Design of error-control coding schemes for three problems of noisy information transmission, storage and processing

van Gils, W.J.

DOI:
10.6100/IR274904

Published: 01/01/1988

Document Version
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

DESIGN OF ERROR-CONTROL CODING SCHEMES FOR THREE PROBLEMS OF NOISY INFORMATION TRANSMISSION, STORAGE AND PROCESSING

Wil J. van Gils

DESIGN OF ERROR-CONTROL CODING SCHEMES FOR THREE PROBLEMS OF NOISY INFORMATION TRANSMISSION, STORAGE AND PROCESSING

Wil J. van Gils

DESIGN OF ERROR-CONTROL CODING SCHEMES FOR THREE PROBLEMS OF NOISY INFORMATION TRANSMISSION, STORAGE AND PROCESSING

Proefschrift

ter verkrijging van de graad van doctor aan de Technische Universiteit Eindhoven, op gezag van de rector magnificus, prof. dr. F.N. Hooge, voor een commissie aangewezen door het college van dekanen in het openbaar te verdedigen op dinsdag 5 januari 1988 te 16.00 uur

door

Willibrordus Johannes van Gils

geboren te Tilburg

This thesis was approved by the promotors
prof. dr. J.H. van Lint and prof. dr. ir. H.C.A. van Tilborg

SUMMARY

This thesis deals with the design of error-control coding schemes for three different problems of noisy information transmission, storage and processing. These problems have in common that they are of interest from a practical, industrial point of view and that they cannot be solved elegantly by traditional error-control coding schemes.

Problem one is concerned with the transmission and storage of messages in which different parts are of mutually different importance. So it is natural to give parts of mutually different importance different protection against errors. This can be done by using different coding schemes for the different parts, but more elegantly by using a single so-called Unequal Error Protection coding scheme.

The second coding scheme is designed to be used as an automatically readable product identification code in an automated manufacturing environment. The identification number (and possibly other useful information) of a product is encoded into a square matrix of round dots on a contrasting background. Problems to be dealt with in practice are the rotations of dot matrices and the corruption of dots due to printing imperfections, dust particles and reading failures. To this end source codes and so-called square-cyclic channel codes have been designed.

The third part of this thesis describes an approach towards error-control coding for systems in which digit as well as symbol errors can occur, where a symbol is a position-fixed group of digits. Examples of such systems are computer systems and compound channels. We give the detailed design of the codes and the decoders for three particular applications. These are a generalized Triple Modular Redundant fault-tolerant computer, a memory array composed of three 9-bit wide units for storage of 16-bit words, and a '(4,2) concept' fault-tolerant computer. Finally some general theory on these so-called combined Symbol and Digit Error-Control codes is developed.

PREFACE

As already suggested by the title, this thesis is not monolithical. Apart from an introduction (Chapter 0), it consists of three chapters, each of which is subdivided into one or more sections. These sections were written at intervals and either appeared in journals, were scheduled to appear or were submitted for publication. The sections are therefore self-contained and can be read independently of one another. Co-author of sections 3.3 and 3.4 is J.P. Boly. The research work for these papers was done in strong co-operation, but for both papers it holds that the main part of the work was done by the first author.

The author is greatly indebted to the management of the Philips Research Laboratories, Eindhoven, The Netherlands, for the opportunity to carry out and to publish the work described here. Stimulating discussions with Professor J.H. van Lint, Professor H.C.A. van Tilborg, C.P.M.J. Baggen, G.F.M. Beenker, C.J.L. van Driel, L.M.H.E. Driessen, Professor J.-M. Goethals, T. Krol and L.M.G.M. Tolhuizen have greatly contributed to the contents of this thesis. Special thanks are due to J.-P. Boly for the fine co-operation on the subject of combined Symbol and Digit Error-Control codes.

CONTENTS

0. Introductory chapter

This introductory chapter gives the motivation for the research work reported in this thesis. It also provides some basic concepts of coding theory necessary for understanding the results. For an extensive treatment of the theory of error-correcting codes the reader is referred to the books of Blahut [1], van Lint [10] and MacWilliams and Sloane [11].

Coding theory preliminaries

In data transmission, storage and processing systems a desired level of error control can be guaranteed by using error-correcting codes. A *linear* $[n,k]$ *block code* C of *length* n and *dimension* k $(k < n)$ over the *alphabet* $GF(q)$, the Galois field containing q elements, is a k-dimensional subspace of the n-dimensional vector space $GF(q)^n$. A (linear) *encoding* of the *message set* $M := GF(q)^k$ is a linear mapping from M onto the code space C; a *message* $\underline{m} = (m_1, m_2, \ldots, m_k) \in M$ is mapped onto the *codeword* $\underline{c} = \underline{m}G$, where G denotes a k-by-n matrix over $GF(q)$ whose rows generate C. The matrix G is called a *generator matrix* of the code C and the fraction $R := k/n$ is called the *(information) rate* of the code. If a generator matrix G contains the k-by-k identity matrix I as a submatrix, then G is called *systematic.* In that situation the message is a part of the corresponding codeword.

If a message $\underline{m} \in GF(q)^k$ has to be transmitted (respectively stored or processed), then one does not transmit (respectively store or process) the message, but the codeword attached to it. During transmission (respectively storage or processing) the codeword can be corrupted. The nature of the corruption depends on the specific channel used. In this thesis we will assume that the channel is (or is very close to) a q-ary symmetric channel. For a q-ary symmetric channel the probability that an arbitrary, transmitted symbol from $GF(q)$ will be received as an arbitrary, different symbol from $GF(q)$ is constant, say ε. The corrupted version $\underline{r}$ of a codeword $\underline{c}$ can be seen as the addition of an error pattern

(*additive noise*) $\underline{e} \in GF(q)^n$ to $\underline{c}$: $\underline{r} = \underline{c} + \underline{e}$, where the probability that $\underline{e}$ occurs is independent of the codeword $\underline{c}$ sent.

It is the task of the *decoder*, which is at the receiving end of the channel, to estimate the original message $\underline{m}$ as good as possible from the corrupted version $\underline{r} = \underline{c} + \underline{e}$ of the codeword $\underline{c} = \underline{m}G$. The decoder's strategy is to choose the most likely codeword $\hat{\underline{c}}$, given that $\underline{r}$ was received. Provided the codewords are all equally likely, this strategy is optimal in the sense that it minimizes the probability of the decoder making a mistake. It is called *maximum likelihood decoding.*

To describe the maximum likelihood decoder more precisely we need the definitions of (Hamming) weight and distance. For a vector $\underline{x}$ in $GF(q)^n$ the *(Hamming) weight* $wt(\underline{x})$ is defined as the number of nonzero components in $\underline{x}$:

$$wt(\underline{x}) := |\{i \ : \ x_i \neq 0 \ , \ i = 1, \ldots, n\}|.$$

For two vectors $\underline{x}$ and $\underline{y}$ in $GF(q)^n$, the *(Hamming) distance* $d(\underline{x}, \underline{y})$ between $\underline{x}$ and $\underline{y}$ is defined as the number of positions in which $\underline{x}$ and $\underline{y}$ differ:

$$d(\underline{x}, \underline{y}) := |\{i \ : \ x_i \neq y_i \ , \ i = 1, \ldots, n\}|.$$

Hence, for $\underline{x}, \underline{y}$ in $GF(q)^n$ we have

$$d(\underline{x}, \underline{y}) = wt(\underline{x} - \underline{y}).$$

We assume that the channel error rate ε is smaller than $1/q$. Then, to minimize the probability of making a miscorrection, the decoder decodes a received vector $\underline{r}$ as a nearest (in Hamming distance sense) codeword $\hat{\underline{c}}$, i.e., it picks an error vector $\hat{\underline{e}}$ which has smallest weight:

$$d(\underline{r}, \hat{\underline{c}}) = \text{minimum}\{d(\underline{r}, \underline{c}) : \underline{c} \in C\},$$

or equivalently

$$wt(\hat{\underline{e}}) = \text{minimum}\{wt(\underline{e}) : \underline{r} - \underline{e} \in C\}.$$

This procedure is called *(complete) nearest neighbour decoding.* It is equivalent to maximum likelihood decoding if $\varepsilon < 1/q$. In practice such a complete decoding strategy would be too complex

for implementation. Therefore an incomplete, so-called *bounded distance* decoder is used, which only corrects the error patterns of weight at most some fixed value t. To determine this value t, we need the definition of the minimum (Hamming) distance of a code. For a linear code C the *minimum (Hamming) distance* is defined as the minimum distance between two different codewords of C,

$$d := \text{minimum}\{d(\underline{x}, \underline{y}) : \underline{x}, \underline{y} \in C, \underline{x} \neq \underline{y}\}.$$

Because C is linear, the minimum Hamming distance of C is equal to the minimum Hamming weight of C,

$$d = \text{minimum}\{wt(\underline{x}) : \underline{x} \in C, \underline{x} \neq \underline{0}\},$$

where $\underline{0}$ denotes the allzero vector of length n. When the minimum (Hamming) distance of a code C equals d, then all error patterns of weight at most some fixed value t can be corrected if and only if $t \leq \lfloor (d-1)/2 \rfloor$, where $\lfloor x \rfloor$ denotes the largest integer less than or equal to x. The code is called *t-error-correcting*. All received words that are outside the spheres with radius t around codewords can be detected to be in error. Because all error patterns of weight at least $t+1$ and at most $d-(t+1)$ can be detected, the code C is called *$(d-t-1)$-error-detecting*. In practice, it is not feasible to compare a received word to all codewords to determine which is closest. To overcome this problem we introduce a so-called syndrome decoder.

An $(n-k)$-by-n matrix H over $GF(q)$ is called a parity-check matrix of the linear code C if

$$\underline{c}H^T = \underline{0} \Longleftrightarrow \underline{c} \in C.$$

For a vector $\underline{x} \in GF(q)^n$,

$$\underline{s} := \underline{x}H^T$$

is called the *syndrome* of $\underline{x}$. So for all codewords $\underline{c}$ in C the syndrome equals $\underline{0}$. It is well-known that for all error patterns $\underline{e}$ of weight at most $\lfloor (d-1)/2 \rfloor$ the syndromes are mutually different. Hence these syndromes can be used in the decoder. Two elements in $GF(q)^n$ are in the same coset of C if and only if they have

identical syndromes. For all cosets of C we determine a minimum weight element contained in it and call it a coset leader of that coset. Note that each of the cosets $\underline{x}+C, wt(\underline{x}) \leq \lfloor (d-1)/2 \rfloor$ has a unique coset leader. The coset leaders are used in the *syndrome decoder* which works as follows:

1. whenever $\underline{r}$ is received, the syndrome $\underline{s} := \underline{r}H^T$ is computed;
2. the coset leader $\underline{l}$ of the coset with syndrome $\underline{s}$ is taken as the estimate for the error pattern;
3. the estimate for the codeword is $\hat{\underline{c}} := \underline{r} - \underline{l}$.

In the incomplete syndrome decoder only the syndromes of cosets with a coset leader of weight at most some fixed value $t, t \leq \lfloor (d-1)/2 \rfloor$, are used for error-correction, the other syndromes being used for error-detection. Step 2 of the syndrome decoder can be implemented as a list of syndrome, coset leader pairs. If this list becomes too long, other implementations of step 2 are needed. For example, for codes defined in an algebraic way, e.g. BCH and Reed-Solomon codes, this can be implemented by more sophisticated (algebraic) algorithms.

We say that a received word contains an *error* if it is not a codeword and we do neither know the position nor the value of the corruption. We say that a received word contains an *erasure* if it is not a codeword and we know the position, but not the value of the corruption. Of course, combinations of erasures and errors can also occur. A linear $[n,k,d]$ code of length n, dimension k, and minimum distance d can correct e erasures and t errors if $e+2t \leq d-1$ [1,Ch.9,Sec.2].

In coding theory, it is very popular to construct new codes from old ones. The most trivial way to do this is by adding to any codeword $(c_1, \ldots, c_n)$ one symbol, namely its *overall parity-check*

$$c_{n+1} := -\sum_{i=1}^{n} c_i.$$

Other minor changes to codes are called *extending, puncturing, expurgating, augmenting, lengthening* and *shortening*, whose definitions can be found in [11,Ch.1,Sec.9]. More complex ways of

constructing new codes consist in combining several codes into new ones. This is done to construct good codes and for ease of decoding. These methods can be found in [11,Ch.18].

For ease of encoding and decoding so-called cyclic codes [11, Chs.3, 4, 7, 8], [10, Ch.6] are often used. A linear code is called cyclic if the cyclic shifts of the codewords of the code again yield codewords of the code. The most famous classes are the BCH codes [11,Ch.9], [10,Ch.6] and the Reed-Solomon (RS) codes [11, Ch.10], [10, Ch.6].

Motivation

The investigations reported in this thesis were initiated by the following considerations. In practical situations, the size q of the alphabet is a power of 2: $q = 2^m, m \geq 1$. If $m > 1$, then a symbol from $GF(2^m)$ is built up from m binary digits (bits). In the past four decades a lot of linear coding schemes have been constructed with the following three assumptions:

- all q-ary message digits are equally important,
- codewords are only corrupted by additive noise,
- either binary digit error-control or 2^m-ary symbol error-control is required, in other words the channel is supposed to be a binary symmetric channel or a 2^m-ary symmetric channel for some $m > 1$.

Chapters 1,2, and 3 deal with three practical situations in which in each of them exactly one of the above assumptions is not fulfilled. These three situations demand three different coding schemes which have the following three respective properties:

- different parts of a message should get different protection against errors because they are of mutually different importance,
- 'rotations' (that are certain permutations of the symbols) of codewords during transmission, in addition to corruption by additive noise up to a certain level, should not cause miscorrections by the decoder,

- both bit and (m-bit) symbol errors should be coped with because the behaviour of the channel is a combination of that of a binary symmetric channel and that of a 2^m-ary symmetric channel.

Those three coding schemes are briefly discussed below.

1. Unequal Error Protection

Most error-correcting codes considered in the literature have the property that their correcting capabilities are described in terms of the correct reception of the entire message. These codes can successfully be applied in those cases where all positions in a message word require equal protection against errors.

However, many applications exist in which some message positions are more important than others. For example in transmitting numerical data, errors in the sign or high-order digits are more serious than are errors in the low-order digits. As another example consider the transmission of message words from different sources simultaneously in only one codeword, where the different sources have mutually different demands concerning the protection against errors. Linear codes that protect some positions in a message word against a larger number of errors than other ones are called *Linear Unequal Error Protection (LUEP) codes.* Masnick and Wolf [12] introduced the concept of unequal error protection (UEP). But, in contrast with what one would expect, they considered error protection of single positions in codewords. In Chapter 1 we consider error protection of single positions in the input message words, following the formal definitions of Dunning and Robbins [6]. They introduced the so-called *separation vector* to measure the error-correcting capability of an LUEP code. Whenever a k-dimensional LUEP code over $GF(q)$ with separation vector $\underline{s} = (s_1, s_2, \ldots, s_k)$ is used on a q-ary symmetric channel, complete nearest neighbour decoding guarantees the correct interpretation of the ith message digit if no more than $(s_i - 1)/2$ errors have occurred in the transmitted codeword.

Chapter 1 deals with LUEP codes. A basic problem is to find an LUEP code with a given dimension and separation vector such

that its length is minimal and hence its information rate is maximal. In Section 1.1 we derive a number of bounds on the length of LUEP codes. For the special case where all message positions are equally protected, some of our bounds reduce to the well-known Singleton, Plotkin and Griesmer bounds. Section 1.1 provides a table containing the parameters of all binary LUEP codes with maximal separation vector and length less than or equal to 15. The construction of these codes is given in the Appendix of Chapter 1. The second part of Section 1.1 deals with cyclic UEP codes. It gives a table of all binary cyclic UEP codes of length at most 39 and it provides classes of binary cyclic UEP codes that are majority logic decodable. Majority logic decoding means that the decoder estimates a message bit by taking the majority vote over a number of votes generated from the received word. In Section 1.2, methods for combining codes, such as the direct sum, direct product, and $|u|u+v|$ construction, concatenation, etc., are extended to LUEP codes.

Section 1.1 is a reprint from IEEE Transactions on Information Theory, vol. IT-29, no. 6, pp. 866-876, Nov. 1983, except for the tables which have been updated. Section 1.2 was published in the same journal, vol. IT-30, no. 3, pp. 544-546, May 1984. The constructions in the Appendix appeared in Philips Journal of Research, vol. 39, no.6, pp. 293-304, 1984. Finally, we like to refer to Driessen et al. [4], who describe the application that stimulated the research in Unequal Error Protection.

2. Two-dimensional square-cyclic dot codes

The widespread use of bar codes in automated manufacturing clearly shows the need for an automatically readable product identification code. A bar code is built up from a number of parallel bars. The relative widths and mutual distances of these bars determine the meaning of the bar code.

We believe, however, that *dot codes* provide a better alternative to bar codes in this area of technology. A dot code consists of a square matrix of round dots on a contrasting background. The meaning of the dot code is determined by the absence or presence

of dots. In a dot code the information is recorded in two dimensions, whereas in a bar code only one direction is used to encode information. This difference enables the dot code to offer higher information density, thereby allowing smaller product identification areas. For example, at the flat top of an electric motor shaft there is not enough room for a bar code. Furthermore, in automated manufacturing it is easy to write the dot codes onto the mechanical parts by an engraving process. With bar codes this would be more complicated. The dot codes can be read by a standard TV camera and can be recognized by a relatively inexpensive picture processing system.

We shall therefore introduce a method for the transmission of numbers from one point to another point by means of square matrices of round dots. These square dot matrices can be translated into square binary matrices by representing the presence of a dot by a one (1) and the absence of a dot by a zero (0). In a practical situation it was observed that only random dot corruptions (causing random bit errors) occurred in the dot squares. These errors were due to printing imperfections, dust particles, and reading failures. Furthermore, because of the possibly random rotation of the mechanical parts during the manufacturing process, decoding of the dot matrices should be possible irrespective of the orientation of the matrices. For example, one should again think of a square dot matrix on the flat top of a rotated shaft of an electric motor, without any synchronization indication outside the dot matrix.

Chapter 2 describes a possible solution to this transmission problem, where we have to deal with random corruptions but also with 'rotations' of codewords. The solution is split into a source coding scheme and a channel coding scheme. In the source coding scheme product identification numbers are transformed into channel message words. The channel coding scheme encodes the channel message words into channel codewords, which are transmitted as square dot matrices. The source code depends on the channel code and is such that the four rotations of a dot matrix are all decoded into the same product identification number. We describe two source coding schemes. One is the optimal one, in the sense that it uses the minimum number of bits to encode a

product identification number into a channel message word. The other scheme is not optimal, but gives rise to a very simple and fast encoding/decoding algorithm. The channel coding scheme uses so-called *square-cyclic* codes. In a square-cyclic code, the rotation of a codeword (as a dot matrix) again gives a codeword of the code. We construct square-cyclic codes from well-known quasi-cyclic and (shortened) cyclic codes.

This research was stimulated by the application of dot codes in product identification schemes as described in [2] and [13]. Chapter 2 appeared in the IEEE Transactions on Information Theory, vol. IT-33, no. 5, September 1987.

3. Combined symbol and digit error-control

Up to now coding experts have spent a great deal of effort constructing binary codes that can correct random bit errors, such as, for example, BCH codes. A lot of research into codes over larger alphabets, such as the Reed-Solomon (RS) codes, has also been done. These RS codes are able to correct symbol errors, a symbol being a position-fixed group of (binary) digits. In many applications, however, one encounters situations where both types of errors, i.e., random bit and random symbol errors, occur. For example, this is the case in computers, memory arrays and compound channels. Up to now substantial effort has only been put into designing codes that can detect single symbol errors, in addition to their single-bit error-correcting and double-bit error-detecting capabilities [3,5,7,8,14,15]. These codes were designed for memory systems composed of m-bit wide chips, where m is larger than one. In such architectures a chip failure causes a random (m-bit) symbol error, which has to be detected. Single bit errors caused by the failure of single memory cells are corrected, double bit errors are detected. The need for a wider class of codes that are able to detect and correct digit errors and erasures and symbol errors and erasures was first recognized by Krol in his design of the '(4,2) concept' fault-tolerant computer [9].

Chapter 3 deals with the design of these so-called *combined Symbol and Digit Error-Control (SDEC) codes.* The first three

sections give the design of SDEC codes for three particular applications and describe their implementations.

Section 3.1 describes a so-called generalized Triple Modular Redundant fault-tolerant computer design. In the Triple Modular Redundancy (TMR) concept, computer hardware is triplicated and majority voting is applied to improve the overall system availability and reliability. Seen from the point of view of coding theory, the TMR technique is a realization of a [3,1] repetition code. The question posed by us was how to construct [3,1] codes that cannot only correct symbol errors, caused by the failure of one of the three identical parts in the system, but also multiple bit errors caused by the memories. These codes would save the use of bit-error-correcting codes for the memories. In Section 3.1, [3,1] codes over $GF(2^m)$, $m = 4, 8, 16$ are constructed, their error-control capacities are shown, and their decoder designs are described.

Section 3.2 describes codes for storing 16-bit words in a memory array consisting of three 9-bit wide units, a unit being a single card or a single chip. These codes are able to correct single bit errors, to detect up to four bit errors, and to detect the failure of a complete memory unit. The codes have an elegant structure which makes fast decoding possible by simple means.

In Section 3.3 the construction, properties and decoding of four nonequivalent [4,2] codes over $GF(2^8)$ are described. These codes are able to correct single (8-bit) symbol errors, to correct up to three bit errors, and to correct the combination of a symbol erasure and at most one bit error. In addition all error patterns containing one symbol erasure and two bit errors can be detected. These codes can be used in a (4,2) concept fault-tolerant computer [9] and in memory systems composed of 8-bit wide chips or cards.

