceptably. We may assess reliability improvement in several ways. Firstly, the coded system will have an improved mean time between failures (MTBF) over that of an uncoded memory. We call the ratio of the coded MTBF to the uncoded MTBF the coding gain (CG). Alternatively, if we are concerned with the mass-production of a memory product, we may want to know the probability of system failure at certain time intervals, such as a week or a year, as this will predict the number of service calls for system maintenance we get, at these times! The cost of ECC is added complexity. This can take several forms depending on the coding method used. Typically, the coded system may require more (redundant) memory chips, or the memory access time will be lengthened. Also, if correction is achieved by software, the processor will have to be interrupted, and this may not be acceptable. As expected there is a trade-off between the coding gain we get and the dollars we have to spend.

In the paper we look at several coding techniques and present the results of some of our work on assessing coding gains and system performance.

Chip and System Failure

Figure 1 shows a simplified diagram of a typical 4116 type 16K by 1 bit MOS dynamic RAM. The memory is organised in two blocks separated by the column sense amplifiers. A particular cell is selected by activating one of the 128 row lines and reading from one of the 128 column decoder lines. The 7 bit row and column addresses are multiplexed into the chip via the same pins and latched into the appropriate decoder. Notice that the memory appears to the user as 16K words of one bit each. Although wider memories exist (4,8 bits), maximum packing on silicon can only be achieved in this way. In addition, the user can build a memory of any data width. This "by-one" structure determines to a large extent the type of coding that can be used.

The types of failure that occur in such a chip are either "hard" or "soft". Soft errors are single-cell errors that are caused by random noise or circuit stress effects such as temperature, or they can be caused by alpha particles changing the cell charge. This radiation effect is inherent in the packaging of the chip, and these soft errors account for the majority (70%) of errors. Soft errors can be removed by simply re-writing the correct value into the cell. Although they are more prominent we shall see that in a system with ECC it is the hard errors that dominate system reliability. Hard failures are permenent defects that result in one or more cells becoming "stuck-at" a particular value on read. We can classify these faults as single cell, row failure, column failure, combined row-column, half chip, and full chip. Typical ratios are 50% single, 20% row or column, 10% combined and less than 0.1% others (Ref.1). Figure 2 shows the effects of these failures. The effect of these failures on the users memory map depends on how the address bits are mapped onto to rows and columns. If sequential addressing is assumed, that is low order onto rows, then the effect of a row failure would be to cause 256 bit errors spaced 256 bits apart in the map. A column failure would cause a contiguous group of 256 errors. Other bit mappings will cause "bursts" of errors within the map. An important difference between memory and channel coding is due to the fact that errors are usually stucks, that is, they are stationary in time. This can be used to advantage in the coding method.

Figure 3 shows the organisation of a typical memory system made out of by 1 RAMs. The width of the data word is k and there are M rows of chips. If coding is applied, then redundant columns of chips are added to hold the parity checks thus increasing the width to n. The code rate is therefor k/n and the redundancy $1-(k/n)$. If spares are available these take the form of extra rows. To see the effect of chip failures on an uncoded memory, consider the case of a 1 megaword memory consisting of $M=4$ rows of 64K RAM chips, each row containing $k=32$ data chips. There are $D=128$ chips in the system which will fail if any error hard or soft occurs. The reliability of the system is thus dominated by the soft failures. If the soft failure rate is $f=10E-5$ per device then the $MTBF = 1/f.k.M = 781$ hours, or approximately one month. Expressed another way, we can define the reliability (R) of a device as the probability of correct operation at time t. Thus $R=\exp(-ft)$, assuming constant failure rates (ref.2), and the probability of system failure at time t is $(1-R^D)$. Applying this to the previous system, this gives a probability of system failure at the 48 hour point of 6%. Alternatively, we can say that 6% of our products will only last 48 hours. This is clearly not acceptable!