Finally, Section 3.4 gives, after the 'preparing' sections, a more theoretical discussion of combined Symbol and Digit Error-Control codes. It starts with the definition of the *minimum distance profile*, a measure for the symbol and digit error-control capacities of a code. Equivalence of SDEC codes is discussed and the construction of several classes of SDEC codes is given. Furthermore, Section 3.4 contains tables of parameters of SDEC codes over alphabets of 2-,3-,4-,6- and 8-bit symbols.

Section 3.1 appeared in IEEE Transactions on Computers, vol.

C-35, no.7, pp. 623-631, July 1986. Section 3.2 was published in Philips Journal of Research, vol. 41, no. 4, pp. 391-399, 1986. Section 3.3 appeared in IEEE Transactions on Information Theory, vol. 33, no.6, November 1987. Section 3.4 has been submitted to the same journal for publication.

References

[1] R.E. Blahut, *Theory and Practice of Error Control Codes*, Reading,MA: Addison-Wesley 1983.

[2] J.W. Brands, W. Venema, "Product identification with dot codes", *Philips CAM messages*, no. 2, pp. 18-20, Jan. 1984, and *Philips CAMera*, no. 15-IDT-CA-84001, pp. 4-5, Febr. 1984.

[3] C.L. Chen, "Error-correcting codes with byte error detection capability", *IEEE Trans. on Computers*, vol. C-32, no. 7, pp. 615-621, July 1983.

[4] L.M.H.E. Driessen, W.A.L. Heijnemans, E. de Niet, J.H. Peters, A.M.A. Rijkaert, "An experimental digital video recording system", *IEEE Trans. on Consumer Electronics*, vol. CE-32, no. 3, August 1986.

[5] L.A. Dunning, "SEC-BED-DED codes for error control in byte organized memory systems", *IEEE Trans. on Computers*, vol. C-34, no. 6, pp. 557-562, June 1985.

[6] L.A. Dunning and W.E. Robbins, "Optimal encodings of linear block codes for unequal error protection", *Inform. Contr.*, vol. 37, pp. 150-177, 1978.

[7] L.A. Dunning and M.R. Varanasi, "Code constructions for error control in byte organized memory systems", *IEEE Trans. on Computers*, vol. C-32, no. 6, pp. 535-542, June 1983.

[8] S. Kaneda, "A class of odd weight column SEC-DED-SbED codes for memory systems applications", *IEEE Trans. on Computers*, vol. C-33, no. 8, pp. 737-739, August 1984.

[9] T. Krol, "(N,K) concept fault-tolerance", *IEEE Trans. on Computers*, vol. C-35, no. 4, pp. 339-349, April 1986.

[10] J.H. van Lint, *Introduction to Coding Theory*, New York: Springer 1982.

[11] F.J. MacWilliams and N.J.A. Sloane, *The Theory of Error-Correcting Codes*, Amsterdam: North-Holland 1977.

[12] B. Masnick and J. Wolf, "On linear unequal error protection codes," *IEEE Trans. on Inform. Theory*, vol. 13, pp. 600-607, Oct. 1967.

[13] *Philips Identivision System: the answer to your product identification problems*, publication Philips Nederland, department Projects Technical Automation, Eindhoven, The Netherlands.

[14] S.M. Reddy, "A class of linear codes for error control in byte per card organized digital systems", *IEEE Trans. on Computers*, vol. C-27, no. 5, pp. 455-459, May 1978.

[15] M.R. Varanasi, T.R.N. Rao and Son Pham, "Memory package error detection and correction", *IEEE Trans. on Computers*, vol. C-32, no. 9, pp. 872-874, Sept. 1983.

1.1

Two topics on linear unequal error protection codes: bounds on their length and cyclic code classes

Wil J. van Gils

Abstract

It is possible for a linear block code to provide more protection for selected positions in the input message words than is guaranteed by the minimum distance of the code. Linear codes having this property are called linear unequal error protection (LUEP) codes. Bounds on the length of a LUEP code that ensures a given unequal error protection are derived. A majority decoding method for certain classes of cyclic binary UEP codes is treated. A list of short (i.e., of length less than 16) binary LUEP codes of optimal (i.e., minimal) length and a list of all cyclic binary UEP codes of length less than 40 are included.

I. Introduction

Most error-correcting block codes considered in the literature have the property that their correcting capabilities are described in terms of the correct reception of the entire message. These codes can successfully be applied in those cases where all positions in a message word require equal protection against errors.

However, many applications exist in which some message positions are more important than other ones. For example in transmitting numerical data, errors in the sign or in the high-order digits are more serious than are errors in the low-order digits. As another example consider the transmission of message words from different sources simultaneously in only one codeword, where the different sources have different demands concerning the protection against errors.

Linear codes that protect some positions in a message word against a larger number of errors than other ones are called linear unequal error protection (LUEP) codes. Masnick and Wolf [8] introduced the concept of unequal error protection (UEP). But, in constrast with one would expect, they considered error protection of single positions in codewords. In this paper we consider error protection of single positions in the input message words, following the formal definitions of Dunning and Robbins [2]. They introduced a so-called separation vector to measure the error-correcting capability of a LUEP code. Whenever a k-dimensional LUEP code over $GF(q)$ with separation vector $\underline{s} = (s_1, s_2, \ldots, s_k)$ is used on a q-ary symmetric channel, complete nearest neighbour decoding [7, p. 11] guarantees the correct interpretation of the ith input message digit if no more than $\lfloor (s_i - 1)/2 \rfloor$ errors have occurred in the transmitted codeword.

A basic problem is to find a LUEP code with a given dimension and separation vector such that its length is minimal and hence its information rate is maximal. In Section III we derive a number of bounds on the length of LUEP codes. For the special case where all message positions are equally protected, some of our bounds reduce to the well-known Singleton, Plotkin, and Griesmer Bounds. Some earlier work on bounds was done by Katsman [6];

he derived Corollary 14 for the binary case. Our bounds give better results than the bound in [6]. Table I provides a table of binary LUEP codes with maximal separation vector and length less than or equal to 15.

In Section IV we consider classes of cyclic UEP codes that can be decoded by majority decoding methods. Earlier results on cyclic UEP codes were obtained by Dyn'kin and Togonidze [3]. Table II provides a table of all binary cyclic UEP codes of length less than or equal to 39.

II. Definitions and preliminaries

A. The separation vector

Let q be a prime power and let $GF(q)$ be the Galois field of order q. A linear $[n,k]$ code C of length n and dimension k over $GF(q)$ is a k-dimensional linear subspace of $GF(q)^n$. A generator matrix G of this code is a k-by-n matrix whose rows form a basis of C. The bijection from $GF(q)^k$ onto C that maps any element $\underline{m} \in GF(q)^k$ of the message set onto a codeword $\underline{c} = \underline{m}G$ is called an encoding of C by means of the generator matrix G. For $\underline{x} \in GF(q)^n$, $wt(\underline{x})$ denotes the (Hamming) weight of $\underline{x}$, i.e.,the number of nonzero components in $\underline{x}$.

Dunning and Robbins [2] have introduced the following formal definition.

Definition 1. For a linear $[n,k]$ code C over the alphabet $GF(q)$ the *separation vector* $\underline{s}(G) = (s(G)_1, \ldots, s(G)_k)$ of length k, with respect to a generator matrix G of C, is defined by

$$s(G)_i := \min \{wt(\underline{m}G) : \underline{m} \in GF(q)^k, m_i \neq 0\}, \quad i = 1, \ldots, k. \tag{1}$$

This means that for any $\alpha, \beta \in GF(q), \alpha \neq \beta$, the sets $\{\underline{m}G : \underline{m} \in GF(q)^k, m_i = \alpha\}$ and $\{\underline{m}G : \underline{m} \in GF(q)^k, m_i = \beta\}$ are at distance $s(G)_i$ apart $(i = 1, \ldots, k)$. This observation implies the following error-correcting capability of a code when we use it on a q-ary symmetric channel.

Theorem 1. For a linear $[n,k]$ code C over $GF(q)$, which uses the matrix G for its encoding, complete nearest neighbour decoding guarantees the correct interpretation of the ith message digit whenever the error pattern has a Hamming weight less than or equal to $\lfloor (s(G)_i - 1)/2 \rfloor$ ($\lfloor x \rfloor$ denotes the largest integer less than or equal to x).

From Definition 1 it is immediately clear that the minimum distance of the code equals $d = \min \{s(G)_i : i = 1, \ldots, k\}$. If a linear code C has a generator matrix G such that the components of the separation vector $\underline{s}(G)$ are not mutually equal, then the code C is called a *linear unequal error protection (LUEP)* code.

One can easily decode LUEP codes by applying a syndrome decoding method using a standard array (cf. [7,p.15]). This decoding method reaches the correction capability given by Theorem 1, because of the following fact. For a fixed coset R of a linear code C, encoded by means of the generator matrix G, let U be the set of all possible coset leaders of R. For $\underline{r} \in R$, $\underline{r} + U$ contains all codewords that are closest to $\underline{r}$, i.e., at distance $d(\underline{r}, C) := \min \{wt(\underline{r} - \underline{c}) : \underline{c} \in C\}$ from $\underline{r}$. If $i \in \{1, \ldots, k\}$ is such that the weight of the elements of U is less than or equal to $\lfloor (s(G)_i - 1)/2 \rfloor$, then the ith digits of the messages corresponding to the elements of $\underline{r} + U$ are easily seen to be mutually equal. Hence, if $\underline{c} = \underline{m}G$ is the transmitted codeword and $\underline{r}$ is the received word such that $wt(\underline{r} - \underline{c}) \leq \lfloor (s(G)_i - 1)/2 \rfloor$ then syndrome decoding correctly reproduces the ith digit m_i of the message $\underline{m}$ sent.

For two vectors $\underline{x}, \underline{y} \in \mathbf{N}^k$ ($\mathbf{N}$ denotes the set of natural numbers) we define the ordering $\geq$ by

$$\underline{x} \geq \underline{y} \text{ if } x_i \geq y_i \text{ for all } i = 1, \ldots, k, \tag{2}$$

where the ordering $\geq$ in $x_i \geq y_i$ denotes the natural ordering in the integers. We call a vector $\underline{x} \in \mathbf{N}^k$ nonincreasing if $x_i \geq x_{i+1}$ for $i = 1, \ldots, k-1$.

By simultaneously permuting the message positions in the message words and the rows of a generator matrix G, we may obtain a generator matrix $\overline{G}$ for the code such that $\underline{s}(\overline{G})$ is nonincreasing, i.e., $s(\overline{G})_i \geq s(\overline{G})_{i+1}$ for $i = 1, \ldots, k-1$. From now on

we assume that the message positions and the rows in generator matrices are ordered such that the corresponding separation vectors are nonincreasing.

B. Optimal Encoding

The separation vector defined by (1) depends upon the choice of a generator matrix for the code. But, fortunately every code has a so-called *optimal generator matrix* G^*, whose separation vector $\underline{s}(G^*)$ is componentwise larger than or equal to the separation vector $\underline{s}(G)$ of any other generator matrix G of the code (notation: $\underline{s}(G^*) \geq \underline{s}(G)$). This was shown in [2]. From [2] we mention the following two results, which we will need later on. For a linear $[n,k]$ code C and $\rho \in \{0,\ldots,n\}$ let $C(\rho)$ denote the set of codewords in C of weight less than or equal to ρ, i.e., $C(\rho) := \{\underline{c} \in C : wt(\underline{c}) \leq \rho\}$.

Theorem 2 [2, Theorems 4 and 6]. a) A generator matrix G of a linear $[n,k]$ code C is optimal if and only if for any $\rho \in \{1,\ldots,n\}$ a subset X of rows of G exists such that the linear span $< C(\rho) >$ of $C(\rho)$ equals the linear span $< X >$ of X.
b) For $\rho \in \{1,\ldots,n\}$, $\dim< C(\rho) >$ - $\dim< C(\rho-1) >$ components of the separation vector of an optimal generator matrix G of a linear $[n,k]$ code C are equal to ρ.

Theorem 3 [2, Theorems 5 and 6]. For a linear $[n,k]$ code C a minimal weight generator matrix G, i.e., a generator matrix of C with the minimal number of nonzero entries, is optimal and satisfies $wt(G_{i*}) = s(G)_i$ for $i = 1,\ldots,k$, where G_{i*} denotes the ith row of G.

Hence the following definition makes sense.

Definition 2. The separation vector of a linear code is defined as the separation vector of an optimal generator matrix of the code.

We shall use the notation $[n,k,\underline{s}]$ for a linear code of length n, dimension k , and (nonincreasing) separation vector $\underline{s}$.

C. The canonical generator matrix

Boyarinov and Katsman [1] have introduced a special form of a generator matrix, called the canonical form.

Definition 3. A generator matrix G of a linear $[n,k]$ code, whose nonincreasing separation vector $\underline{s}(G)$ has z distinct components $t_1 > t_2 > \ldots > t_z$ with multiplicities respectively $k_1, k_2, \ldots, k_z$, is called *canonical* if the submatrix consisting of the k rows of G and the first k columns of G is a k-by-k lower triangular partitioned matrix having unit matrices of order repectively k_1-by-k_1, k_2-by-k_2,...,k_z-by-k_z on its diagonal. That is, G has the following form:

$$\left[\begin{array}{ccccc|c} I_{k_1} & O & \cdots & O & O & \\ G_{2,1} & I_{k_2} & \cdots & O & O & \\ \vdots & \vdots & \ddots & \vdots & \vdots & P \\ G_{z-1,1} & G_{z-1,2} & \cdots & I_{k_{z-1}} & O & \\ G_{z,1} & G_{z,2} & \cdots & G_{z,z-1} & I_{k_z} & \end{array}\right]. \qquad (3)$$

Any generator matrix G of a code can be transformed into a canonical generator matrix G_{can} of the code, such that $\underline{s}(G_{can}) \geq \underline{s}(G)$,by a number of elementary transformations on the rows of G, that are permutation and addition of rows and multiplication of rows by scalars (cf. [1],[4]). If we want to transform a generator matrix G into a systematic generator matrix G_{syst}, we cannot guarantee that $\underline{s}(G_{syst}) \geq \underline{s}(G)$. For example [4], for q=2,

$$G = \left[\begin{array}{cccccccccc} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \end{array}\right]$$

has separation vector $\underline{s}(G) = (5,4,4,4,4)$. It is easy to see that it is impossible to transform G into a systematic generator matrix G_{syst} such that $\underline{s}(G_{syst}) \geq (5,4,4,4,4)$. Actually, it can be easily verified that a 5-by-10 binary systematic generator matrix with a separation vector of at least (5,4,4,4,4) does not exist.

III. Bounds on the length of LUEP codes

A basic problem is to find LUEP codes with a given dimension and separation vector such that their length is minimal and hence their information rate is maximal.

Definition 4. For any prime power $q, k \in \mathbf{N}$, and $\underline{s} \in \mathbf{N}^k$ we define $n_q(\underline{s})$ as the length of the shortest linear code over $GF(q)$ of dimension k with a separation vector of at least $\underline{s}$, and $n_q^{ex}(\underline{s})$ as the length of the shortest linear code over $GF(q)$ of dimension k with separation vector (exactly) $\underline{s}$.

An $[n_q(\underline{s}), k, \underline{s}]$ code is called *optimal*, if an $[n_q(\underline{s}), k, \underline{t}]$ code with $\underline{t} \geq \underline{s}, \underline{t} \neq \underline{s}$ does not exist. For any prime power q, $k \in \mathbf{N}$, and $\underline{s}, \underline{t} \in \mathbf{N}^k$ the functions $n_q(.)$ and $n_q^{ex}(.)$ satisfy the following properties.

$$n_q(\underline{s}) \leq n_q^{ex}(\underline{s}), \tag{4}$$

$$\underline{s} \leq \underline{t} \implies n_q(\underline{s}) \leq n_q(\underline{t}), \tag{5}$$

$$\underline{s} \leq \underline{t} \not\Rightarrow n_q^{ex}(\underline{s}) \leq n_q^{ex}(\underline{t}). \tag{6}$$

To illustrate (6), observe that $n_2^{ex}(5,4,4) = 8$ (cf. Table I) and $n_2^{ex}(5,4,3) = 9$, which can be seen by easy verification.

Now we derive upper and lower bounds for these functions.

A. Upper bounds

The following theorem provides a trivial upper bound for $n_q(.)$ and $n_q^{ex}(.)$ and an easy way to construct LUEP codes. Let "|" denote concatenation.

Theorem 4. For any prime power q, $k \in \mathbf{N}, v \in \mathbf{N}$, and an arbitrarily partitioned vector $(\underline{s_1}|\underline{s_2}|\ldots|\underline{s_v}) \in \mathbf{N}^k$ we have

$$n_q^{ex}(\underline{s_1}|\underline{s_2}|\ldots|\underline{s_v}) \leq \sum_{i=1}^{v} n_q^{ex}(\underline{s_i}). \tag{7}$$

The same inequality holds for $n_q(.)$ (replace $n_q^{ex}(.)$ in (7) by $n_q(.)$).

Proof. For $u = 1, \ldots, v$, let G_u be a generator matrix of a code with length $n_q^{ex}(\underline{s}_u)$ and separation vector $\underline{s}(G_u) = \underline{s}_u$. Then $G := \mathrm{diag}(G_1, G_2, \ldots, G_k)$ has separation vector $\underline{s}(G) = (\underline{s}_1 | \underline{s}_2 | \ldots | \underline{s}_v)$.

□

Corollary 5. For any prime q, $k \in \mathbf{N}$, and $\underline{s} \in \mathbf{N}^k$ we have

$$n_q^{ex}(\underline{s}) \leq \sum_{i=1}^{k} s_i. \tag{8}$$

Proof. Apply Theorem 4 with $v = k$, and $G_u = [111|\ldots|1]$, 1-by-s_u, for all $u = 1, \ldots, k$.

□

Hence for any $\underline{s} \in \mathbf{N}^k$ it is possible to construct a k-dimensional code over $GF(q)$ with separation vector $\underline{s}$.

B. Lower bounds

We start with a trivial, but useful, lower bound on $n_q(.)$.

Theorem 6. For any prime power q, $k \in \mathbf{N}$, and $\underline{s} \in \mathbf{N}^k$ we have

$$n_q(s_1, s_2, \ldots, s_k) \geq n_q(s_1 - 1, s_2 - 1, \ldots, s_k - 1) + 1. \tag{9}$$

Proof. By deleting a column from a k by $n_q(\underline{s})$ matrix G with separation vector $\underline{s}(G) \geq (s_1, s_2, \ldots, s_k)$, we obtain a k-by-$(n_q(\underline{s}) - 1)$ matrix G' with separation vector $\underline{s}(G') \geq (s_1 - 1, s_2 - 1, \ldots, s_k - 1)$.

□

Theorem 7. For $q = 2$, any $k \in \mathbf{N}$ and $(s_1, s_2, \ldots, s_k) \in \mathbf{N}^k$ we have

$$n_2(s_1, s_2, \ldots, s_k) \geq n_2(2\lfloor \frac{s_1 + 1}{2} \rfloor, 2\lfloor \frac{s_2 + 1}{2} \rfloor, \ldots, 2\lfloor \frac{s_k + 1}{2} \rfloor) - 1. \tag{10}$$

The same inequality holds when we replace $n_2(.)$ by $n_2^{ex}(.)$.

Proof. By adding an overall parity-check to a binary $[n = n_2(s_1, \ldots, s_k), k]$ code with a separation vector of at least $(s_1, \ldots, s_k)$, we obtain an $[n+1, k]$ code with a separation vector of at least $(2\lfloor \frac{s_1+1}{2} \rfloor, 2\lfloor \frac{s_2+1}{2} \rfloor, \ldots, 2\lfloor \frac{s_k+1}{2} \rfloor)$.

□

Theorem 8. For any prime power q, $k \in \mathbf{N}$, and nonincreasing $\underline{s} \in \mathbf{N}^k$ we have

$$n_q(s_1, s_2, \ldots, s_k) \geq 1 + n_q(s_1, s_2, \ldots, s_{k-1}). \tag{11}$$

Proof. By deleting the column $\underline{e_k} := (0, 0, \ldots, 0, 1)^T$ and the kth row from an optimal canonical (cf. Definition 3) generator matrix of a linear $[n = n_q(\underline{s}), k]$ code over $GF(q)$ with a separation vector of at least $\underline{s}$, we obtain a generator matrix of an $[n-1, k-1]$ code with a separation vector of at least $(s_1, s_2, \ldots, s_{k-1})$.

□

Corollary 9. For any prime power q, $k, j \in \mathbf{N}, 1 \leq j \leq k$, and nonincreasing $\underline{s} \in \mathbf{N}^k$ we have

$$n_q(s_1, s_2, \ldots, s_k) \geq j + n_q(s_1, s_2, \ldots, s_{k-j}). \tag{12}$$

Corollary 10. For any prime power q, $k \in \mathbf{N}$, and nonincreasing $\underline{s} \in \mathbf{N}^k$ we have

$$n_q(s_1, s_2, \ldots, s_k) \geq s_1 + k - 1. \tag{13}$$

For $s_1 = s_2 = \ldots = s_k$ Corollary 10 reduces to the Singleton bound (cf. [7, Ch.1, Th.11]).

Theorem 11. For any prime power q and $k \in \mathbf{N}$, and any $v \in \mathbf{N}$ and nonincreasing $\underline{s} \in \mathbf{N}^k$ such that s_{v-1} is strictly larger than s_v and

$$\sum_{i=v}^{k} s_i \leq n_q^{ex}(\underline{s}) - 1 \tag{14}$$

we must have

$$n_q^{ex}(s_1,\ldots,s_k) \geq 1 + n_q(s_1-1,\ldots,s_{v-1}-1,s_v,\ldots,s_k). \quad (15)$$

Proof. Let $v \in \mathbf{N}$ and a nonincreasing vector $\underline{s} \in \mathbf{N}^k$ be such that $s_{v-1} > s_v$ and (14) holds. Let G be a minimal weight generator matrix of an $[n = n_q^{ex}(\underline{s}), k, \underline{s}]$ code over $GF(q)$. Because of (14) and Theorem 3, G has a column containing zero elements in the last $k-v+1$ positions. By deleting this column from G we obtain a k-by-$(n-1)$ matrix G', whose separation vector satisfies $\underline{s}(G') \geq (s_1-1,\ldots,s_{v-1}-1,s_v,\ldots,s_k)$, since $s_{v-1} > s_v$.

□

Theorem 12. For any prime power q, $k \in \mathbf{N}$ and nonincreasing $\underline{s} \in \mathbf{N}^k$ we have that

$$n_q^{ex}(s_1,\ldots,s_k) \geq s_i + n_q(\hat{s}_1,\ldots,\hat{s}_{i-1},\hat{s}_{i+1},\ldots,\hat{s}_k) \quad (16)$$

holds for any $i \in \{1,\ldots,k\}$, where

$$\hat{s}_j := \begin{cases} s_j - \lfloor (q-1)s_i/q \rfloor & \text{for } j < i; \\ \lceil s_j/q \rceil & \text{for } j > i, \end{cases} \quad (17)$$

where $\lceil x \rceil$ denotes the smallest integer larger than or equal to x.

Proof. Let C be a linear $[n = n_q^{ex}(\underline{s}), k, \underline{s}]$ code over $GF(q)$ and let G be a minimal weight generator matrix for C. By Theorem 3, $wt(G_{i*}) = s_i$ for all $i = 1,\ldots,k$.

Fix $i \in \{1,\ldots,k\}$. Without loss of generality the first s_i columns of G have a 1 in the ith row. Deleting these first s_i columns and the ith row from G, we obtain a $(k-1)$ by $(n-s_i)$ matrix $\hat{G}$. Clearly $\hat{G}$ has rank $(k-1)$, for otherwise there would be a nontrivial linear combination of rows of $\hat{G}$ that equals $\underline{0}$, and hence the corresponding linear combination of rows of G would have distance less than s_i to αG_{i*} for some $\alpha \in GF(q)\backslash\{0\}$, a contradiction. Hence $\hat{G}$ is a generator matrix of an $[n-s_i, k-1]$ code with separation vector $\underline{\hat{s}} := \underline{s}(\hat{G}) = (\hat{s}_1,\ldots,\hat{s}_{i-1},\hat{s}_{i+1},\ldots,\hat{s}_k)$.

Let $j \in \{1,\ldots,k\}, j \neq i$, and let $\underline{m} \in GF(q)^k$ be such that $m_i = 0, m_j \neq 0$, and $\underline{c} := \underline{m}G = (\underline{c_1}, \underline{c_2})$, where $\underline{c_1}$ has length s_i, satisfies $wt(\underline{c_2}) = \hat{s}_j$. Since $m_j \neq 0$, we have that

$$wt(\underline{c_1}) + \hat{s}_j \geq s_j. \quad (18)$$

Furthermore, for some $\alpha \in GF(q)\backslash\{0\}$ at least $\lceil wt(\underline{c}_1)/(q-1)\rceil$ components of $\alpha\underline{c}_1$ equal 1, and hence

$$wt(G_{i*} - \alpha\underline{c}) \leq s_i - \lceil wt(\underline{c}_1)/(q-1)\rceil + \hat{s}_j. \tag{19}$$

On the other hand we have that

$$wt(G_{i*} - \alpha\underline{c}) \geq \max\{s_i, s_j\}. \tag{20}$$

The combination of (18), (19), and (20) yields (16) and (17).

□

Lemma 13. For any prime power q, $k \in \mathbf{N}$, and nonincreasing $\underline{s} \in \mathbf{N}^k$ a linear $[n_q(\underline{s}), k]$ code over $GF(q)$ with a nonincreasing separation vector $\underline{s}^*$ such that $\underline{s} \leq \underline{s}^* \leq s_1\underline{1}$ ($s_1\underline{1}$ denotes the k-vector with all components equal to s_1) exists, i.e., $n_q^{ex}(\underline{s}^*) = n_q(\underline{s})$.

Proof. Let G be a minimal weight generator matrix of an $[n = n_q(\underline{s}), k]$ code with separation vector $\underline{s}(G) \geq \underline{s}$. If $s(G)_1 > s_1$ then replace a nonzero element in the first row of G by zero to obtain a matrix G', whose separation vector satisfies $\underline{s}(G') \geq \underline{s}$ and $s(G')_1 = s(G)_1 - 1$. We may transform G' into a minimal weight generator matrix G'' spanning the same linear subspace.

Now, we repeat the above procedure until we obtain a k-by-n matrix G^* such that $\underline{s} \leq \underline{s}(G^*) \leq s_1\underline{1}$.

□

The combination of Theorem 12 and Lemma 13 gives the following corollary.

Corollary 14. For any prime power q, $k \in \mathbf{N}$, and nonincreasing $\underline{s} \in \mathbf{N}^k$, $n_q(\underline{s})$ satisfies the inequalities

$$n_q(s_1, \ldots, s_k) \geq s_1 + n_q(\lceil s_2/q\rceil, \ldots, \lceil s_k/q\rceil), \tag{21}$$

$$n_q(s_1, \ldots, s_k) \geq \sum_{i=1}^{k} \lceil s_i/q^{i-1}\rceil. \tag{22}$$

For $s_1 = s_2 = \ldots = s_k$ Corollary 14 reduces to the Griesmer bound (cf. [7,Ch. 17, Th.24]). Deleting the $\lceil\ \rceil$ brackets in (22)

we obtain an analog of the Plotkin bound (cf. [7,Ch.2,Th.1]) for LUEP codes.

Lemma 13 also implies the following corollary.

Corollary 15. For any prime power q, $k \in \mathbf{N}$, and nonincreasing $\underline{s} \in \mathbf{N}^k$ we have

$$n_q(\underline{s}) = \min \{n_q^{ex}(\underline{s}') : \underline{s} \leq \underline{s}' \leq s_1\underline{1}\}. \tag{23}$$

This corrollary allows us to use the bounds on $n_q^{ex}(.)$ to obtain bounds on $n_q(.)$.

Katsman[6] has shown Corollary 14 for $q = 2$. In many cases a combination of Corollary 15 and the bounds on $n_q^{ex}(.)$ give better results than Corollary 14. For example, Corollary 14 yields that $n_2(5,4,3,3,3,3) \geq 11$, while a combination of Corollary 15 and the Theorems 6 and 12 yield that $n_2(5,4,3,3,3,3) \geq 12$ (using the values of Table I for $n \leq 10$). Actually $n_2(5,4,3,3,3,3) = 12$ (cf. Table I). Another interesting fact is to observe that Theorem 12 gives better results than the bound in [6], i.e., Theorem 12 for $i = 1$ and $q = 2$. For example, Theorem 12 yields that

$$n_2^{ex}(6,6,3,3,3,3,3) \geq 6 + n_2(3,2,2,2,2,2) = 14, \text{ for } i = 1$$

and

$$n_2^{ex}(6,6,3,3,3,3,3) \geq 3 + n_2(5,5,2,2,2,2) = 15, \text{ for } i = 7.$$

Table I provides the separation vectors of the binary optimal LUEP codes of length less than or equal to 15. n denotes the length of the code, k denotes the dimension, and $d(n,k)$ denotes the maximal minimum distance of a binary linear code of length n and dimension k. The brackets and commas commonly appearing in separation vectors have been deleted. Only in the cases where a component of a separation vector is larger than 9, it is followed by a point (.). The construction of the codes in Table I can be found in the Appendix. The various possibilities for a separation vector of an optimal LUEP code of small length and dimension show how difficult it would be to determine all possibilities for larger lengths and dimensions.

n	k	$d(n,k)$	separation vector
4	2	2	A 32
5	2	3	A 42
5	3	2	A 322
6	2	4	A 52
6	3	3	A 422
6	4	2	A 3222
7	2	4	A 62, I 54
7	3	4	A 522
7	4	3	A 4222
7	5	2	A 32222
8	2	5	A 72, I 64
8	3	4	A 622, C 544
8	4	4	A 5222
8	5	2	A 42222, J 33332
8	6	2	A 322222
9	2	6	A 82, I 74
9	3	4	A 722, C 644, G 554
9	4	4	A 6222, C 5444
9	5	3	A 52222, J 44442, B 43333
9	6	2	A 422222, J 333322
9	7	2	A 3222222
10	2	6	A 92, I 84, I 76
10	3	5	A 822, C 744, L 664
10	4	4	A 7222, C 6444, G 5544
10	5	4	A 62222, C 54444
10	6	3	A 522222, J 444422, J 433332
10	7	2	A 4222222, J 3333222
10	8	2	A 32222222
11	2	7	A 10.2, I 94, I 86
11	3	6	A 922, C 844, K_1 764
11	4	5	A 8222, C 7444, E 6644
11	5	4	A 72222, C 64444, G 55444

Table I: *The separation vectors of all binary optimal LUEP codes of length less than or equal to 15.*

n	k	$d(n,k)$	separation vector
11	6	4	A 622222, J 544442, B 533333
11	7	3	A 5222222, J 4444222, J 4333322
11	8	2	A 42222222, J 33332222
11	9	2	A 322222222
12	2	8	A 11.2, I 10.4, I 96
12	3	6	A 10.22, C 944, E 864, K_2 774, K_1 766
12	4	6	A 9222, C 8444, K_1 7644
12	5	4	A 82222, C 74444, E 66444, M 55554
12	6	4	A 722222, C 644444, G 554444
12	7	4	A 6222222, J 5444422, J 5333332
12	8	3	A 52222222, J 44442222, J 43333222
12	9	2	A 422222222, J 333322222
12	10	2	A 3222222222
13	2	8	A 12.2, I 11.4, I 10.6, I 98
13	3	7	A 11.22, C 10.44, K_1 964, E 884, L 866
13	4	6	A 10.222, C 9444, L 8644, F 7744, K_1 7666
13	5	5	A 92222, C 84444, K_1 76444, L 66664, H 66555
13	6	4	A 822222, C 744444, D 664444, M 555544
13	7	4	A 7222222, J 6444442, B 6333333, J 5544442, K_1 5444444
13	8	4	A 62222222, J 54444222, J 53333322
13	9	3	A 522222222, J 444422222, J 433332222
13	10	2	A 4222222222, J 3333222222
13	11	2	A 32222222222
14	2	9	A 13.2, I 12.4, I 11.6, I 10.8
14	3	8	A 12.22, C 11.44, L 10.64, K_1 984, K_1 966
14	4	7	A 11.222, C 10.444, K_1 9644, L 8844, L 8666
14	5	6	A 10.2222, C 94444, L 86444, F 77444, N 76666

Table I(*continued*): *The separation vectors of all binary optimal LUEP codes of length less than or equal to 15.*

n	k	$d(n,k)$	separation vector
14	6	5	A 922222, C 844444, E 764444, L 666644, J 665552
14	7	4	A 8222222, C 7444444, J 6644442, Q 6544444, M 5555444
14	8	4	A 72222222, J 64444422, J 63333332, J 55444422 K_1 54444444
14	9	4	A 622222222, J 544442222, J 533333222
14	10	3	A 5222222222, J 4444222222, J 4333322222
14	11	2	A 42222222222, J 33332222222
14	12	2	A 322222222222
15	2	10	A 14.2, I 13.4, I 12.6, I 11.8
15	3	8	A 13.22, C 12.44, K_1 11.64, K_1 10.84, L 10.66, K_2 994, K_1 988
15	4	8	A 12.222, C 11.444, L 10.644, K_1 9844, K_1 9666
15	5	7	A 11.2222, C 10.4444, K_1 96444, L 88444, L 86666
15	6	6	A 10.22222, C 944444, L 864444, K_2 774444, J 766662, K_1 766644, O 765554
15	7	5	A 9222222, C 8444444, P 7644444, L 6666444, J 6655522
15	8	4	A 82222222, J 74444442, B 73333333, J 66444422, J 65444442 L 64444444, R 55554443, S 55544444
15	9	4	A 722222222, J 644444222, J 633333322, J 554444222 K_1 544444444
15	10	4	A 6222222222, J 5444422222, J 5333332222
15	11	3	A 52222222222, J 44442222222, J 43333222222
15	12	2	A 422222222222, J 333322222222
15	13	2	A 3222222222222

Table I(*continued*): *The separation vectors of all binary optimal LUEP codes of length less than or equal to 15.*

In [4] also a number of constructions of optimal LUEP codes and

methods for combining (LUEP) codes to obtain LUEP codes of larger length are given.

IV. Cyclic UEP codes

A. The separation vector of a cyclic UEP code

A cyclic $[n,k]$ code over $GF(q)$ is the direct sum of a number of minimal ideals in the residue class ring $GF(q)[x]/(x^n-1)$ of polynomials in x over $GF(q)$ modulo (x^n-1) (cf. [7,Ch.8,Sec.3]).

Theorem 16 [2]. For a cyclic code C that is the direct sum of v minimal ideals, an ordering $M_1, M_2, \ldots, M_v$ of generator matrices of these minimal ideals exist such that

$$G := \left[\begin{array}{c} M_1 \\ \hline M_2 \\ \hline \vdots \\ \hline M_v \end{array}\right] \tag{24}$$

is an optimal generator matrix.

Proof. For $\rho \in \{1,\ldots,n\}$, $< C(\rho) >$ is a cyclic code. Hence $< C(\rho) >$ is the direct sum of minimal ideals of $GF(q)[x]/(x^n-1)$. By applying Theorem 2a) we get the theorem.

□

The following corollaries are immediate consequences of Theorems 2 and 16.

Corollary 17. For a minimal ideal in $GF(q)[x]/(x^n-1)$ all components of the separation vector are mutually equal.

Corollary 18. For a cyclic code C with an optimal generator matrix G defined by formula (24) the ith and jth component of the separation vector $\underline{s} = \underline{s}(G)$ are equal if the ith and jth row of G are in the same minimal ideal of $GF(q)[x]/(x^n-1)$.

If the weight of the generator polynomial of a cyclic code C equals the minimum distance d of the code, then all components of the separation vector are mutually equal, sind $C =< C(d) >$ (cf. Theorem 2). If this is not the case, we can compute the separation vector of a cyclic code by comparing the weight distributions of its cyclic subcodes. A number of separation vectors in Table II were computed in this way.

Theorem 19. For $i = 1, 2$ let M_i be a minimal ideal in $GF(q)[x]/(x^n - 1)$ with minimum distance d_i and weight distribution $(A_j^{(i)})_{j=0}^n$ such that $M_1 \neq M_2$ and $d_1 \geq d_2$; let $(A_j)_{j=0}^n$ be the weight distribution of their direct sum $M_1 \oplus M_2$. Then the components of the separation vector of $M_1 \oplus M_2$ are all equal to the minimum distance d of $M_1 \oplus M_2$ if $d < d_2$ or if $d = d_2$ and $A_d^{(2)} < A_d$; they take two different values if $d = d_2$ and $A_d^{(2)} = A_d$, namely d_2 and $\min\{j : A_j^{(2)} < A_j\}$.

Proof. If $d < d_2$ or if $d = d_2$ and $A_d^{(2)} < A_d$ then a sum of an element in $M_1 \backslash \{\underline{0}\}$ and one in $M_2 \backslash \{\underline{0}\}$ exists such that its weight equals d. For $d = d_2$ and $A_d^{(2)} = A_d$, if $A_j^{(2)} < A_j$ then a sum of an element in $M_1 \backslash \{\underline{0}\}$ and one in $M_2 \backslash \{\underline{0}\}$ exists such that its weight equals j; if $A_j^{(2)} = A_j$ it does not. Combining these observations with Theorem 16 and Corollary 18 proves the theorem.

□

B. A majority decoding method for certain binary cyclic UEP code classes

In this section we discuss certain classes of binary cyclic UEP codes which can be decoded by a majority decoding method. It is easy to implement this method and it is very useful whenever the number of independent votes on each message digit equals (or is not much less than) the separation component corresponding to that message position. For a cyclic $[n, k]$ code, we number the message positions from 0 to $k - 1$, the code positions form 0 to $n - 1$.

Unlike the usual definition [9,p.6] we define an orthogonal check set of size δ on a code position j of a linear $[n,k]$ code C to be δ subsets $B_1^{(j)}, B_2^{(j)}, \ldots, B_\delta^{(j)}$ of $\{0,1,\ldots,n-1\}$ that satisfy the following three conditions,

$$B_r^{(j)} \cap B_s^{(j)} = \emptyset \text{ for } r \neq s, \tag{25}$$

$$\bigcup_{r=1}^{\delta} B_r^{(j)} \subset \{0,1,\ldots,n-1\} \backslash \{j\}, \tag{26}$$

$$c_j = \sum_{p \in B_r^{(j)}} c_p \text{ for all } \underline{c} \in C \text{ and all } r \in \{1,\ldots,\delta\}. \tag{27}$$

We define the weight of a check $B_r^{(j)}$ as the number of elements in $B_r^{(j)}$. We extend this definition of orthogonal check sets on code positions to orthogonal check sets on message positions. For a binary linear $[n,k,\underline{s}]$ code C that uses an optimal generator matrix G for its encoding, we define an orthogonal check set of size δ on a message position j to be δ subsets $D_1^{(j)}, D_2^{(j)}, \ldots, D_\delta^{(j)}$ of $\{0,1,\ldots,n-1\}$ and δ linear functions $f_1, f_2, \ldots, f_\delta$ of $m_0, m_1, \ldots, m_{j-1}$ that satisfy the following three conditions, similar to (25)-(27),

$$D_r^{(j)} \cap D_s^{(j)} = \emptyset \text{ for } r \neq s, \tag{28}$$

$$\bigcup_{r=1}^{\delta} D_r^{(j)} \subset \{0,1,\ldots,n-1\}, \tag{29}$$

$$m_j = \sum_{p \in D_r^{(j)}} c_p + f_r(m_0, m_1, \ldots, m_{j-1})$$
$$\text{for all } \underline{c} \in C \text{ and all } r \in \{1,\ldots,\delta\}. \tag{30}$$

If we have an orthogonal check set of size δ_j on message position j for $j = 0,1,\ldots,k-1$ such that $\delta_0 \geq \delta_1 \geq \ldots \geq \delta_{k-1}$, then we perform the following decoding rule.

- Whenever we receive a vector $\underline{u}$, we estimate the message digit m_j for $j = 0,1,\ldots,k-1$, starting with $j = 0$ and increasing j by 1 until j equals $k-1$. We estimate m_j by $\hat{m}_j$ which is defined as the majority of the votes

$$\sum_{p \in D_r^{(j)}} u_p + f_r(\hat{m}_0, \hat{m}_1, \ldots, \hat{m}_{j-1}), \quad r = 1,\ldots,\delta_j. \tag{31}$$

It is easily seen that this majority decoding method guarantees the correct estimation of the jth message digit (i.e., $\hat{m}_j = m_j$), whenever the Hamming distance between the transmitted codeword $\underline{c} = \underline{m}G$ and the received word $\underline{u}$ is less than or equal to $\lfloor(\delta_j - 1)/2\rfloor$. For δ_j being even, an error in the jth message position is detected whenever the error pattern has Hamming weight $\delta_j/2$. Hence, the jth component s_j of the separation vector of the code C satisfies $s_j \geq \delta_j$ $(j = 0, 1, \ldots, k-1)$.

Now, we consider a special class of binary cyclic codes. Let $p(x)$ and $q(x)$ be two irreducible polynomials in $GF(2)[x]$ with degrees respectively k_p and k_q and exponents respectively T_p and T_q such that T_p and T_q are relatively prime. Let C_p and C_q be respectively $[T_p, k_p]$ and $[T_q, k_q]$ codes with check polynomials repectively $p(x)$ and $q(x)$. Furthermore let C_p have an orthogonal check set of size δ_p on any code position such that any check has the same even weight. These strong restrictions are sufficient to build codes that have orthogonal check sets on their message positions of "large" size. Define C to be the binary cyclic $[n := T_pT_q, k_p + k_q]$ code with check polynomial $p(x)q(x)$ and C^* the binary cyclic $[T_pT_q, k_p + k_q + 1]$ code with check polynomial $(x+1)p(x)q(x)$. The following theorem provides a lower bound on the first k_p components of the separation vectors $\underline{s} := \underline{s}((M_p^T|M_q^T)^T)$ and $\underline{s}^* := \underline{s}((M_p^T|M_q^T|M_0^T)^T)$ of respectively C and C^*, where M_p, M_q, and M_0 are generator matrices of the minimal ideals in $GF(2)[x]/(x^n+1)$ with check polynomials respectively $p(x), q(x)$, and $(x+1)$.

Theorem 20. a) The separation vector $\underline{s}$ of the code C defined above satisfies

$$s_i \geq \begin{cases} T_q\delta_p + 1 & \text{if } C_q \text{ is an even-weight code,} \\ & \text{for } i = 0, 1, \ldots, k_p - 1; \\ T_q\delta_p & \text{otherwise, for } i = 0, 1, \ldots, k_p - 1. \end{cases} \tag{32}$$

b) The separation vector $\underline{s}^*$ of the code C^* defined above satisfies

$$s_i^* \geq T_q\delta_p, \quad \text{for } i = 0, 1, \ldots, k_p - 1. \tag{33}$$

Proof. a) Without loss of generality we consider the first message digit m_0. Let G_p and $G_q := [\underline{y_0}|\underline{y_1}|\ldots|\underline{y_{T_q-1}}]$ be systematic generator matrices of C_p respectively C_q. Without loss of generality the first column of G_p equals $\underline{e_0} := (1,0,0,\ldots,0)^T$.