Error Correction Techniques

The first question to be considered in choosing an ECC system is the trade-off between hardware and software correction. If the detection of errors only is to be performed by hardware then the memory will operate as fast with ECC as without. This is because the ECC logic need not be placed in series with the data path but rather in parallel with the data bus. In such a configuration the ECC hardware monitors all memory accesses. If an error is detected on memory read, then the processor is interrupted and must perform a suitable correction routine. This routine would require the address of the error and the syndrome of the error pattern (which uniquely specifies the error pattern). ECC chips such as the Zilog Z8160 provide latched syndrome data readable by the processor, but not address latches which would have to be external. If it is not acceptable to interrupt the processor then transparant error correction must be used. In this case the decoder must be inserted in series with the data path. If access times are not to suffer then the correction logic must operate with a minimum of delay, and this is the reason why simple codes are used in memory coding when compared with channel coding. Complex codes simply take too long to decode. In practice a mixed hardware/software approch is usually adopted with simple error correction hardware being backed up by more powerful routines in software.

Most ECC systems on computer memories are based on the modified single error correcting (SEC), double error detecting (DED) Hamming code (refs 1,2). These codes are modified because for easier hardware generation of the check bits, each check bit operates on equal numbers of data bits. The cost of using such a code is a degradation in access time of about 10%, and a redundancy of 30% for a k=16 data word, 20% for k=32. The code can be used as a basis for more complex schemes such as erasure decoding or spares switching, outlined below. If implemented in hardware, then single errors in a word are corrected transparently but double errors are only detected and require processor interrupt.

Other coding schemes have also been proposed but not used in any wide sense, due to their inherent complexity. These include the use of multiple error correcting BCH codes (ref.3), convolutional codes (ref.4),Reed-Solomon codes for byte-wise memories (ref.5), and low-redundancy low-power vertical / horizontal parity.

Erasure decoding can be used to correct multiple errors in a word. On detecting the first error the processor is interrupted, and determines if the error is hard or soft. Also the error is automatically corrected. If the error is hard the decoder stores the position of the error for future use. When the second error occurs the Hamming code "erases" that is, it sees a double error but cannot correct it. By using the stored position of the known error, however, the processor can compute the position of the new error and correct it. The erasure method can be extended to multiple correction because the correction power of a code is bounded by 2t+r<d where t is the error correction power, e is the erasure power, and d is the Hamming distance of the code.

Spares switching can be used to extend the lifetime of the memory. With this technique the processor is interrupted and determines if the error is hard or soft. If it is hard a spare chip is reconfigured into the address space of the failed device, and the corrected data is written into it. Spares can be swapped in in rows or individually. The latter presents address decoding problems, although it is more efficient in memory.

The Performance of Coded Memories

In our previous papers (refs.10,11) we have analysed the performance of coded memories both analytically and by computer simulation. In particular, we have considered an arbitrary r - error correcting code applied to each chip row, and spares switching. By modelling the failure process as a random Poisson process and assuming independent catastrophic chip failures we can show that the coding gain CG= R.(1 + 1 + (M-1)/M + (M-1)(M-2)/M^2 +...) for an SEC code such as the Hamming code , where R is the code rate k/n. For many rows this is well approximated by CG=R.(2/3 + M/2) , showing that the coding gain is quite modest unless M is large.

We have extended our analysis to the case where we take account of the actual failure modes inside the chip. It can be seen that in a coded memory a particular word will fail if two errors occur in it. This is very unlikely to occur due to a single hard or soft error. Thus although these errors are the most frequent, the implication is that most memory system failures will be caused by a row failure in one chip and a column failure in another chip in the same row of chips. In particular, for equiprobable row and column failures we can show that CG =   M + 1 . By way of example consider the memory mentioned earlier. Let the chip failure rate be f = 1E-5 per hour with 99% being single cell, and the other 1% being equally divided between row and column failures. If we assume single cell errors then there are effectively 64K rows and the formula in the precceding paragraph gives an MTBF of 411k hours i.e. 48 years. However, using M=4 and f = 1E-7 for the row/column case we get an MTBF of 298k hours i.e. 34 years. This shows that with this type of distribution of single versus row/column errors, it is the latter that determine the system performance. Indeed we can conclude that error scrubbing is useless, as the hard row/column error will get you first.