Since C_p has an orthogonal check set of size δ_p on any code position such that any check has the same even weight, say $2w$, we have $2w\delta_p$ mutually different columns $\underline{a_i^{(1)}}, \underline{a_i^{(2)}}, \ldots, \underline{a_i^{(2w)}}$, $i = 1,\ldots,\delta_p$ of G_p such that

$$\underline{a_i^{(1)}} + \underline{a_i^{(2)}} + \ldots + \underline{a_i^{(2w)}} = \underline{e_0} \text{ for } i = 1,\ldots,\delta_p. \tag{34}$$

The matrix

$$G := \left[\begin{array}{cccc} G_p & G_p & \cdots & G_p \\ \hline G_q & G_q & G_q & \cdots \quad G_q \end{array}\right] \tag{35}$$

is an optimal generator matrix of C. Since the greatest common divisor $gcd(T_p, T_q)$ of T_p and T_q equals one, any pair $(\underline{x}^T|\underline{y}^T)^T$, where $\underline{x}$ and $\underline{y}$ are columns of respectively G_p and G_q, occurs exactly once as a column of G. By combining this fact with formula (34), we get that for any $i \in \{1,\ldots,\delta_p\}$ and any $j \in \{0,\ldots,T_q-1\}$ the equality

$$\left[\begin{array}{c} a_i^{(1)} \\ \hline y_j \end{array}\right] + \left[\begin{array}{c} a_i^{(2)} \\ \hline y_j \end{array}\right] + \cdots + \left[\begin{array}{c} a_i^{(2w)} \\ \hline y_j \end{array}\right] = \left[\begin{array}{c} e_0 \\ \hline 0 \end{array}\right] \tag{36}$$

holds, where the $2w\delta_p T_q$ columns occuring in the left-hand side (LHS) in (36) are mutually different columns of G.

Equation (36) implies $T_q\delta_p$ orthogonal checks on the message digit m_0. (In this case, the linear functions f_r $(r = 1,\ldots,T_q\delta_p)$ in formula (30) are all equal to the zero mapping.) If C_q is an even-weight code, $m_0 = c_0 + c_{T_p} + c_{2T_p} + \cdots + c_{(T_q-1)T_p}$ is an additional check for m_0, orthogonal to the $T_q\delta_p$ previous ones.

b) Follows as in the proof of a).

□

If in addition C_q has an orthogonal check set of size δ_q on any code position such that any check has the same even weight, then we have the following lower bound for $\underline{s}$.

Theorem 21. a) If $wt((x^{T_q}+1)/q(x))$ is even and $T_q\delta_p + 1 >$

$T_p(\delta_q + 1)$, then the separation vector $\underline{s}$ of the $[T_pT_q, k_p + k_q]$ code with check polynomial $p(x)q(x)$ satisfies

$$s_i \geq \begin{cases} T_q\delta_p + 1 & \text{for } i = 0, \ldots, k_p - 1; \\ T_p(\delta_q + 1) & \text{for } i = k_p, \ldots, k_p + k_q - 1. \end{cases} \tag{37}$$

b) If $wt((x^{T_q} + 1)/q(x))$ is odd and $T_q\delta_p > T_p(\delta_q + 1)$, then the separation vector $\underline{s}$ of the $[T_pT_q, k_p + k_q]$ code with check polynomial $p(x)q(x)$ satisfies

$$s_i \geq \begin{cases} T_q\delta_p & \text{for } i = 0, \ldots, k_p - 1; \\ T_p(\delta_q + 1) & \text{for } i = k_p, \ldots, k_p + k_q - 1. \end{cases} \tag{38}$$

Proof. a) For $i = 0, \ldots, k_p - 1$, (37) was shown in Theorem 20. Without loss of generality we consider the message digit m_{k_p}. For $j = 0, 1, \ldots, T_p - 1$, m_{k_p} equals

$$m_{k_p} = c_{jT_q} + \sum_{i=0}^{k_p-1} m_i G_{i,jT_q}, \tag{39}$$

where G is the matrix of (35). If an error pattern of weight less than or equal to $\lfloor T_p(\delta_q + 1)/2 \rfloor$ occurs, then the message digits $m_0, \ldots, m_{k_p-1}$ are correctly decodable, since $T_q\delta_p + 1 > T_p(\delta_q + 1)$. If we fill in these values of $m_0, \ldots, m_{k_p-1}$ in (39), then the $T_p(\delta_q + 1)$ checks on m_{k_p} obtained from (36) and (39) are mutually orthogonal (i.e., satisfy formulas (28)-(30)). Hence $s_{k_p} \geq T_p(\delta_q + 1)$.
b) Follows as in a).

□

Example 1. Take $p(x) := x^3 + x + 1$, $q(x) := x^4 + x^3 + x^2 + x + 1$. $T_p = 7$, $T_q = 5$. The [7,3] code C_p with check polynomial $p(x)$ has an orthogonal check set of size $\delta_p = 3$ on any code position, where all checks have weight 2 (for example, for the 0-position we have the checks $\{1, 5\}$, $\{2, 3\}$, and $\{4, 6\}$). The [5,4] code C_q with check polynomial $q(x)$ has an orthogonal check set of size $\delta_q = 1$ on any code position, where the check has weight 4 (for example, for the 0-position we have the check $\{1, 2, 3, 4\}$). By Theorem 21 the [35,7] code C with check polynomial $p(x)q(x)$ has a separation

vector $\underline{s}$ which satisfies $\underline{s} \geq (16,16,16,14,14,14,14)$.

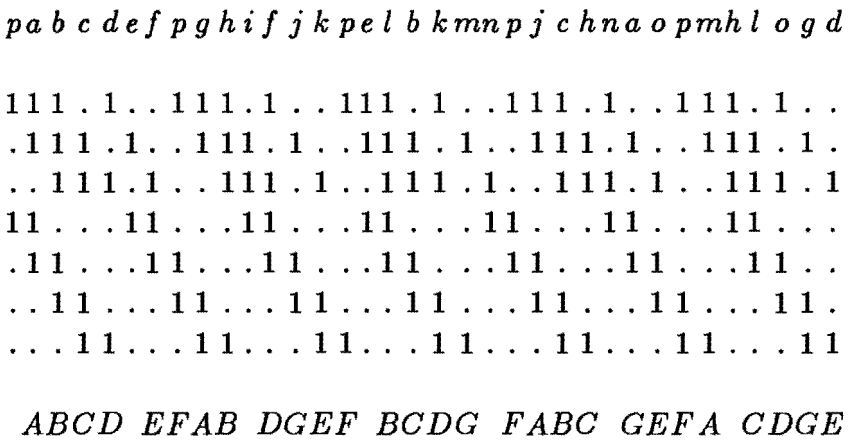


Figure 1: *Generator matrix of the [35,7] code with check polynomial* $(x^3 + x + 1)(x^4 + x^3 + x^2 + x + 1)$.

Figure 1 shows an optimal generator matrix for C (points (.) should be read as zeros (0)), together with two rows of characters, corresponding with the orthogonal checks on m_0 and m_3 given below. Note that, we did not take G_p and G_q to be systematic as we did in the proof of Theorem 20, because this is not necessary, but only convenient for the proof of Theorem 20. The generator matrix in Figure 1 directly shows that C_p and C_q are cyclic codes. The message bit m_0 is equal to the following sixteen orthogonal checks:

$$\begin{array}{lllllllll}
a: & c_1 + c_{26} & & f: & c_6 + c_{11} & & k: & c_{13} + c_{18} \\
b: & c_2 + c_{17} & & g: & c_8 + c_{33} & & l: & c_{16} + c_{31} \\
c: & c_3 + c_{23} & & h: & c_9 + c_{24} & & m: & c_{19} + c_{29} \\
d: & c_4 + c_{34} & & i: & c_{10} + c_{30} & & n: & c_{20} + c_{25} \\
e: & c_5 + c_{15} & & j: & c_{12} + c_{22} & & o: & c_{27} + c_{32}
\end{array}$$

$$p: \quad c_0 + c_7 + c_{14} + c_{21} + c_{28}.$$

The message bit m_3 is equal to the following fourteen orthogonal

checks:

$$
\begin{array}{lllllllllllll}
A: & c_1 & + & c_8 & + & c_{22} & + & c_{29} & c_0 & + & m_0 & & \\
B: & c_2 & + & c_9 & + & c_{16} & + & c_{23} & c_5 & + & m_1 & & \\
C: & c_3 & + & c_{17} & + & c_{24} & + & c_{31} & c_{10} & + & m_1 & + & m_2 \\
D: & c_4 & + & c_{11} & + & c_{18} & + & c_{32} & c_{15} & + & m_0 & + & m_1 \\
E: & c_6 & + & c_{13} & + & c_{27} & + & c_{34} & c_{20} & + & m_2 & & \\
F: & c_7 & + & c_{14} & + & c_{21} & + & c_{28} & c_{25} & + & m_0 & + & m_2 \\
G: & c_{12} & + & c_{19} & + & c_{26} & + & c_{33} & c_{30} & + & m_0 & + & m_1 + m_2.
\end{array}
$$

Actually the separation vector of C equals (16,16,16,14,14,14,14), as one can easily check.

The [35,8] code C^* with check polynomial $(x+1)p(x)q(x)$ has a separation vector equal to (15,15,15,7,7,7,7,7). For C^*, $a, b, c, \ldots, o$ are fifteen orthogonal checks for m_0; $A, B, C, \ldots, G$ are seven orthogonal checks for m_3. For the message bit m_7 we have the following seven checks:

$$
\begin{array}{lllllllllllll}
c_0 & + & c_7 & + & c_{14} & + & c_{21} & + & c_{28} & + & m_0 & & \\
c_1 & + & c_8 & + & c_{15} & + & c_{22} & + & c_{29} & + & m_0 & + & m_1 \\
c_2 & + & c_9 & + & c_{16} & + & c_{23} & + & c_{30} & + & m_0 & + & m_1 + m_2 \\
c_3 & + & c_{10} & + & c_{17} & + & c_{24} & + & c_{31} & + & m_1 & + & m_2 \\
c_4 & + & c_{11} & + & c_{18} & + & c_{25} & + & c_{32} & + & m_0 & + & m_2 \\
c_5 & + & c_{12} & + & c_{19} & + & c_{26} & + & c_{33} & + & m_1 & & \\
c_6 & + & c_{13} & + & c_{20} & + & c_{27} & + & c_{34} & + & m_2 & &
\end{array}
.
$$

We can extend Theorem 20 to codes with a check polynomial that is a product of more than two irreducible polynomials in $GF(2)[x]$.

Theorem 22. For $i = 1, \ldots, v$ let $p_i(x)$ be an irreducible polynomial in $GF(2)[x]$ of degree k_i and exponent T_i such that $gcd(T_i, T_j) = 1$ for all $i, j, i \neq j$, and let the $[T_i, k_i]$ binary code with check polynomial $p_i(x)$ have an orthogonal check set of size δ_i such that all checks have the same even weight. Then the code C of length $n = \prod_{i=1}^{v} T_i$ and dimension $\sum_{i=1}^{v} k_i$ with check polynomial $\prod_{i=1}^{v} p_i(x)$ has separation vector $\underline{s}$ that satisfies

$$s_j \geq n\delta_i / T_i \tag{40}$$

for $i = 1, \ldots, v$ and $j = (\sum_{u=1}^{i-1} k_u), \ldots, (\sum_{u=1}^{i} k_u - 1)$.

Proof. Analogous to Theorem 20.

□

In many cases we can do much better than formula (40) by adding other checks, e.g. checks like the ones in formula (39). This will be shown in the next example.

Example 2. Take $p(x) := x^3 + x + 1$, $q(x) := x^2 + x + 1$, and $r(x) := x^4 + x^3 + x^2 + x + 1$. $T_p = 7$, $T_q = 3$, $T_r = 5$. Let C be the [105,9] code with check polynomial $p(x)q(x)r(x)$. C has an optimal generator matrix $G := [M_p^T | M_q^T | M_r^T]^T$, where M_p, M_q and M_r are repetitions of respectively

$$P = \begin{bmatrix} 1110100 \\ 0111010 \\ 0011101 \end{bmatrix}, \; Q = \begin{bmatrix} 110 \\ 011 \end{bmatrix}, \; R = \begin{bmatrix} 11000 \\ 01100 \\ 00110 \\ 00011 \end{bmatrix}.$$

The [7,3] code with generator matrix P has three orthogonal checks $\{1,5\}, \{2,3\}$, and $\{4,6\}$ of weight 2 on code position 0. The [3,2] code with generator matrix Q has one check $\{1,2\}$ of weight 2 on code position 0. the [5,4] code with generator matrix R has one check $\{1,2,3,4\}$ of weight 4 on code position 0. Hence $\delta_p = 3$, $\delta_q = 1$, and $\delta_r = 1$. Any vector $(\underline{x}^T | \underline{y}^T | \underline{z}^T)^T$, where $\underline{x}, \underline{y}$ and $\underline{z}$ are columns of respectively P, Q, and R, occurs exactly once as a column of G. Now for any $i \in \{0, \ldots, T_q - 1\}$ and $j \in \{0, \ldots, T_r - 1\}$ we have

$$\begin{bmatrix} P_{*1} \\ Q_{*i} \\ R_{*j} \end{bmatrix} + \begin{bmatrix} P_{*5} \\ Q_{*i} \\ R_{*j} \end{bmatrix} = \begin{bmatrix} P_{*2} \\ Q_{*i} \\ R_{*j} \end{bmatrix} + \begin{bmatrix} P_{*3} \\ Q_{*i} \\ R_{*j} \end{bmatrix} = \begin{bmatrix} P_{*4} \\ Q_{*i} \\ R_{*j} \end{bmatrix} + \begin{bmatrix} P_{*6} \\ Q_{*i} \\ R_{*j} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{41}$$

This implies $T_q T_r \delta_p = 45$ orthogonal checks on m_0. These checks do not contain the fifteen code digits $\{c_j : j \equiv 0 \bmod 7\}$, which corresponds to the columns of G given in Figure 2.

0	21	42	63	84	35	56	77	98	14	70	91	7	28	49
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
1	1	1	1	1	.	.	.	.	.	1	1	1	1	1
.	.	.	.	.	1	1	1	1	1	1	1	1	1	1
1	1	.	.	.	1	1	.	.	.	1	1	.	.	.
.	1	1	.	.	.	1	1	.	.	.	1	1	.	.
.	.	1	1	.	.	.	1	1	.	.	.	1	1	.
.	.	.	1	1	.	.	.	1	1	.	.	.	1	1
a	*a*	*a*	*b*	*c*	*c*	*c*	*c*	*a*	*b*	*b*	*b*	*b*	*c*	*a*

Figure 2: *Columns G_{*j} of G where $j \equiv 0 \bmod 7$. The first row gives the column numbers.*

From Figure 2 it is easy to see that

$$\begin{array}{lccccccccc} a: & c_0 & + & c_{21} & + & c_{42} & + & c_{49} & + & c_{98} \\ b: & c_7 & + & c_{14} & + & c_{63} & + & c_{70} & + & c_{91} \\ c: & c_{28} & + & c_{35} & + & c_{56} & + & c_{77} & + & c_{84} \end{array}$$

are three additional orthogonal checks on m_0, which are orthogonal to the 45 checks implied by formula (41). Hence we have 48 orthogonal checks on m_0. Analogously we can find 48 orthogonal checks on m_1 and m_2.

For any $i \in \{0, \ldots, T_p - 1\}$ and any $j \in \{0, \ldots, T_r - 1\}$ we have

$$\begin{bmatrix} P_{*i} \\ Q_{*1} \\ R_{*j} \end{bmatrix} + \begin{bmatrix} P_{*i} \\ Q_{*2} \\ R_{*j} \end{bmatrix} = \underline{e_3} := (0,0,0,1,0,0,0,0,0)^T. \qquad (42)$$

This implies $T_p T_r \delta_q = 35$ orthogonal checks on m_3. These checks do not contain the code digits $\{c_j : j \equiv 0 \bmod 3\}$, which correspond to the columns of G given in Figure 3.

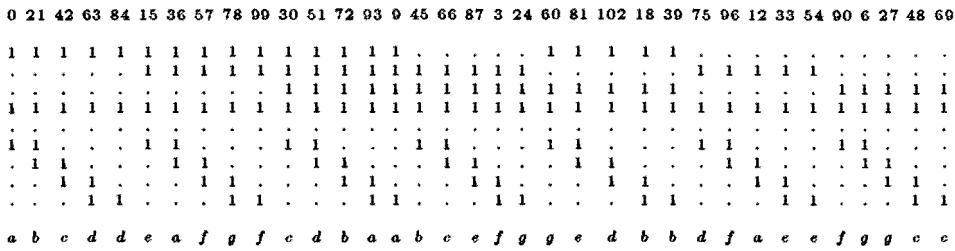


Figure 3: *Columns G_{*j} of G where $j \equiv 0 \bmod 3$.*

From Figure 3 it is easy to see that

$$\begin{array}{lccccccccc}
a: & c_0 & + & c_9 & + & c_{12} & + & c_{36} & + & c_{93} \\
b: & c_{18} & + & c_{21} & + & c_{39} & + & c_{45} & + & c_{72} \\
c: & c_{30} & + & c_{42} & + & c_{48} & + & c_{66} & + & c_{69} \\
d: & c_{51} & + & c_{63} & + & c_{75} & + & c_{84} & + & c_{102} \\
e: & c_{15} & + & c_{33} & + & c_{54} & + & c_{81} & + & c_{87} \\
f: & c_3 & + & c_{57} & + & c_{90} & + & c_{96} & + & c_{99} \\
g: & c_6 & + & c_{24} & + & c_{27} & + & c_{60} & + & c_{78}
\end{array}$$

are seven additional checks on m_3, which are orthogonal to the 35 checks implied by formula (42). Hence we have 42 orthogonal checks on m_3. Analogously we can find 42 checks on m_4.

For any $i \in \{0, \ldots, T_p - 1\}$ and any $j \in \{0, \ldots, T_q - 1\}$ we have

$$\begin{bmatrix} P_{*i} \\ Q_{*j} \\ R_{*1} \end{bmatrix} + \begin{bmatrix} P_{*i} \\ Q_{*j} \\ R_{*2} \end{bmatrix} + \begin{bmatrix} P_{*i} \\ Q_{*j} \\ R_{*3} \end{bmatrix} + \begin{bmatrix} P_{*i} \\ Q_{*j} \\ R_{*4} \end{bmatrix} = \underline{e_5} := (0,0,0,0,0,1,0,0,0)^T. \tag{43}$$

This implies $T_p T_q \delta_r = 21$ orthogonal checks on m_5. These checks do not contain the 21 code digits $\{c_j : j \equiv 0 \bmod 5\}$. Since $s_0, s_1, \ldots, s_4 \geq 42$,

$$c_j + \sum_{i=0}^{4} m_i G_{ij},$$

for $j \in \{0, 5, 10, \ldots, 100\}$ build 21 additional checks on m_5 (cf. the proof of Theorem 21). Hence we have 42 orthogonal checks on m_5. Analogously we can find 42 checks on m_6, m_7, and m_8.

We have shown that the [105,9] code with check polynomial $(x^3+x+1)(x^2+x+1)(x^4+x^3+x^2+x+1)$ has a separation vector

of at least (48,48,48,42,42,42,42,42,42). Actually equality holds, as one can easily check. So we have derived a majority decoding scheme for the code that reaches the actual separation vector.

For a binary cyclic code whose check polynomial is the product of $(x+1)$ and two primitive polynomials we have the following theorem.

Theorem 23. For the primitive polynomials $p(x), q(x) \in GF(2)[x]$ of degrees respectively k_p and k_q such that $k_p > k_q$ and $gcd(k_p, k_q) = 1$, the binary cyclic $[(2^{k_p}-1)(2^{k_q}-1), k_p+k_q+1]$ code with check polynomial $(x+1)p(x)q(x)$ has separation vector $\underline{s}$ that satisfies

$$s_i = \begin{cases} (2^{k_q}-1)(2^{k_p-1}-1) & \text{for } i = 0,\ldots,k_p-1, \\ (2^{k_p}-1)(2^{k_q-1}-1) & \text{for } i = k_p,\ldots,k_p+k_q. \end{cases} \tag{44}$$

Proof. The $[2^{k_p}-1, k_p]$ cyclic code C_p with primitive check polynomial $p(x)$ of degree k_p is a simplex code (also called a maximal-length feedback shift register code (cf. [7,p.31])), i.e., all elements of $GF(2)^{k_p}\backslash\{0\}$ occur as columns in a generator matrix of C_p). Hence we have an orthogonal check set of size $\delta_p := (2^{k_p-1}-1)$ on any code position, where all weights of the checks equal 2. The same holds for the $[2^{k_q}-1, k_q]$ code C_q with primitive check polynomial $q(x)$; $\delta_q := (2^{k_q-1}-1)$. Since $gcd(k_p, k_q) = 1$, we may apply Theorem 20b) to the first k_p message bits as well as to the message bits $m_{k_p},\ldots,m_{k_p+k_q-1}$.

Furthermore

$$wt((x^n+1)/p(x) + \sum_{i=0}^{n-1} x^i) = (2^{k_q}-1)(2^{k_p-1}-1)$$

and

$$wt((x^n+1)/q(x) + \sum_{i=0}^{n-1} x^i) = (2^{k_p}-1)(2^{k_q-1}-1),$$

where $n := (2^{k_p}-1)(2^{k_q}-1)$.

These observations imply (44) for $i = 0,1,\ldots,k_p+k_q-1$. From the observations above it also follows that

$$s_{k_p+k_q} = \min\{(2^{k_p}-1)(2^{k_q}-1), (2^{k_p}-1)(2^{k_q-1}-1), (2^{k_q}-1)(2^{k_p-1}-1)\}$$

$$= (2^{k_p} - 1)(2^{k_q - 1} - 1),$$

since $k_p > k_q$.

□

Dyn'kin and Togonidze [3] mentioned Theorem 23 without a proof.

Example 3. Take $k_p = 3$, $k_q = 2$, $p(x) = x^3 + x + 1$, $q(x) = x^2 + x + 1$.

$$G := \begin{bmatrix} 111.1..111.1..111.1.. \\ .111.1..111.1..111.1. \\ ..111.1..111.1..111.1 \\ 11.11.11.11.11.11.11. \\ .11.11.11.11.11.11.11 \\ 11111111111111111111 \end{bmatrix}$$

is an optimal generator matrix for the [21,6] cyclic code C with check polynomial $(x+1)p(x)q(x)$. $\underline{s}(G) = (9,9,9,7,7,7)$. m_0, m_3, and m_5 have the following checks:

$$\begin{aligned} m_0 &= c_1 + c_{19} = c_2 + c_{17} = c_3 + c_9 \\ &= c_4 + c_{13} = c_5 + c_8 = c_6 + c_{18} \\ &= c_{10} + c_{16} = c_{11} + c_{20} = c_{12} + c_{15}, \end{aligned}$$

$$\begin{aligned} m_3 &= c_1 + c_8 = c_2 + c_{16} = c_4 + c_{11} \\ &= c_5 + c_{19} = c_7 + c_{14} = c_{10} + c_{17} \\ &= c_{13} + c_{20}, \end{aligned}$$

$$\begin{aligned} m_5 &= c_0 + c_1 + c_2 + m_0 + m_2 \\ &= c_3 + c_4 + c_5 + m_0 \\ &= c_6 + c_7 + c_8 + m_1 + m_2 \\ &= c_9 + c_{10} + c_{11} + m_2 \\ &= c_{12} + c_{13} + c_{14} + m_0 + m_1 + m_2 \\ &= c_{15} + c_{16} + c_{17} + m_1 \\ &= c_{18} + c_{19} + c_{20} + m_0 + m_1. \end{aligned}$$

The generator matrix G' for this code, whose rows are the cyclic shifts of the generator polynomial, has separation vector $\underline{s}(G') = (7,7,7,7,7,7)$. A binary [21,6] code has a minimum distance of at most 8 (cf. [5]).

We can also extend Theorem 23 to the following theorem. The proof is analogous to that of Theorem 23.

Theorem 24. For the primitive polynomials $p_i(x) \in GF(2)[x]$, $i = 1, \ldots, v$ of degrees respectively $k_i, i = 1, \ldots, v$ such that $k_1 > k_2 > \ldots > k_v$ and $gcd(k_i, k_j) = 1$ for all $i, j, i \neq j$, the binary cyclic

$$[\prod_{i=1}^{v}(2^{k_i} - 1), 1 + \sum_{i=1}^{v} k_i]$$

code with check polynomial

$$(x+1)\prod_{i=1}^{v} p_i(x)$$

has a separation vector $\underline{s}$ which satisfies

$$s_i = (2^{k_j - 1} - 1)\prod_{u=1}^{v}(2^{k_u} - 1)/(2^{k_j} - 1) \tag{45}$$

for $i = \sum_{u=1}^{j-1} k_u, \ldots, \sum_{u=1}^{j} k_u - 1$ and $j = 1, \ldots, v$, and

$$s_i = (2^{k_v - 1} - 1)\prod_{u=1}^{v-1}(2^{k_u} - 1) \tag{46}$$

for $i = \sum_{u=1}^{v} k_u$.

Table II contains the parameters of all binary cyclic UEP codes of length less than or equal to 39. In this table, the exponents $i, j, k, \ldots$ of a primitve nth root of unity α are given for each code of length n such that $\alpha^i, \alpha^j, \alpha^k, \ldots$ are nonzeros of the code. For example, the first row of the table denotes a binary cyclic [15,7,(5,5,3,3,3,3,3)] code with nonzeros $\{\alpha^i \; ; \; i \in C_5 \cup C_0 \cup C_3\}$, where C_i $(i = 5, 0, 3)$ denotes the cyclotomic coset modulo 15 containing i. The order of the nonzeros corresponds to the order of the components in the separation vector, i.e., if the order of the nonzeros is $i, j, k, \ldots$, then the separation vector equals $\underline{s}((M_i^T | M_j^T | M_k^T | \ldots))$, where M_t $(t = i, j, k, \ldots)$ denotes a generator matrix of the minimal ideal in $GF(2)[x]/(x^n + 1)$ with nonzeros $\{\alpha^y : y \in C_t\}$. In the above example $\underline{s}((M_5^T | M_0^T | M_3^T)) = (5, 5, 3, 3, 3, 3, 3)$.

The last column of the table contains the minimum length or a bound on the minimum length of a binary linear code with a separation vector of at least the one of the corresponding cyclic code. The separation components (and the corresponding nonzeros) larger than the minimum distance of the code are underlined.

length	dim.	nonzeros	separation vector $\underline{s}$	$n(\underline{s})$
15	7	$\underline{5},0,3$	$\underline{5,5},3,3,3,3,3$	14
	9	$\underline{1},0,3$	$\underline{4,4,4,4},3,3,3,3,3$	14
	9	$\underline{0},1,7$	$\underline{5},4,4,4,4,4,4,4,4$	15
	11	$\underline{0},1,5,7$	$\underline{5},2,2,2,2,2,2,2,2,2,2$	15
	13	$\underline{0},1,3,7$	$\underline{3},2,2,2,2,2,2,2,2,2,2,2,2$	15
21	6	$\underline{3},0,7$	$\underline{9,9,9},7,7,7$	≥ 20
	7	$\underline{0},1$	$\underline{9},8,8,8,8,8,8$	21
	8	$\underline{7},3,9$	$\underline{8,8},6,6,6,6,6,6$	20
	9	$\underline{7},0,3,9$	$\underline{7,7},3,3,3,3,3,3,3$	18
	9	$\underline{0},1,7$	$\underline{7},6,6,6,6,6,6,6,6$	19
	10	$\underline{0},1,9$	$\underline{9},4,4,4,4,4,4,4,4,4$	20
	11	$\underline{7},1,9$	$\underline{6,6},4,4,4,4,4,4,4,4,4$	19
	12	$\underline{3},1,9$	$\underline{6,6,6},4,4,4,4,4,4,4,4,4$	≥ 20
	12	$\underline{7},0,1,3$	$\underline{6,6},5,5,5,5,5,5,5,5,5$	21
	13	$\underline{0},1,5$	$\underline{7},4,4,4,4,4,4,4,4,4,4,4,4$	21
	13	$\underline{1},0,3,9$	$\underline{4,4,4,4,4,4},3,3,3,3,3,3,3$	≥ 18
	15	$\underline{0},1,5,7$	$\underline{7},2,2,2,2,2,2,2,2,2,2,2,2,2,2$	21
	17	$\underline{3},1,5,7$	$\underline{4,4,4},2,2,2,2,2,2,2,2,2,2,2,2,$ $2,2$	21
	18	$\underline{0,3},1,5,7$	$\underline{3,3,3,3},2,2,2,2,2,2,2,2,2,2,2,$ $2,2,2$	21
	19	$\underline{0},1,3,5,9$	$\underline{3},2,2,2,2,2,2,2,2,2,2,2,2,2,2,$ $2,2,2,2$	21
25	21	$\underline{0},1$	$\underline{5},2,2,2,2,2,2,2,2,2,2,2,2,2,2,$ $2,2,2,2,2,2$	25

Table II: *All binary cyclic UEP codes of length less than or equal to 39.*

length	dim.	nonzeros	separation vector $\underline{s}$	$n(\underline{s})$
27	7	0, 3	9, 6, 6, 6, 6, 6, 6	19
	19	0, 1	9, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2	27
	20	9, 1	6, 6, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2	27
	21	0, 9, 1	3, 3, 3, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2	24
	25	0, 1, 3	3, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2	27
31	16	0, 1, 3, 15	9, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6	≥ 29
33	12	11, 3	12, 12, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6	≥ 29
	13	0, 1, 11	11, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10	≥ 32
	13	11, 0, 3	11, 11, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3	≥ 28
	21	0, 1, 5	11, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4	≥ 32
	23	0, 1, 5, 11	11, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2	33
	23	1, 11, 0, 3	5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3	≥ 32
	31	0, 1, 3, 5	3, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2	33
35	7	5, 7	16, 16, 16, 14, 14, 14, 14	≥ 34
	8	5, 0, 7	15, 15, 15, 7, 7, 7, 7, 7	≥ 32
	11	7, 0, 5, 15	7, 7, 7, 7, 5, 5, 5, 5, 5, 5, 5	22
	13	0, 1	15, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8	≥ 33
	15	5, 1	12, 12, 12, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8	≥ 33

Table II (*continued*): *All binary cyclic UEP codes of length less than or equal to 39.*

length	dim.	nonzeros	separation vector $\underline{s}$	$n(\underline{s})$
35	16	0, 1, 15	15, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4	≥ 32
	17	0, 1, 7	7, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6	≥ 28
	18	5, 1, 15	8, 8, 8, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4	≥ 29
	19	0, 5, 1, 15	5, 5, 5, 5, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4	≥ 26
	19	5, 1, 7	8, 8, 8, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6	≥ 30
	19	7, 1, 15	6, 6, 6, 6, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4	≥ 27
	20	0, 5, 7, 1	7, 7, 7, 7, 7, 7, 7, 7, 6, 6, 6, 6, 6, 6, 6, 6, 6	≥ 31
	22	5, 7, 1, 15	6, 6, 6, 6, 6, 6, 6, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4	≥ 31
	25	0, 1, 3	7, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4	≥ 33
	28	0, 1, 3, 5	5, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4	35
	29	0, 1, 3, 7	7, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2	35
	31	0, 1, 3, 5, 15	5, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2	35
	31	5, 1, 3, 7	4, 4, 4, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2	35
	32	0, 5, 1, 3, 7	3, 3, 3, 3, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2	35

Table II (*continued*): *All binary cyclic UEP codes of length less than or equal to 39.*

length	dim.	nonzeros	separation vector $\underline{s}$	$n(\underline{s})$
39	13	$\underline{0}, 1$	$\underline{15}, 12, 12, 12, 12, 12, 12, 12, 12, 12,$ $12, 12, 12$	≥ 37
	14	$\underline{13}, 3$	$\underline{14, 14}, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6$	≥ 35
	15	$\underline{0}, 1, 13$	$\underline{13}, 10, 10, 10, 10, 10, 10, 10, 10, 10,$ $10, 10, 10, 10, 10$	≥ 36
	15	$\underline{13}, 0, 3$	$\underline{13, 13}, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3$	≥ 33
	25	$\underline{1}, 0, 3$	$\underline{6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6}, 3, 3, 3$ $3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3$	≥ 35
	25	$\underline{0}, 1, 7$	$\underline{13}, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,$ $4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4$	39
	27	$\underline{1, 13}, 0, 3$	$\underline{6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6},$ $3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3$	≥ 37
	27	$\underline{0}, 1, 7, 13$	$\underline{12}, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,$ $2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2$	39
	37	$\underline{0}, 1, 3, 7$	$\underline{3}, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,$ $2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2$ $2, 2, 2, 2, 2, 2, 2$	39

Table II (*continued*): *All binary cyclic UEP codes of length less than or equal to 39.*

Acknowledgment

This paper is based on the author's Master's Thesis [4], written under supervision of Professor J.H. van Lint at the Eindhoven University of Technology. The subject "unequal error protection" was brought to the author's attention by L.M.H.E. Driessen of Philips Research Laboratories Eindhoven, The Netherlands.

The author wishes to thank Professor J.H. van Lint, Dr. H.C.A. van Tilborg, Professor J.-M. Goethals, and L.M.H.E. Driessen for their stimulating comments during the preparation of this paper.

Thanks are also due to the anonymous referees for their prompt and excellent reviews.

References

[1] I.M. Boyarinov and G.L. Katsman, "Linear unequal error protection codes", *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 168-175, March 1981.

[2] L.A. Dunning and W.E. Robbins, "Optimal encodings of linear block codes for unequal error protection", *Inform. Contr.*, vol. 37, pp. 150-177, 1978.

[3] V.N. Dyn'kin and V.A. Togonidze, "Cyclic codes with unequal symbol protection",*Problemy Peredachi Informatsii*, vol. 12, no. 1, pp. 24-28, 1976.

[4] W.J. van Gils, "On linear unequal error protection codes", Eindhoven Univ. Tech., EUT-Rep. 82-WSK-02, June 1982.

[5] H.J. Helgert and R.D. Stinaff, "Minimum-distance bounds for binary linear codes", *IEEE Trans. Inform. Theory*, vol. IT-19, pp. 344-356, May 1973.

[6] G.L. Katsman, "Bounds on volume of linear codes with unequal information-symbol protection",*Problemy Peredachi Informatsii*, vol. 16, no.2, pp. 25-32, 1980.

[7] F.J. MacWilliams and N.J.A. Sloane, *The Theory of Error-Correcting Codes*, Amsterdam: North-Holland 1978.

[8] B. Masnick and J. Wolf, ""On linear unequal error protection codes",*IEEE Trans. Inform. Theory*, vol. 13, pp. 600-607, Oct. 1967.

[9] J.L. Massey, *Threshold Decoding*, Cambridge, MA: MIT Press, 1963.

1.2

Linear unequal error protection codes from shorter codes

Wil J. van Gils

Abstract

The methods for combining codes, such as the direct sum, direct product, and $|u|u+v|$ constructions, concatenation, etc., are extended to linear unequal error protection codes.

I. Introduction

Consider a linear code C over the Galois field $GF(q)$ to be used on a q-ary symmetric channel. Input message words $\underline{m} \in GF(q)^k$ are encoded into codewords $\underline{c} = \underline{m}G \in GF(q)^n$, where G, k, and n denote a generator matrix, the dimension, and the length of the linear code, respectively.

In many applications it is necessary to provide different protection levels for different components m_i of the input message word $\underline{m}$. For example, in transmitting numerical data, errors in the high-order digits are more serious than are errors in the low-order digits, and therefore high-order digits should have more protection than low-order digits. A suitable measure for these protection levels on separate positions in input message words is the *separation vector* [3].

Definition. For a linear $[n, k]$ code C over the alphabet $GF(q)$, the *separation vector* $\underline{s}(G) = (s(G)_1, s(G)_2, \ldots, s(G)_k)$ with respect to a generator matrix G of C, is defined by

$$s(G)_i := \min\{wt(\underline{m}G) \mid \underline{m} \in GF(q)^k, m_i \neq 0\},\ i = 1, \ldots, k,$$

where $wt(.)$ denotes the Hamming weight function.

This separation vector $\underline{s}(G)$ guarantees the correct interpretation of the ith message digit whenever nearest neighbour decoding [8,p. 11] is applied and no more than $(s(G)_i - 1)/2$ errors have occurred in the transmitted codeword [3]. A linear code that has a generator matrix G such that the components of the corresponding separation vector $\underline{s}(G)$ are not mutually equal is called a linear unequal error protection (LUEP) code [3].

For our convenience we define the following ordering function. For a vector $\underline{x} \in \mathbf{N}^k$, the ordered version, $ord(\underline{x})$, of $\underline{x}$ is the vector resulting from ordering the components of $\underline{x}$ in non-increasing order, i.e. $\{x_i | i = 1, \ldots, k\} = \{ord(\underline{x})_i | i = 1, \ldots, k\}$ and $ord(\underline{x})_i \geq ord(\underline{x})_{i+1}$ for $i = 1, .., k-1$ ($\mathbf{N}$ denotes the set of natural numbers).

Any LUEP code C has a so-called optimal generator matrix G^*. That is, the ordered version of the separation vector $\underline{s}(G^*)$ of

G^* is componentwise larger than or equal to the ordered version of the separation vector $\underline{s}(G)$ of any generator matrix G of C [3], denoted by $ord(\underline{s}(G^*)) \geq ord(\underline{s}(G))$. The vector $\underline{s} = ord(\underline{s}(G^*))$ is called the separation vector of the linear code C. We use the notation $[n, k, \underline{s}]$ for C.

For any prime power $q, k \in \mathbf{N}$, and $\underline{s} \in \mathbf{N}^k$ we define $n_q(\underline{s})$ to be the length of the shortest linear code over $GF(q)$ of dimension k with a separation vector of at least $\underline{s}$ [5,6]. A linear $[n_q(\underline{s}), k, \underline{s}]$ code over $GF(q)$ is called *optimal* if a linear $[n_q(\underline{s}), k, \underline{t}]$ code over $GF(q)$ with $\underline{t} \geq \underline{s}, \underline{t} \neq \underline{s}$ does not exist [5,6]. In [5,6] a number of bounds on the function $n_q(\underline{s})$ are derived and a nearly complete list of all separation vectors of binary optimal LUEP codes of length less than 16 is given. In this paper we give a number of constructions of LUEP codes of longer length, which often yield optimal LUEP codes.

II. Combining codes to obtain LUEP codes of longer length

In this section we apply methods for combining codes to LUEP codes. Earlier constructions and/or construction methods were given in [1,2,3,7,13].

Construction 1. Let G_1 be an optimal generator matrix of an $[n, k, \underline{s} = \underline{s}(G_1)]$ code C_1 over $GF(q)$ such that $s_i \leq n(q-1)/q$ for all $i = 1, \ldots, k$. For $m \in \mathbf{N}$ and $j = 0, 1, \ldots, q^m - 1$ define A_j to be an m-by-n matrix whose columns are all equal to the q-ary representation of j, i.e. $\sum_{u=1}^{m} (A_j)_{uv} q^{u-1} = j$ for all $v = 1, ..., n$. Then the $(m + k)$ by nq^m matrix

$$G_2 = \left[\begin{array}{c|c|c|c} A_0 & A_1 & \cdots & A_{q^m-1} \\ \hline G_1 & G_1 & \cdots & G_1 \end{array}\right]$$

is a generator matrix of an $[nq^m, m+k, (n(q-1)q^{m-1}\underline{1}, q^m\underline{s})]$ code C_2 over $GF(q)$, where $\underline{1}$ denotes the all-one vector of length m. If

C_1 is an optimal LUEP code, so is C_2.

Proof. Using that $s_i \leq n(q-1)/q$ for all $i = 1, \ldots, k$, it is easy to show that the parameters of the code C_2 are those given above. Suppose C_1 is optimal. Then by using m applications of formula (21) from Corollary 14 in [6] the length of a $(k+m)$-dimensional code over $GF(q)$ with separation vector $(n(q-1)q^{m-1}\underline{1}, q^m\underline{s})$ is at least $n(q^m-1)+n_q(s_1,s_2,\ldots,s_k) = nq^m$ and similarly the length of a code with separation vector larger than $(n(q-1)q^{m-1}\underline{1}, q^m\underline{s})$ is at least nq^m+1 (by $\underline{x}$ is larger than $\underline{y}$ we mean that $\underline{x} \geq \underline{y}$ and $x_i > y_i$ for at least one index i). Hence C_2 is optimal.

□

Example 1. If in Construction 1 we take $m = 1$ and for G_1 a generator matrix of a binary $[2^t-1, 2^t-t-1, (3,3,\ldots,3)]$ Hamming code, then G_2 is a generator matrix of an optimal $[2^{t+1}-2, 2^t-t, (2^t-1, 6, 6, \ldots, 6)]$ binary LUEP code.

Construction 2 (the direct sum construction [10]). If for $i = a, b$, C_i is a linear $[n_i, k_i, \underline{s^i}]$ code, then the direct sum $\{(\underline{c_a}, \underline{c_b}) | \underline{c_a} \in C_a, \underline{c_b} \in C_b\}$ is a linear $[n_a+n_b, k_a+k_b, ord((\underline{s^a}, \underline{s^b}))]$ code.

Construction 3A (the $|u|u+v|$ construction [9,11]). If for $i = a, b$, C_i is a linear $[n, k_i, \underline{s^i}]$ code with an optimal generator matrix G_i such that $\underline{s^i} := \underline{s}(G_i) = ord(\underline{s}(G_i))$, then

$$G := \left[\begin{array}{c|c} G_a & G_a \\ \hline & G_b \end{array}\right] \tag{1}$$

is a generator matrix of a linear $[2n, k_a+k_b]$ code C , where $\underline{s} := \underline{s}(G)$ satisfies

$$s_j \geq \min\{2s_j^a\ ,\ \max\{s_j^a, s_{k_b}^b\}\},\quad \text{for } j = 1, \ldots, k_a,$$

$$s_{k_a+j} = s_j^b,\quad \text{for } j = 1, \ldots, k_b. \tag{2}$$

Proof. For $\underline{m^a} \in GF(q)^{k_a}, \underline{m^b} \in GF(q)^{k_b}, \underline{m} := (\underline{m^a}, \underline{m^b})$ we have that $wt(\underline{m}G) = wt(\underline{m^a}G_a) + wt(\underline{m^a}G_a + \underline{m^b}G_b)$. Hence for $j = 1, \ldots, k_a$ it holds that if $m_j^a \neq 0, \underline{m^b} = \underline{0}$ then $wt(\underline{m}G) \geq 2s_j^a$; if $m_j^a \neq 0, \underline{m^b} \neq \underline{0}$ then $wt(\underline{m}G) \geq wt(m^aG_a) \geq s_j^a$ and

$$\begin{aligned} wt(\underline{m}G) &\geq wt(\underline{m^a}G_a) + wt(\underline{m^b}G_b) - wt(\underline{m^a}G_a) \\ &= wt(\underline{m^b}G_b) \geq s_{k_b}^b, \end{aligned}$$

since $s^b_{k_b}$ is the minimum distance of the code C_b. This proves the first inequality in formula (2). The derivation of the second part of formula (2) is similar.