Acknowledgements
The authors would like to acknowledge financial support fron the UK SERC, the Joint Services Electronic Program (USA), and NATO Scientific Affairs.

References
1. Intel Corp. "Memory System Reliability with ECC", Intel Memory Applications Handbook", 1980
2. Goodman, R.M.F., "Error Correction Coding for VLSI Memories" Conf. New LSI Techniques, IEE Oct 1980
3. Levine, L. and Meyers, W. "Semiconductor Memory Reliability with Error Detecting and Correcting Codes ",Computer, Oct 1976
4. Metzner, J.J. "Convolutionally Encoded Memory Protection", IEEE Trans. Computers, Vol C31,No 7, June 1982
5. Kaneda, S. and Fujiwara, E. "Single Byte Error Correcting Double Byte Error Detecting Codes for Memory Systems",IEEE Trans. Computers,Vol C31, No 7, June 1982
6. Edwards, L. "Low Cost Alternative to Hamming Codes Corrects Memory Errors",Computer Design, July 1981
7. Sundberg, C.E. "Erasures and Error Decoding for Semiconductor Memories",IEEE Trans. Computers,Vol C27,No 8, Aug 1978
8. Walker, W.K.S., Sundberg, C.E., and Black, C.J. "A Reliable Spacebourne Memory with a Single Error and Erasure Correction Scheme",IEEE Trans. Computers,Vol C28,No 7,July 1979
9. Carter, W.C. and McCarthy, C.E. "Implementation of an Experimental Fault Tolerent Memory System",IEEE Trans. Computers, Vol C25,No 6,June 1976
10. Goodman, R.M.F. and McEliece, R.J. "Lifetime Analyses of Error Control Coded Semiconductor Ram systems", Proc.IEE,Vol 129,Pt.E,No 3, May 1982
11. Goodman, R.M.F. and McEliece, R.J. "Hamming Codes, Computer Memories, and the Birthday Surprise", 4th Allerton Conference on Computing and Control, USA, Sept 1982