□

Example 2:. If, in the previous construction method, C_a is a binary [13,12] even weight code and C_b is a binary linear [13,6, (5,5,5,5,4,4)] code [5,6], then $s(G)_j \geq 4$ for $j = 1, \ldots, 12$, $s(G)_j = 5$ for $j = 13, 14, 15, 16$, and $s(G)_j = 4$ for $j = 17, 18$ by formula (2). Actually, C is a $[26, 18, (5, 5, 5, 5, 4, 4, 4, \ldots, 4)]$ LUEP code. This code is nearly optimal or optimal, since the length of an eighteen-dimensional binary code with a separation vector of at least $(5, 5, 5, 5, 4, 4, 4, \ldots, 4)$ can be shown to be at least 25 (by [6,Cor.14]).

Construction 3B (the $|u|u + v|parity(u)|$ construction [12]): If $q = 2$ then by adding the sum of the first n columns of the generator matrix G in formula (1) to this matrix G, we obtain a generator matrix G' of a $[2n + 1, k_a + k_b]$ code with separation vector $\underline{s} := \underline{s}(G')$ satisfying

$$s_j \geq \min \{s^a_j + 2\lceil s^a_j/2 \rceil \ , \ \max \{2\lceil s^a_j/2 \rceil, s^b_{k_b}\}\}, \quad \text{for } j = 1, \ldots, k_a,$$

$$s_{k_a+j} = s^b_j \ \text{ for } \ j = 1, \ldots, k_b,$$

($\lceil x \rceil$ denotes the smallest integer larger than or equal to x).

Example 3. Let C_a be the trivial [9,9] binary code and C_b be a binary [9,5,(4,3,3,3,3)] LUEP code (cf. [6,Table I]). Then Construction 3B yields a $[19, 14, (4, 3, 3, \ldots, 3)]$ code. Note that for a binary linear [19,14] code, the minimum distance is at most 3, and for a binary linear code of length 19 and minimum distance 3, the dimension is at most 14.

For completeness, we include the direct product construction[3].

Construction 4 (the direct product construction[3]). By taking the direct product [8,Ch. 18, Sec. 2] of a linear $[n_a, k_a]$ code with optimal generator matrix G_a and a linear $[n_b, k_b]$ code with

optimal generator matrix G_b, both over the same field, we obtain a linear $[n_a n_b, k_a k_b]$ code with optimal generator matrix $G_a \star G_b$ and separation vector $ord(\underline{s}(G_a) \star \underline{s}(G_b))$, where $\star$ denotes the Kronecker product.

Proof. Cf. [3,Ths. 3,8].

□

Construction 5 (concatenation[4]). Let C be a linear $[N, K, \underline{S} = (S_1, \ldots, S_K)]$ code over $GF(q^k)$ with an optimal generator matrix G_C, and let D be a linear $[n, k, d]$ code over $GF(q)$ with minimum distance d and generator matrix G_D. The encoding procedure of the concatenation of these codes is as follows.
Let $\underline{m} = (m_1^{(1)}, \ldots, m_k^{(1)} | \ldots\ldots | m_1^{(K)}, \ldots, m_k^{(K)})$ be a Kk-tuple over $GF(q)$. This Kk-tuple is equivalent to a K-tuple $\underline{M} = (M^{(1)}, \ldots, M^{(K)})$ over $GF(q^k)$, which is encoded into

$$(A^{(1)}, \ldots, A^{(N)}) := (M^{(1)}, \ldots, M^{(K)}) G_C.$$

Now we regard each $A^{(i)}$ as a k-tuple $(a_1^{(i)}, \ldots, a_k^{(i)})$ over $GF(q)$ and encode it into

$$(c_1^{(i)}, \ldots, c_n^{(i)}) := (a_1^{(i)}, \ldots, a_k^{(i)}) G_D \quad (i = 1, \ldots, N).$$

If $\underline{m}$ is a q-ary Kk-tuple such that $m_i^{(j)} \neq 0$, then $M^{(j)} \neq 0$ and hence, $\underline{A} = \underline{M} G_C$ satisfies $wt(\underline{A}) \geq S_j$, which in turn implies that $wt(\underline{c}) \geq S_j d$. Hence, we have shown the following theorem.

Theorem. The concatenation of a linear $[N, K, \underline{S} = (S_1, \ldots, S_K)]$ outer code over $GF(q^k)$ and a linear $[n, k, d]$ inner code over $GF(q)$ is a linear $[Nn, Kk, \underline{s}]$ code over $GF(q)$, where

$$s_{(j-1)k+i} \geq d S_j \tag{3}$$

for $i = 1, \ldots, k$ and $j = 1, \ldots, K$.

Example 4. Let α be a primitive element of $GF(4)$. The concatenation of the optimal [7,3,(5,4,4)] LUEP code over $GF(4)$ with generator matrix

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & \alpha & \alpha^2 & 0 \\ 0 & 0 & 1 & 1 & \alpha^2 & \alpha & 0 \end{bmatrix}$$

and the binary [3,2,2] even-weight code is a linear [21,6, (10,10,8,8,8,8)] binary LUEP code. The maximal minimum distance of a linear [21,6] binary code equals 8 [8,p. 676].

Example 5. Let α be a primitive element of $GF(8)$. The concatenation of the optimal [10,2,(9,8)] LUEP code over $GF(8)$ with generator matrix

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 & 0 \end{bmatrix}$$

and the [7,3,4] simplex code [8,p. 31] over $GF(2)$ is an optimal [70,6,(36,36,36,32,32,32)] binary LUEP code.

Acknowledgment

The author wishes to thank one of the anonymous referees for pointing out to him Vasil'ev's construction [12] which leads to Construction 3B and Example 3. Thanks are due to both referees for their prompt and excellent reviews.

References

[1] I.M. Boyarinov and G.L. Katsman, "Linear unequal error protection codes", *IEEE Trans. Inform. Theory*, vol. IT-27, no.2, pp. 168-175, March 1981.

[2] L.M.H.E. Driessen, "On an infinite series of [4n,2n] binary codes", *IEEE Trans. Inform. Theory*, vol. IT-30, no.2, pp. 392-395, March 1984.

[3] L.A. Dunning and W.E. Robbins, "Optimal encodings of linear block codes for unequal error protection", *Inform. Control*, vol.37, pp. 150-177, May 1978.

[4] G.D. Forney, Jr., *Concatenated Codes*, Cambridge, MA.: MIT, 1966.

[5] W.J. van Gils, "On linear unequal error protection codes", EUT-Rep. 82-WSK-02, Department of Mathematics and Computing Science, Eindhoven University of Technology, Eindhoven, The Netherlands, June 1982.

[6] W.J. van Gils, " Two topics on linear unequal error protection codes: bounds on their length and cyclic code classes", *IEEE Trans. Inform. Theory*, vol. IT-29, no.6, pp. 866-876, Nov. 1983.

[7] G.L. Katsman, "Bounds on volume of linear codes with unequal information- symbol protection", *Problemy Peredachi Informatsii*, vol. 16, no.2, pp. 25-32, April-June 1980.

[8] F.J. MacWilliams and N.J.A. Sloane, *The Theory of Error-Correcting Codes*, Amsterdam: North-Holland, 1978.

[9] M. Plotkin, "Binary codes with specified minimum distance", *IRE Trans. Inform. Theory*, vol. IT-6, no.4, pp. 445-450, Sept. 1960.

[10] D. Slepian, "Some Further Theory of Group Codes", *Bell Syst. Techn. J.*, vol. 39, pp. 1219-1252, Sept. 1960.

[11] N.J.A. Sloane and D.S. Whitehead, "A new family of single-error-correcting codes", *IEEE Trans. Inform. Theory*, vol. IT-16, no.6, pp. 717-719, Nov. 1970

[12] J.L. Vasil'ev, "On nongroup close-packed codes", *Problemy Kibernetiki*, vol. 8, pp. 337-339, 1962.

[13] V.A. Zinov'ev and V.V. Zyablov, "Codes with unequal protection of information symbols", *Problemy Peredachi Informatsii*, vol. 15, no. 3, pp. 50-60, July-Sept. 1979.

Appendix

Construction of binary LUEP codes of length less than or equal to 15

Wil J. van Gils

Abstract

This appendix gives constructions of codes whose parameters are given in Table I of Section 1.1. This table provides the separation vectors of all optimal binary LUEP codes of length less than or equal to 15.

This appendix gives constructions of codes whose parameters are given in Table I of Section 1.1. This table provides the separation vectors of all optimal binary LUEP codes of length less than or equal to 15. In the table, n denotes the length of the code, k denotes the dimension, and $d(n,k)$ denotes the maximal minimum distance of a binary code of length n and dimension k. The brackets and commas commonly appearing in separation vectors have been deleted. Only in the cases where a component of a separation vector is larger than 9, it is followed by a point (.). Examples of codes having the parameters given in Table I are constructed below. The bounds in Section 1.1 can be used to show that certain LUEP codes are optimal. They are also useful in showing that Table I is complete. In cases where these bounds did not work, methods of exhaustive search were used to show that codes with certain parameters do not exist. In this appendix we frequently use two results of Section 1.1. Hence we repeat these results in the following two theorems.

Theorem 1 [Section 1.1,Theorem 12]. For any $k \in \mathbf{N}$ and nonincreasing $\underline{s} \in \mathbf{N}^k$ we have that

$$n^{ex}(s_1,\ldots,s_k) \geq s_i + n(\hat{s}_1,\ldots,\hat{s}_{i-1},\hat{s}_{i+1},\ldots,\hat{s}_k)$$

holds for any $i \in \{1,\ldots,k\}$, where

$$\hat{s}_j := \begin{cases} s_j - \lfloor s_i/2 \rfloor & \text{for } j < i \\ \lceil s_j/2 \rceil & \text{for } j > i, \end{cases}$$

(where $\lfloor x \rfloor$ denotes the largest integer smaller than or equal to x, and $\lceil x \rceil$ denotes the smallest integer larger than or equal to x).

Theorem 2 [Section 1.1,Corollary 14]. For any $k \in \mathbf{N}$ and any nonincreasing $\underline{s} \in \mathbf{N}^k$, the function $n(\underline{s})$ satisfies the following inequalities,

a. $n(s_1,s_2,\ldots,s_k) \geq s_1 + n(\lceil s_2/2 \rceil,\ldots,\lceil s_k/2 \rceil)$,

b. $n(s_1,s_2,\ldots,s_k) \geq \sum_{i=1}^{k} \lceil s_i/2^{i-1} \rceil$.

Construction A. For $n, k \in \mathbf{N}, n \geq k+1$, the k by n matrix

$$\left[\begin{array}{c|c|c} & 1 & 1111\ldots\ldots111 \\ \hline & 1 & \\ & 1 & \\ I_k & \vdots & O_{k-1,n-k-1} \\ & 1 & \end{array}\right] \tag{1}$$

is a generator matrix of an optimal binary $[n, k, (n-k+1, 2, 2, \ldots, 2)]$ code (I_k denotes the identity matrix of order k, $O_{k-1,n-k-1}$ denotes the all-zero $(k-1)$-by-$(n-k-1)$ matrix).

Proof. It is easy to check that the parameters of the code are correct. Furthermore by Theorem 2b the length of a k-dimensional binary code with separation vector $(n-k+1, 2, 2, \ldots, 2)$ is at least n , and with separation vector larger than $(n-k+1, 2, 2, \ldots, 2)$ is at least $n+1$ (by $\underline{s} > \underline{t}$ ($\underline{s}$ larger than $\underline{t}$) we mean $\underline{s} \geq \underline{t}, \underline{s} \neq \underline{t}$).

□

Construction B. For $k \in \mathbf{N}, k \geq 4$, the k-by-$(2k-1)$ matrix

$$\left[\begin{array}{c|c|c} 00\ldots0 & 11\ldots1 & 0 \\ \hline & & 1 \\ & & 1 \\ I_{k-1} & I_{k-1} & \vdots \\ & & 1 \end{array}\right] \tag{2}$$

is a generator matrix of an optimal binary $[2k-1, k, (k-1, 3, 3, \ldots, 3)]$ code.

Proof. It is easy to verify that the parameters of the code are correct. By Theorem 2b , we have that the length of a k-dimensional binary code with separation vector $(k-1, 3, 3, \ldots, 3)$ is at least $2k-1$. Application of Theorem 2b to a k-vector $\underline{s}$ with $s_1 \geq k$ and $s_i \geq 3$ for $i = 2, \ldots, k$ shows that $n(\underline{s}) \geq 2k$. Application of the Theorems 1 and 2 to a k-vector $\underline{s}$ such that $s_1 = k-1$, $s_2 \geq 4$,

$s_i \geq 3$ for $i = 3, \ldots, k-1$, and $s_k = 3$ shows that

$$\begin{aligned} n^{ez}(\underline{s}) &\geq 3 + n(s_1 - 1, \ldots, s_{k-1} - 1) \\ &\geq 3 + s_1 - 1 + n(\lceil (s_2 - 1)/2 \rceil, \ldots, \lceil (s_{k-1} - 1)/2 \rceil) \\ &\geq 3 + k - 2 + n(\underbrace{2, 1, 1, \ldots, 1}_{k-2}) \\ &\geq 3 + k - 2 + k - 1 = 2k. \end{aligned}$$

Furthermore it is not difficult to check that a binary $[2k-1, k, (k-1, 4, 4, \ldots, 4)]$ code does not exist. Finally, by Theorem 2b the length of a k-dimensional binary code with separation vector of at least $(k-1, 5, 4, 4, \ldots, 4)$ is at least $2k$. These observations show that the code in Construction B is optimal.

□

Construction C. For $n, k \in \mathbf{N}, n \geq \max\{2k, k+4\}$, the k-by-n matrix

$$\left[\begin{array}{c|c|c|c} 00\ldots0 & 11\ldots1 & 11\ldots1 & 10 \\ \hline & & & 11 \\ & & & 11 \\ I_{k-1} & I_{k-1} & O & \vdots \\ & & & 11 \end{array}\right] \tag{3}$$

is a generator matrix of an optimal binary $[n, k, (n-k, 4, 4, \ldots, 4)]$ code.

Proof. Similar to the proof of Construction A.

□

Construction D. For $p, q \in \mathbf{N}, p \geq q \geq 2$, the $(p+q+2)$-by-

$(2p + 3q + 3)$ matrix

$$\left[\begin{array}{c|c|c|c|c} & & & & \overbrace{}^{q-1} \\ 00\ldots0 & 1110 & 11\ldots1 & 00\ldots0 & 11\ldots1 \\ 00\ldots0 & 1101 & 00\ldots0 & 11\ldots1 & 11\ldots1 \\ \hline & 0101 & & & \\ & 0101 & & & \\ & \vdots\vdots & I_p & O & O \\ & 0101 & & & \\ I_{p+q} & \hline 1010 & & & \\ & 1010 & & & \\ & \vdots\vdots & O & I_q & O \\ & 1010 & & & \end{array}\right] \tag{4}$$

is a generator matrix of an optimal binary $[2p + 3q + 3, p + q + 2, (p + q + 2, 2q + 2, 4, 4, \ldots, 4)]$ code.

Proof. Similar to the proof of Construction A.

□

Construction E. For $p, q, r \in \mathbf{N}, p \geq 3, r \geq 2, q \geq r - p + 2$, the p-by-$(2p + q + 2r - 4)$ matrix

$$\left[\begin{array}{c|c|c|c|c} 11\ldots1 & 00\ldots0 & \overbrace{11\ldots1}^{r} & \overbrace{00\ldots0}^{r} & \overbrace{11\ldots1}^{q} \\ 00\ldots0 & 00\ldots0 & 11\ldots1 & 11\ldots1 & 00\ldots0 \\ \hline & & 1 & 1 & 0 \\ I_{p-2} & I_{p-2} & \vdots\ O & \vdots\ O & \vdots\ O \\ & & 1 & 1 & 0 \end{array}\right] \tag{5}$$

is a generator matrix of an optimal binary $[2p + q + 2r - 4, p, (p + q + r - 2, 2r, 4, 4, \ldots, 4)]$ code.

Proof. Similar to the proof of Construction A.

□

Construction F. For $p, q \in \mathbf{N}, p \geq 3, q \geq 2, q \geq p - 2$, the

p-by-$(p+3q)$ matrix

$$\left[\begin{array}{c|c|c|c|c} & & \overbrace{\quad}^{q+1} & \overbrace{\quad}^{q+1} & \overbrace{\quad}^{q-(p-2)} \\ 00\ldots0 & 11\ldots1 & 11\ldots1 & 00\ldots0 & 11\ldots1 \\ 00\ldots0 & 11\ldots1 & 00\ldots0 & 11\ldots1 & 11\ldots1 \\ \hline & & 1 & 1 & 0 \\ I_{p-2} & I_{p-2} & \vdots\ O & \vdots\ O & \vdots\ O \\ & & 1 & 1 & 0 \end{array}\right] \tag{6}$$

is a generator matrix of an optimal binary $[p+3q, p, (2q+1, 2q+1, 4, 4, \ldots, 4)]$ code.

Proof. Similar to the proof of Construction A. □

Construction G. For $p \in \mathbf{N}$, the $2p$-by-$4p$ matrix

$$\left[\begin{array}{c|c|c|c} 00\ldots0 & 1110 & 11\ldots1 & 00\ldots0 \\ 00\ldots0 & 1101 & 00\ldots0 & 11\ldots1 \\ \hline & 1010 & & \\ & \vdots\vdots & O & I_{p-1} \\ & 1010 & & \\ I_{2p-2} & \hline 0101 & & \\ & \vdots\vdots & I_{p-1} & O \\ & 0101 & & \end{array}\right] \tag{7}$$

is a generator matrix of a binary $[4p, 2p, (p+2, p+2, 4, 4, \ldots, 4)]$ code. For $p = 2, 3$ the codes are optimal, but in general they are not.

In [1] the codes from Construction G are treated extensively, the weight enumerators and automorphism groups are determined completely and a majority logic decoding method for these codes is given. For $p = 3$ we obtain a [12,6,(5,5,4,4,4,4)] optimal LUEP code. By deleting the row and column pairs (6,4), (5,3), and (4,2) successively we obtain [11,5,(5,5,4,4,4)], [10,4,(5,5,4,4)], and [9,3,(5,5,4)] optimal LUEP codes respectively.

Construction H. For $p \in \mathbf{N}, p \geq 3$, the $(p+2)$-by-$(4p+1)$ matrix

$$\left[\begin{array}{c|c|c|c|c} 00\ldots0 & 11\ldots1 & 11\ldots1 & 00\ldots0 & 0 \\ 00\ldots0 & 11\ldots1 & 00\ldots0 & 11\ldots1 & 0 \\ \hline & & & & 1 \\ & & & & 1 \\ I_p & I_p & I_p & I_p & \vdots \\ & & & & 1 \end{array}\right] \tag{8}$$

is a generator matrix of a length-optimal binary $[4p+1, p+2, (2p, 2p, 5, 5, \ldots, 5)]$ LUEP code.

Proof. It is easy to check that the code has the given parameters. By Theorem 2b the length of a $(p+2)$-dimensional binary code with separation vector $(2p, 2p, 5, 5, \ldots, 5)$ is at least $4p+1$. In general an $[n(\underline{s}), k, \underline{s}]$ code is called *length-optimal.*

□

For $p = 3$ this construction gives a [13,5,(6,6,5,5,5)] optimal LUEP code.

Furthermore Table I refers to the following trivial constructions.

Construction I. For $p, q \in \mathbf{N}, p > q$, the 2-by-$(p+2q)$ matrix

$$\left[\begin{array}{c|c|c} 11\ldots1 & 00\ldots0 & 11\ldots1 \\ \underbrace{00\ldots0}_{p} & \underbrace{11\ldots1}_{q} & \underbrace{11\ldots1}_{q} \end{array}\right] \tag{9}$$

is a generator matrix of an optimal binary $[p+2q, 2, (p+q, 2q)]$ LUEP code.

Construction J. If the matrix G_1 has separation vector $\underline{s}(G_1)$ such that $\underline{s}(G_1)_k \geq 2$, then the matrix

$$\left[\begin{array}{c|c} 0 & \\ 0 & \\ \vdots & G_1 \\ 0 & \\ \hline 1 & 1000\ldots0 \end{array}\right] \tag{10}$$

has separation vector $\underline{s}(G_2) = (\underline{s}(G_1), 2)$.

Construction K_i. If the matrix G_1 has separation vector $\underline{s}(G_1)$ then the matrix

$$G_2 := \left[\; G_1 \; | \; \underline{e_i} \; \right] \tag{11}$$

where $\underline{e_i}$ is the vector with a 1 on the ith position and zeros elsewhere, has separation vector $\underline{s}(G_2) = \underline{s}(G_1) + \underline{e_i}$.

The following theorem can be used to determine whether construction K_1 gives an optimal code.

Theorem 3. If $\underline{s}$ is such that for all $\underline{t} \geq \underline{s}, \underline{t} \neq \underline{s}$, it holds that $n(\underline{t}) > n(\underline{s})$ and if G is a generator matrix of a binary optimal $[r+n(\underline{s}), k, (r, 2\underline{s})]$ code, then the code generated by $[\; G \; | \underbrace{\underline{e_1}|\underline{e_1}|\ldots|\underline{e_1}}_{t}]$ is a binary optimal $[r + t + n(\underline{s}), k, (r + t, 2\underline{s})]$ code for t in $\mathbf{N}$ arbitrary.