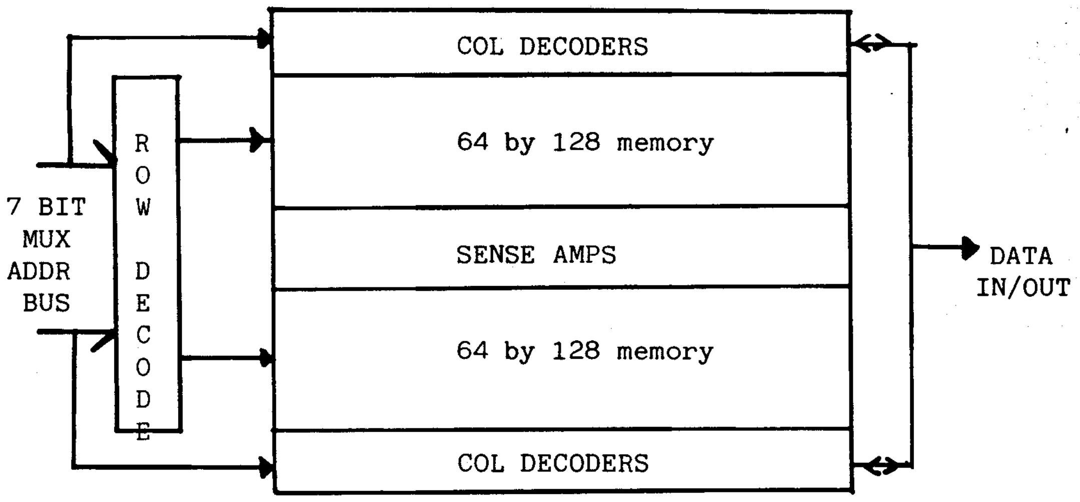

Figure 1. RAM chip

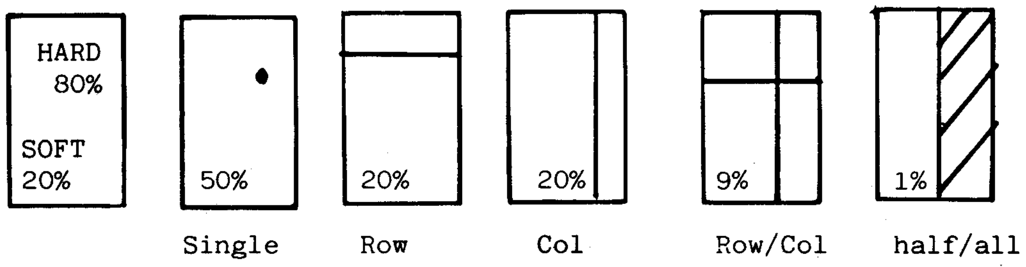

Figure 2. Types of failure

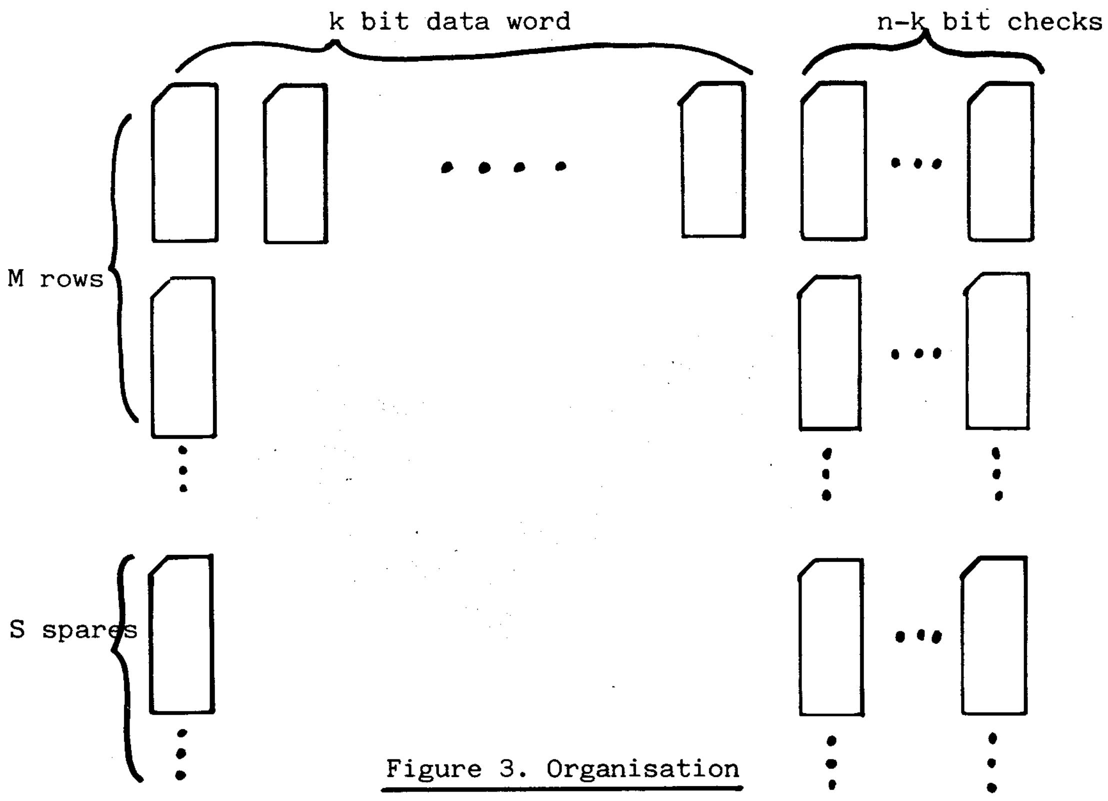

Figure 3. Organisation