Proof. Let $\underline{s}$ and G fulfill the conditions mentioned above. By Theorem 2a we have that

a) $n(r + t, 2\underline{s}) \geq r + t + n(\underline{s})$.

b) $n(r + t + 1, 2\underline{s}) \geq r + t + 1 + n(\underline{s}) > r + t + n(\underline{s})$.

c) $n(r+t, 2\underline{s}+\underline{u}) \geq r+t+n(\lceil s_1+u_1/2\rceil, \ldots, \lceil s_{k-1}+u_{k-1}/2\rceil) \geq r + t + 1 + n(\underline{s})$ for $\underline{u} \geq \underline{0}, \underline{u} \neq \underline{0}$.

Combination of a),b), and c) shows that the code generated by $[\; G \; | \underbrace{\underline{e_1}|\underline{e_1}|\ldots|\underline{e_1}}_{t}]$ is optimal.

□

Construction L. Adding an overall parity-check bit to a binary $[n, k, \underline{s} = (s_1, \ldots, s_k)]$ code gives a binary $[n + 1, k, \underline{s'} = (2\lceil (s_1 + 1)/2\rceil, \ldots, 2\lceil (s_k + 1)/2\rceil)]$ code.

Sporadic constructions referred to in Table I are the following.

Construction M. The 7-by-14 matrix

$$\begin{bmatrix} 00011111000000 \\ 00011000111000 \\ 00010100100101 \\ 00001010010011 \\ 00110001100000 \\ 01001010001000 \\ 10000000000111 \end{bmatrix} \tag{12}$$

is a generator matrix of an optimal binary [14,7,(5,5,5,5,4,4,4)] LUEP code. Deleting the first column and the last row from the matrix in (12) gives an optimal binary [13,6,(5,5,5,5,4,4)] code. Deleting the first two columns and the last two rows from the matrix in (12) gives an optimal binary [12,5,(5,5,5,5,4)] LUEP code.

Construction N. Application of Construction 1 of Section 1.2 with $m = 1, q = 2$ and G_1 a generator matrix of the [7,4,(3,3,3,3)] Hamming code gives an optimal binary [14,5,(7,6,6,6,6)] LUEP code.

Construction O. The 6-by-15 matrix

$$\begin{bmatrix} 000011111000111 \\ 000000000111111 \\ 100011000100100 \\ 010010100010010 \\ 001010010001001 \\ 000100001001001 \end{bmatrix} \tag{13}$$

is a generator matrix of an optimal binary [15,6,(7,6,5,5,5,4)] LUEP code.

Construction P. The 7-by-15 matrix

$$\begin{bmatrix} 000001111111000 \\ 000001110000111 \\ 100001001000100 \\ 010001001000010 \\ 001000100100001 \\ 000100100010001 \\ 000010100001001 \end{bmatrix} \tag{14}$$

is a generator matrix of an optimal binary [15,7,(7,6,4,4,4,4,4)] LUEP code.

Construction Q. By deleting the 8th column from the matrix in (14) we obtain a generator matrix of an optimal binary [14,7,(6,5,4,4,4,4,4)] code.

Construction R. The 8-by-15 matrix

$$\left[\begin{array}{c|c} \begin{matrix}0\\0\\0\\0\\0\\0\\0\end{matrix} & G \\ \hline 1 & 000001000100000 \end{array}\right], \tag{15}$$

where G is the matrix in (12), is a generator matrix of an optimal binary [15,8,(5,5,5,5,4,4,4,3)] LUEP code.

Construction S. The 8-by-15 matrix

$$\left[\begin{array}{l} 100000001100110 \\ 010000001010101 \\ 001000001001111 \\ 000100000110100 \\ 000010001110000 \\ 000001001101000 \\ 000000101011000 \\ 000000010111000 \end{array}\right] \tag{16}$$

is a generator matrix of an optimal binary [15,8,(5,5,5,4,4,4,4,4)] LUEP code.

Reference

[1] L.M.H.E. Driessen, "On an Infinite Series of [4n,2n] Binary Codes", *IEEE Trans. Inform. Theory*, vol. IT-30, no. 2, pp. 392-395, March 1984.

2.

Two-dimensional dot codes for product identification

Wil J. van Gils

Abstract

In automated manufacturing the two-dimensional representation of product identification numbers by means of square dot codes offers a number of advantages over the use of bar codes. In this paper a method is described for the encoding of product identification numbers into square matrices of round dots on a contrasting background. It consists of an optimal source coding scheme for the encoding of identification numbers into channel message words, and a channel coding scheme offering error-detection and error-correction for random bit errors (dot corruptions). For the channel coding scheme, a special class of error-correcting codes is introduced, called *square-cyclic codes*. It is shown how to transform quasi-cyclic and (shortened) cyclic codes into square-cyclic codes.

I. Introduction

The widespread use of bar codes in automated manufacturing clearly shows the need for an automatically readable product identification code. A bar code is built up from a number of parallel bars. The relative widths and mutual distances of these bars determine the meaning of the bar code.

We believe, however, that dot codes provide a better alternative to bar codes in this area of technology. A dot code consists of a square matrix of round dots on a contrasting background. The meaning of the dot code is determined by the absence or presence of dots. In a dot code the information is recorded in a two-dimensional way, whereas in a bar code only one direction is used to encode information. This difference enables the dot code to offer higher information density, thereby allowing smaller product identification areas (see Figure 1). For example, at the

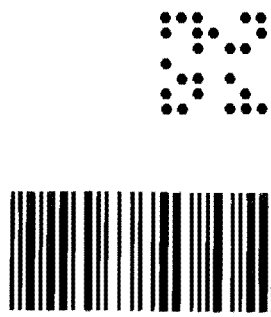


Figure 1: *A comparison of a 7 by 7 dot code and a standard 'code 10' bar code with the same information capacity. The diameter of the dots has been chosen equal to the linewidth of the wide black bars in the bar code. Both codes have been printed with a dot-addressable matrix printer.*

flat top of an electric motor shaft there is not enough room for a bar code. Furthermore, in automated manufacturing it is easy

to write the dot codes onto the mechanical parts by an engraving process. With bar codes this would be more complicated. The dot codes can be read by a standard TV camera and can be recognized by a relatively inexpensive picture processing system.

Therefore, we shall introduce a method for the transmission of numbers from one point to another point by means of square matrices of round dots. These square dot matrices can be translated into square binary matrices by representing the presence of a dot by a one (1) and the absence of a dot by a zero (0). In a practical situation it was observed that only random dot corruptions (causing random bit errors) occurred in the dot squares. These errors were due to printing imperfections, dust particles, and reading failures. Furthermore, because of the possibly random rotation of the mechanical parts during the manufacturing process, recognition of the dot matrices should be possible irrespective of the orientation of the matrices. For example, one should again think of a square dot matrix on the flat top of a rotated shaft of an electric motor, without any synchronization indication outside the dot matrix.

In this paper we give a possible solution to this transmission problem. To this end, we distinguish the following steps in the transmission process.

1. **Source encoding**: the encoding of the source message set (the integers $i, 0 \leq i \leq M-1$) into the channel message set (a subset of the set of all binary vectors of length k).

2. **Channel encoding**: the encoding of channel message words into codewords (square binary matrices) of a binary linear $[n, k]$ code such that random bit errors can be corrected and detected, and such that rotation of a codeword does not cause a decoding error.

3. **Transmission**: the actual transmission of the dot matrix through the noisy channel. During transmission dot matrices can also be rotated over 90, 180 and 270 degrees.

4. **Channel decoding**: the decoding of a rotated, corrupted codeword into an estimate for the transmitted, rotated channel message word.

5. **Source decoding:** the decoding of the estimate for the rotated channel message word into an estimate for the source message.

This scheme is illustrated in Figure 2.

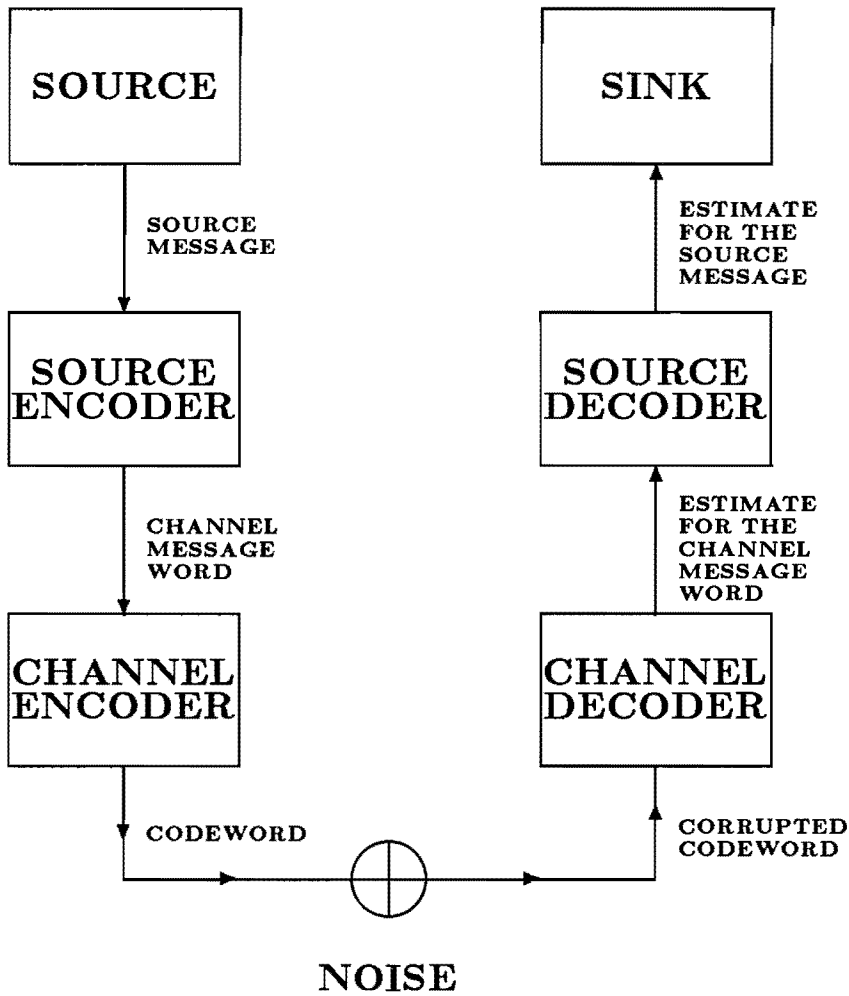


Figure 2: *The transmission scheme of square binary matrices.*

We shall start by considering the structure the error-correcting code should have. This will result in the definition of the class of so-called *square-cyclic codes.* In Section III we describe two source coding schemes that nicely fit the concept of square-cyclic codes. One is the optimal one, in the sense that it uses the minimum number of bits to encode a source message into a channel message word. The other scheme is not optimal, but gives rise to a very simple and fast encoding/decoding algorithm.

In general it is not possible to transform a square-cyclic generator matrix into a systematic square-cyclic generator matrix. However, it is possible to transform it into a *canonical* square-cyclic generator matrix that makes transformation of a codeword into its corresponding channel message word relatively easy. This canonical square-cyclic generator matrix is described in Section IV. In Section V we give the construction of square-cyclic codes from well-known quasi-cyclic and (shortened) cyclic codes respectively. In fact a square-cyclic code is a quasi-cyclic or an extended quasi-cyclic code itself. In this paper no new error-correcting codes are introduced, but a number of well-known, well-structured codes are transformed such that they can be used in our particular application.

II. Definition of square-cyclic codes

If we use an arbitrary t-error-correcting code on our channel, then the decoder has to deal with the corruption as well as with the rotation of the transmitted codeword. One possibility for the decoder is to apply the decoding algorithm to all four rotations of the received word and check which one gives a codeword. In this case it is very possible that among the four estimates of a received word with at most t errors in it there is more than one codeword. Furthermore, in this approach the decoding algorithm is applied four times for each received word. Hence, it seems useful that we choose a linear error-correcting code, whose codewords have the form of square binary matrices, such that rotations over 90, 180 and 270 degrees of any codeword again give codewords of the

code. In that case we can first correct the errors in the corrupted and rotated codeword, and afterwards we can possibly deal with the rotation.

This approach implies that a codeword together with its rotations over 90, 180 and 270 degrees should represent the same channel message word. Hence, the number of channel message words is less than or equal to the number of equivalence classes of the relation defined by the rotation of codewords in the above-mentioned code. We come back to this source coding problem in Section III.

We next define the class of codes, called *square-cyclic codes*, having the property that a rotation of any codeword of such a code is again a codeword of that code.

Definition 1. A binary linear $[n,k]$ code C is called *square-cyclic* if it is possible to transform the codewords of C into square m-by-m matrices for some $m, m^2 \geq n$, allowing empty positions, such that the rotations of any codeword in square matrix form over 90, 180, and 270 degrees again give codewords of the code C in square matrix form.

Example 1. Consider the [48,24] binary linear code C with generator matrix

$$\begin{bmatrix} I & O & O & O & O & O & A & B & C & D & E & F \\ O & I & O & O & O & O & PF & A & B & C & D & E \\ O & O & I & O & O & O & PE & PF & A & B & C & D \\ O & O & O & I & O & O & PD & PE & PF & A & B & C \\ O & O & O & O & I & O & PC & PD & PE & PF & A & B \\ O & O & O & O & O & I & PB & PC & PD & PE & PF & A \end{bmatrix},$$

where all submatrices are 4-by-4 matrices, I is the identity matrix, O is the all-zero matrix, and A, B, C, D, E, F, P are 4-by-4 circulant matrices with first rows (1011), (1101), (0101), (1110), (0100), (1011), (0100), respectively. The 24 information bits of a codeword $\underline{c}$ are numbered by $m_0, m_1, \ldots, m_{23}$, the 24 parity bits are numbered by $p_0, p_1, \ldots, p_{23}$, so

$$\underline{c} = (m_0, m_1, \ldots, m_{23}, p_0, p_1, \ldots, p_{23}).$$

These 48 bits are put into a 7-by-7 square according to the scheme shown in Figure 3. The middle point is not used in this case. From Figure 3 we see that the information bits are put at the border of the square, and the check bits are put in the inner 5-by-5 square.

m_0	m_{23}	m_{19}	m_{15}	m_{11}	m_7	m_3
m_4	p_0	p_{15}	p_{11}	p_7	p_3	m_{22}
m_8	p_4	p_{16}	p_{23}	p_{19}	p_{14}	m_{18}
m_{12}	p_8	p_{20}		p_{22}	p_{10}	m_{14}
m_{16}	p_{12}	p_{17}	p_{21}	p_{18}	p_6	m_{10}
m_{20}	p_1	p_5	p_9	p_{13}	p_2	m_6
m_1	m_5	m_9	m_{13}	m_{17}	m_{21}	m_2

Figure 3: *The distribution of message bits and check bits in a square matrix for the* $[48, 24]$ *code.*

Of course, other distributions are also possible. The code has minimum Hamming distance 12. Its construction will be given in Section V together with an elegant decoding algorithm. From the special structure of the generator matrix and the mapping of the code bits into 7-by-7 square matrices according to the scheme in Figure 3, it is easy to see that the rotations of a codeword in C (in square matrix form) over 90, 180 and 270 degrees again give codewords in C. For an example, see Figure 4.

1010011	1111100
1000101	1001011
1001101	0111110
111x111	001x111
1111101	1001100
0101110	0001111
0101010	1111100

Figure 4: *A codeword of the* $[48, 24]$ *code and its rotation over 90 degrees.*

From Definition 1 it is immediately clear that the length n of a square-cyclic code should be a multiple of four ($n = 4s$) or a multiple of four plus one ($n = 4s+1$) if the middle position of the square is also used (that is only possible if m and n are odd). Without loss of generality we order the bits in codewords of a square-cyclic code such that positions that are mapped onto each other by rotations of the square matrices are grouped together into fourtuples. In other words, if the code bits $c_0, c_1, \ldots, c_{n-1}$ of a codeword $(c_0, c_1, \ldots, c_{n-1})$ are put in an m-by-m square matrix, where m is minimal such that $n \leq m^2$ and m is odd if n is odd, then if code bit c_{4f} is put in position (i, j), the code bits c_{4f+1},c_{4f+2},c_{4f+3} are put in the positions $(m+1-j, i)$, $(m+1-i, m+1-j)$, and $(j, m+1-i)$, respectively. Note that positions in a m-by-m square matrix are numbered by the pairs (i, j) where $i, j = 1, 2, \ldots, m$. Furthermore, if m and n are both odd, the last code bit c_{n-1} is put in the middle of the square matrix, i.e., in position $((m+1)/2, (m+1)/2)$. The four rotations of a codeword (in square matrix form) of a square-cyclic $[n, k]$ code give 4, 2 or 1 different codewords. For $i = 1, 2,$ and 4 define $C(i)$ to be the set of codewords $\underline{c}$ of C whose four rotations give at most i mutually different codewords. It is easy to see that $C(i), i = 1, 2, 4$ are linear subspaces of the code C. In particular $C(4)$ is equal to C. Let p,q, and r be (the unique) nonnegative integers such that the following equations hold,

$$\begin{aligned} k = dimC(4) &= 4p + 2q + r, \\ dimC(2) &= 2p + 2q + r, \\ dimC(1) &= p + q + r, \end{aligned}$$

where $dimC(i)$ denotes the dimension of $C(i)$. Hence the square-cyclic $[n, k]$ code C of length n and dimension k has a generator

matrix of the form

$$G = \begin{bmatrix} A_{11} & A_{12} & \dots & \dots & A_{1S} \\ A_{21} & A_{22} & \dots & \dots & A_{2S} \\ \vdots & \vdots & & & \vdots \\ A_{p1} & A_{p2} & \dots & \dots & A_{pS} \\ B_{11} & B_{12} & \dots & \dots & B_{1S} \\ \vdots & \vdots & & & \vdots \\ B_{q1} & B_{q2} & \dots & \dots & B_{qS} \\ C_{11} & C_{12} & \dots & \dots & C_{1S} \\ \vdots & \vdots & & & \vdots \\ C_{r1} & C_{r2} & \dots & \dots & C_{rS} \end{bmatrix}$$

if n is even, and the same matrix as above with an extra column

$$(\underline{d_1}, \underline{d_2}, \dots, \underline{d_p}, \underline{e_1}, \underline{e_2}, \dots, \underline{e_q}, \underline{f_1}, \dots, \underline{f_r})^T$$

added to it, if n is odd. The matrices $A_{11}, A_{12}, \dots, A_{1S}, A_{21}, \dots, A_{pS}$ are 4-by-4 circulant matrices, $B_{11}, B_{12}, \dots, B_{1S}, B_{21}, \dots, B_{rS}$ are 2-by-4 circulant matrices, $C_{11}, \dots, C_{1S}, C_{21}, \dots, C_{rS}$ are 1-by-4 circulant matrices, and $\underline{d_1}, \dots, \underline{d_p}$ are 4-by-1 all-zero or all-one row vectors, $\underline{e_1}, \dots, \underline{e_q}$ are 2-by-1 all-zero or all-one row vectors, and $\underline{f_1}, \dots, \underline{f_r}$ are 1-by-1 all-zero or all-one row vectors. Here a u-by-v matrix is called circulant if, for all $i = 1, 2, ..., u - 1$ row $(i + 1)$ is the cyclic shift of one position to the right of row i, and row 1 is the cyclic shift of one position to the right of row u. By this definition a 3-by-4 binary circulant matrix consists of twelve zeros or twelve ones; that is it is a repetition of three 1-by-4 binary circulant matrices. If u is a divisor (multiple) of v, then a u-by-v circulant matrix is a repetition of u-by-u (v-by-v) circulant matrices.

Of course parity-check matrices of square-cyclic codes also have an identical form. Generator and parity-check matrices of this form are called *square-cyclic generator and parity-check matrices*, respectively. The most interesting square-cyclic codes are the ones that fill a square matrix completely ($n = m^2$) or nearly completely if m is odd ($n = m^2 - 1$). It should be remarked that in certain applications some positions in a dot matrix, for example the corner dots, need to be reserved for other purposes.

III. Source encoding and decoding

Because we use a square-cyclic code on our channel, and because codewords can be rotated during transmission, it is not possible to attach different source messages to the four rotations of a particular codeword. A codeword and its rotations should all represent the same source message.

To a rotation of a codeword of a code with a generator matrix like the ones in Section II there corresponds a 'rotation' of the k-bit information vector associated with it. Let a k-bit ($k = 4p+2q+r$) information vector $\underline{m}$ be denoted by

$$\underline{m} = (\underline{u_1}, \underline{u_2}, \ldots, \underline{u_p}, \underline{v_1}, \underline{v_2}, \ldots, \underline{v_q}, \underline{w_1}, \underline{w_2}, \ldots, \underline{w_r}),$$

where the vectors $\underline{u_i}$ ($i = 1, \ldots, p$) are row vectors of length four, the vectors $\underline{v_i}$ ($i = 1, \ldots, q$) are row vectors of length two, and the vectors $\underline{w_i}$ ($i = 1, \ldots, r$) are row vectors of length one. So the partitioning of the message $\underline{m}$ depends on the parameters p,q and r, which are parameters determined by the channel code (Section II). By the rotation $R^i(\underline{m})$ of $\underline{m}$ over i positions is meant the cyclic shifts of the individual components $\underline{u_1}, \ldots, \underline{u_p}, \underline{v_1}, \ldots, \underline{v_q}, \underline{w_1}, \ldots, \underline{w_r}$ over i positions to the right simultaneously. So the rotation over one position of

$$\underline{m} = (1000, 0011, 10, 00, 11, 0, 1, 0)$$

is

$$R(\underline{m}) = (0100, 1001, 01, 00, 11, 0, 1, 0).$$

We define the relation $\sim$ on the set $\mathbf{F}_2^k$ of k-bit binary vectors by

$$\underline{m_1} \sim \underline{m_2} \;:\Leftrightarrow\; (R^i(\underline{m_1}) = \underline{m_2} \text{ for } i = 0, 1, 2, \text{ or } 3).$$

This relation is an equivalence relation. Its equivalence classes contain one, two or four elements. As we have seen above, the maximum size of the set of source messages equals the number of equivalence classes of the relation $\sim$ in $\mathbf{F}_2^k$. In the source encoder we add to any source message a representative of an equivalence class. This set of representatives is called the channel message set. In the channel encoder these channel message words are mapped

onto codewords of the error-correcting code. Because the set of channel message words has less than 2^k elements, the set of possible channel encoder outputs is not the entire code space, but only a part of it. However, due to the rotations during transmission, any codeword can appear as input of the channel decoder.

A k-bit vector $(\underline{u_1}, \ldots, \underline{u_p}, \underline{v_1}, \ldots, \underline{v_q}, \underline{w_1}, \ldots, \underline{w_r})$ is said to have period i whenever it is contained in an equivalence class having i elements. The set $\mathbf{F}_2^k$ contains 2^{p+q+r} elements of period one, $2^{2p+2q+r}$ elements of period at most two, and $2^{4p+2q+r}$ elements of period at most four. So, it contains $2^{2p+2q+r} - 2^{p+q+r}$ elements of period two and $2^{4p+2q+r} - 2^{2p+2q+r}$ elements of period four. Hence, the number of equivalence classes equals

$$\begin{aligned} M(p,q,r) &:= (2^{4p+2q+r} - 2^{2p+2q+r})/4 + (2^{2p+2q+r} - 2^{p+q+r})/2 \\ &\quad + 2^{p+q+r} \\ &= 2^{4p+2q+r-2} + 2^{2p+2q+r-2} + 2^{p+q+r-1}. \end{aligned}$$

Next we build a channel message set by choosing $M(p,q,r)$ representatives of the equivalence classes. Therefore we define the following sets:

$$\begin{aligned} \mathcal{P} &= \{(1000), (1100), (1110)\}, \\ \mathcal{Q} &= \{(0000), (1111)\}, \\ \mathcal{S} &= \{(1010)\}, \\ \mathcal{T} &= \{(1000), (0100), (1100), (0110), (1110), (0111)\}, \\ \mathcal{U} &= \{(1010), (0101), (0000), (1111)\}, \\ \mathcal{V} &= \{(10)\}, \\ \mathcal{W} &= \{(00), (11)\}. \end{aligned}$$

It is easy to see that we can take representatives whose first components $\underline{u_1}$ are in the set $\mathcal{P} \cup \mathcal{Q} \cup \mathcal{S}$. Therefore, our first rule requires all representatives to have a first component $\underline{u_1}$ in the set $\mathcal{P} \cup \mathcal{Q} \cup \mathcal{S}$. If $\underline{u_1}$ is in $\mathcal{P}$ this rule determines a unique representative for the corresponding equivalence class. If $\underline{u_1}$ is in $\mathcal{Q} \cup \mathcal{S}$ we could have more than one candidate. Additional rules are necessary to determine a unique representative.

In an equivalence class containing elements with first component $\underline{u_1}$ in $\mathcal{Q}$, at least one element has a second component $\underline{u_2}$ in $\mathcal{P} \cup \mathcal{Q} \cup \mathcal{S}$. In this case ($\underline{u_1}$ in $\mathcal{Q}$) our second rule is the same as

the first one, but now for the second component $\underline{u_2}$; i.e, $\underline{u_2}$ should be in $\mathcal{P} \cup \mathcal{Q} \cup \mathcal{S}$.

In an equivalence class that contains an element with first component $\underline{u_1}$ in $\mathcal{S}$, at least one element has a second component $\underline{u_2}$ in $\mathcal{T} \cup \mathcal{U}$. In this case ($\underline{u_1}$ in $\mathcal{S}$) our second rule requires a representative to have a second component $\underline{u_2}$ in $\mathcal{T} \cup \mathcal{U}$. If $\underline{u_2}$ is in $\mathcal{T}$ this rule determines a unique representative. If $\underline{u_2}$ is in $\mathcal{U}$ our third rule requires that the third component $\underline{u_3}$ of a representative is again in $\mathcal{T} \cup \mathcal{U}$ (of course only if $p \geq 3$).

The above process and a similar process for the components $\underline{v_j}$ $(j = 1, \ldots, q)$ is repeated until a unique representative is found for all equivalence classes. Formally the process is described by the following rules.

The set $\mathcal{F}$ of representatives is recursively defined by:

$$\begin{array}{lcll} \mathcal{F} & = & \bigcup_{\underline{u_1} \in \mathcal{P} \cup \mathcal{Q} \cup \mathcal{S}} \mathcal{F}(\underline{u_1}) & \text{for } p > 0, \\ \mathcal{F} & = & \bigcup_{\underline{v_1} \in \mathcal{V} \cup \mathcal{W}} \mathcal{H}(\underline{v_1}) & \text{for } p = 0, q > 0, \\ \mathcal{F} & = & \{0,1\}^r & \text{for } p = 0, q = 0, r > 0. \end{array}$$

For $1 \leq i < p$:

$$\begin{array}{lcll} \mathcal{F}(\underline{u_1}, \ldots, \underline{u_i}) & = & \{(\underline{u_1}, \ldots, \underline{u_i}, \underline{x}) : \underline{x} \in \{0,1\}^{4(p-i)+2q+r}\} & \text{for } \underline{u_i} \in \mathcal{P}, \\ \mathcal{F}(\underline{u_1}, \ldots, \underline{u_i}) & = & \bigcup_{\underline{u_{i+1}} \in \mathcal{P} \cup \mathcal{Q} \cup \mathcal{S}} \mathcal{F}(\underline{u_1}, \ldots, \underline{u_i}, \underline{u_{i+1}}) & \text{for } \underline{u_i} \in \mathcal{Q}, \\ \mathcal{F}(\underline{u_1}, \ldots, \underline{u_i}) & = & \bigcup_{\underline{u_{i+1}} \in \mathcal{T} \cup \mathcal{U}} \mathcal{G}(\underline{u_1}, \ldots, \underline{u_i}, \underline{u_{i+1}}) & \text{for } \underline{u_i} \in \mathcal{S}, \\ \mathcal{G}(\underline{u_1}, \ldots, \underline{u_i}) & = & \{(\underline{u_1}, \ldots, \underline{u_i}, \underline{x}) : \underline{x} \in \{0,1\}^{4(p-i)+2q+r}\} & \text{for } \underline{u_i} \in \mathcal{T}, \\ \mathcal{G}(\underline{u_1}, \ldots, \underline{u_i}) & = & \bigcup_{\underline{u_{i+1}} \in \mathcal{T} \cup \mathcal{U}} \mathcal{G}(\underline{u_1}, \ldots, \underline{u_i}, \underline{u_{i+1}}) & \text{for } \underline{u_i} \in \mathcal{U}. \end{array}$$

$$\begin{array}{lcll} \mathcal{F}(\underline{u_1}, \ldots, \underline{u_p}) & = & & \\ \mathcal{G}(\underline{u_1}, \ldots, \underline{u_p}) & = & \bigcup_{\underline{v_1} \in \mathcal{V} \cup \mathcal{W}} \mathcal{H}(\underline{u_1}, \ldots, \underline{u_p}, \underline{v_1}) & \text{for } q > 0, \\ \mathcal{F}(\underline{u_1}, \ldots, \underline{u_p}) & = & & \\ \mathcal{G}(\underline{u_1}, \ldots, \underline{u_p}) & = & \{(\underline{u_1}, \ldots, \underline{u_p}, \underline{z}) : \underline{z} \in \{0,1\}^r\} & \text{for } q = 0. \end{array}$$

For $1 \leq j < q$:

$$\mathcal{H}(\underline{u_1}, \ldots, \underline{u_p}, \underline{v_1}, \ldots, \underline{v_j})$$
$$\{(\underline{u_1}, \ldots, \underline{u_p}, \underline{v_1}, \ldots, \underline{v_j}, \underline{y}) : \underline{y} \in \{0,1\}^{2(q-j)+r}\} \text{ for } \underline{v_j} \in \mathcal{V}.$$

$$\mathcal{H}(\underline{u}_1, \ldots, \underline{u}_p, \underline{v}_1, \ldots, \underline{v}_j) = \bigcup_{\underline{v}_{j+1} \in \mathcal{V} \cup \mathcal{W}} \mathcal{H}(\underline{u}_1, \ldots, \underline{u}_p, \underline{v}_1, \ldots, \underline{v}_j, \underline{v}_{j+1}) \text{ for } \underline{v}_j \in \mathcal{W}.$$

$$\mathcal{H}(\underline{u}_1, \ldots, \underline{u}_p, \underline{v}_1, \ldots, \underline{v}_q) = \{(\underline{u}_1, \ldots, \underline{u}_p, \underline{v}_1, \ldots, \underline{v}_q, \underline{z}) \ : \ \underline{z} \in \{0,1\}^r\}.$$

From this definition of the channel message set it is easy to derive algorithms for the encoding of source messages into channel message words and for the decoding of channel message words into source messages. The encoding and decoding algorithms are shown in Figures 5 and 6 respectively. In these algorithms the following definitions are used.

Definition 2.

$$\begin{aligned} A(x,y,z) &:= 2^{4x+2y+z}, \\ C(x,y,z) &:= 2^{4x+2y+z-1} + 2^{2x+2y+z-1}, \\ M(x,y,z) &:= 2^{4x+2y+z-2} + 2^{2x+2y+z-2} + 2^{x+y+z-1}, \\ B(y,z) &:= 2^{2y+z}. \end{aligned}$$

For a binary vector $\underline{y} = (y_0, y_1, \ldots, y_{t-1})$, $bin(\underline{y})$ is defined by

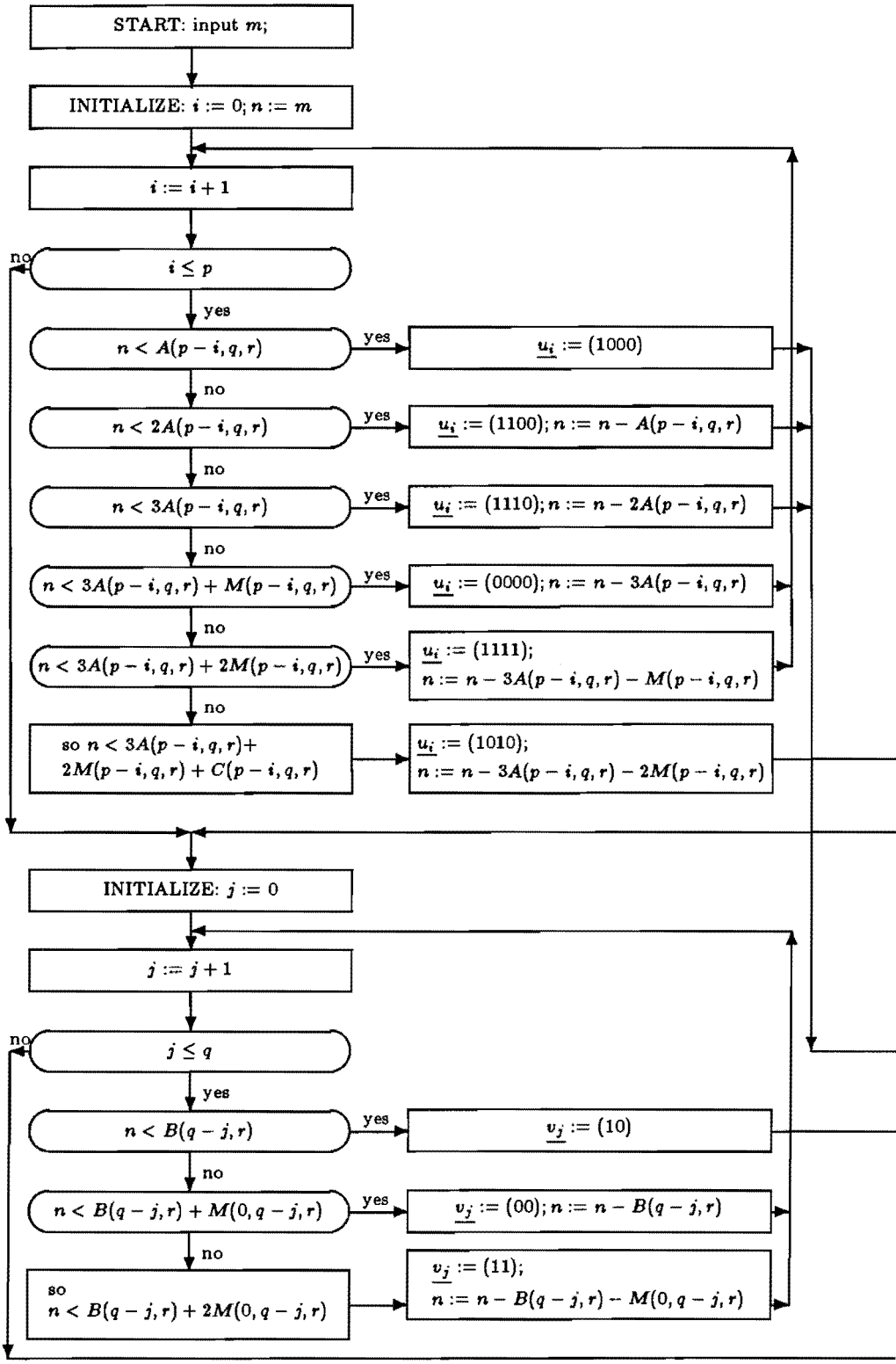
$$bin(\underline{y}) = \sum_{i=0}^{t-1} y_i 2^{t-1-i}.$$

For natural numbers s and t, such that $s < 2^t$, $invbin_t(s)$ is defined by

$$invbin_t(s) = (y_0, y_1, \ldots, y_{t-1}), \text{ where } s = \sum_{i=0}^{t-1} y_i 2^{t-1-i}.$$

For a binary vector $\underline{x} = (x_0, x_1, x_2, x_3)$ of length four, the function $f(\underline{x})$ is defined by

- $f(\underline{x})$:= the minimum number of positions $\underline{x}$ has to be shifted cyclically to the left such that it matches one of the following vectors: (1000), (1100), (1110), (0000), (1111), (1010), that are the elements of $\mathcal{P} \cup \mathcal{Q} \cup \mathcal{S}$,



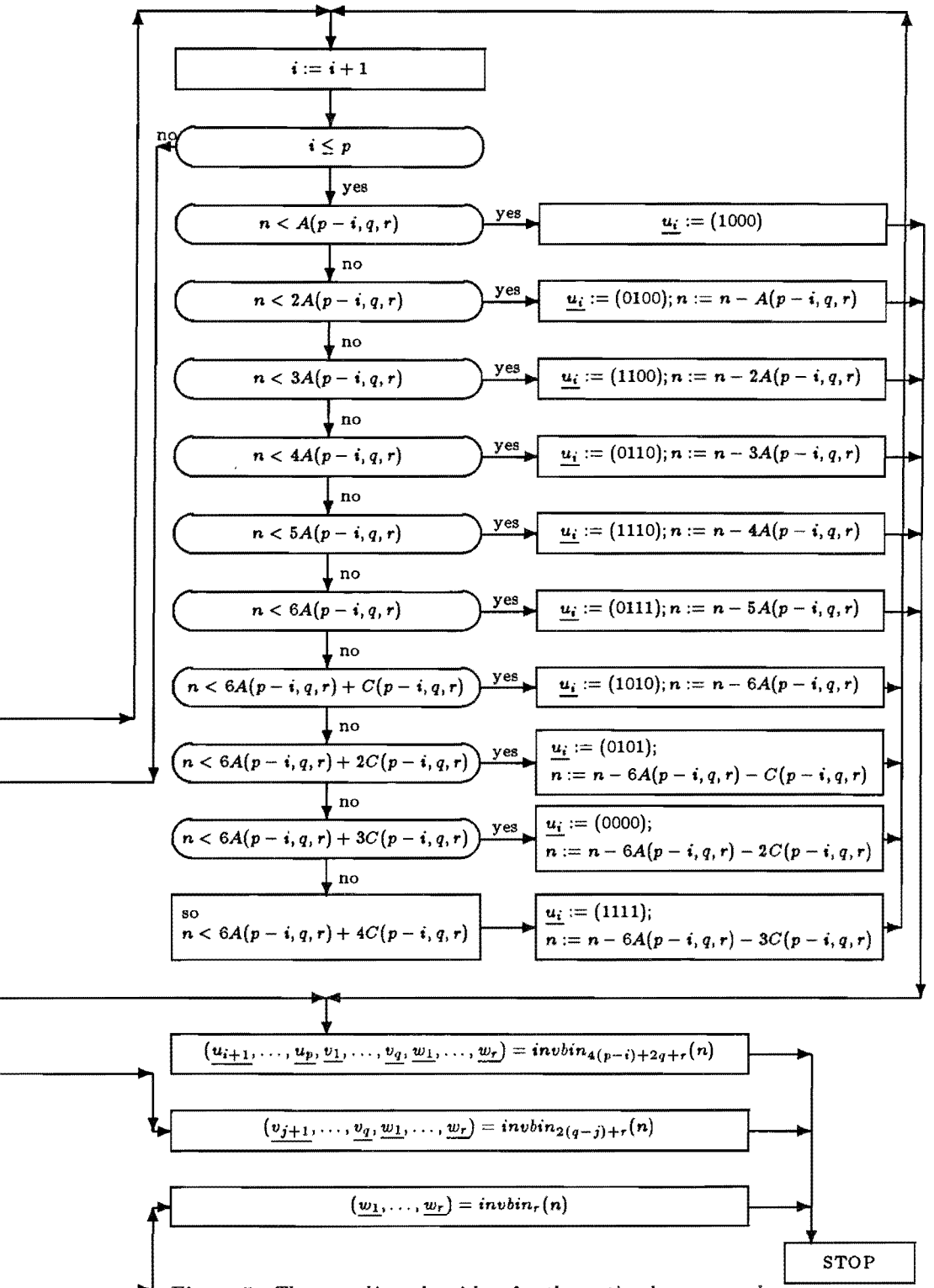
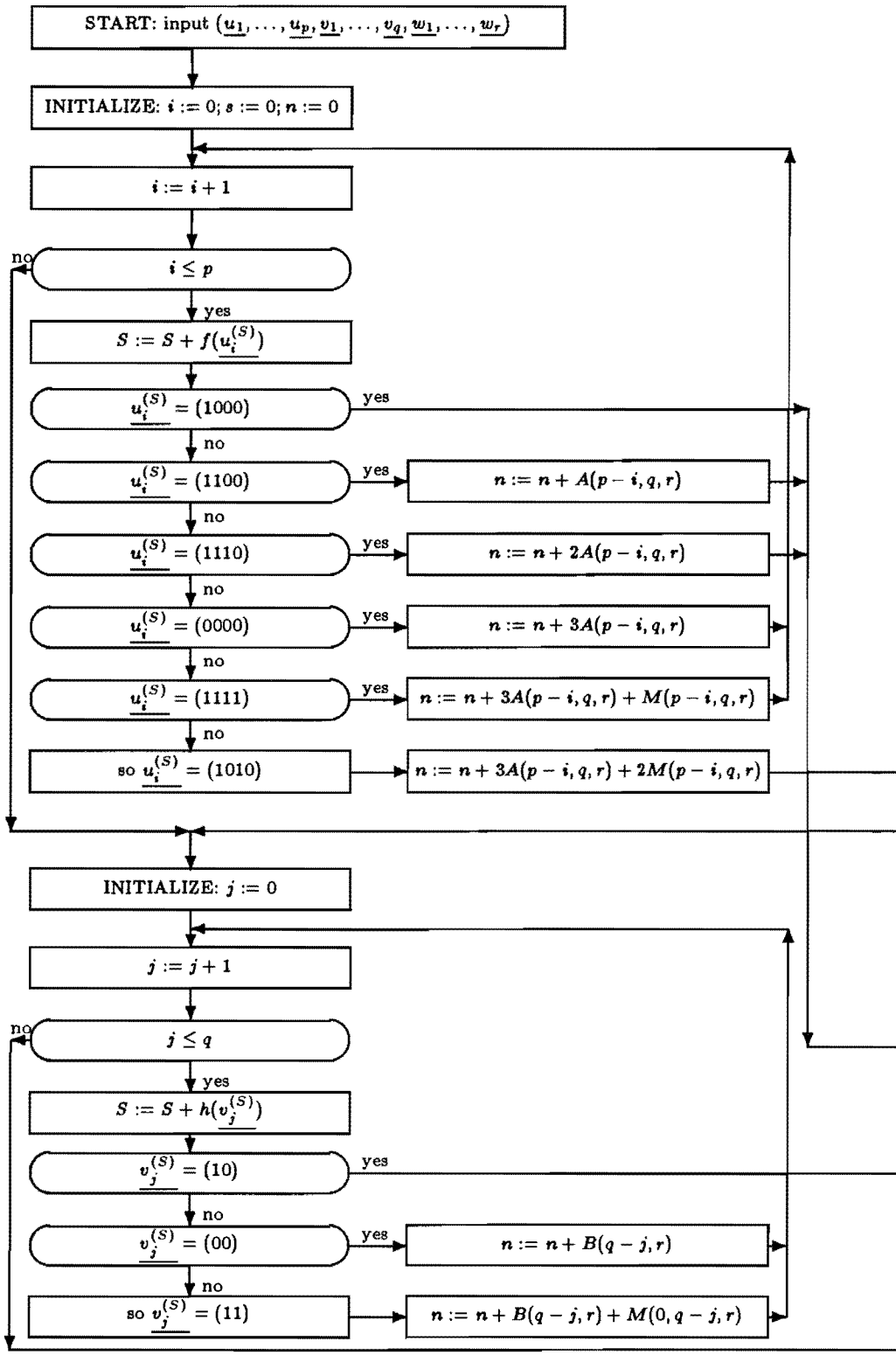


Figure 5: *The encoding algorithm for the optimal source code.*



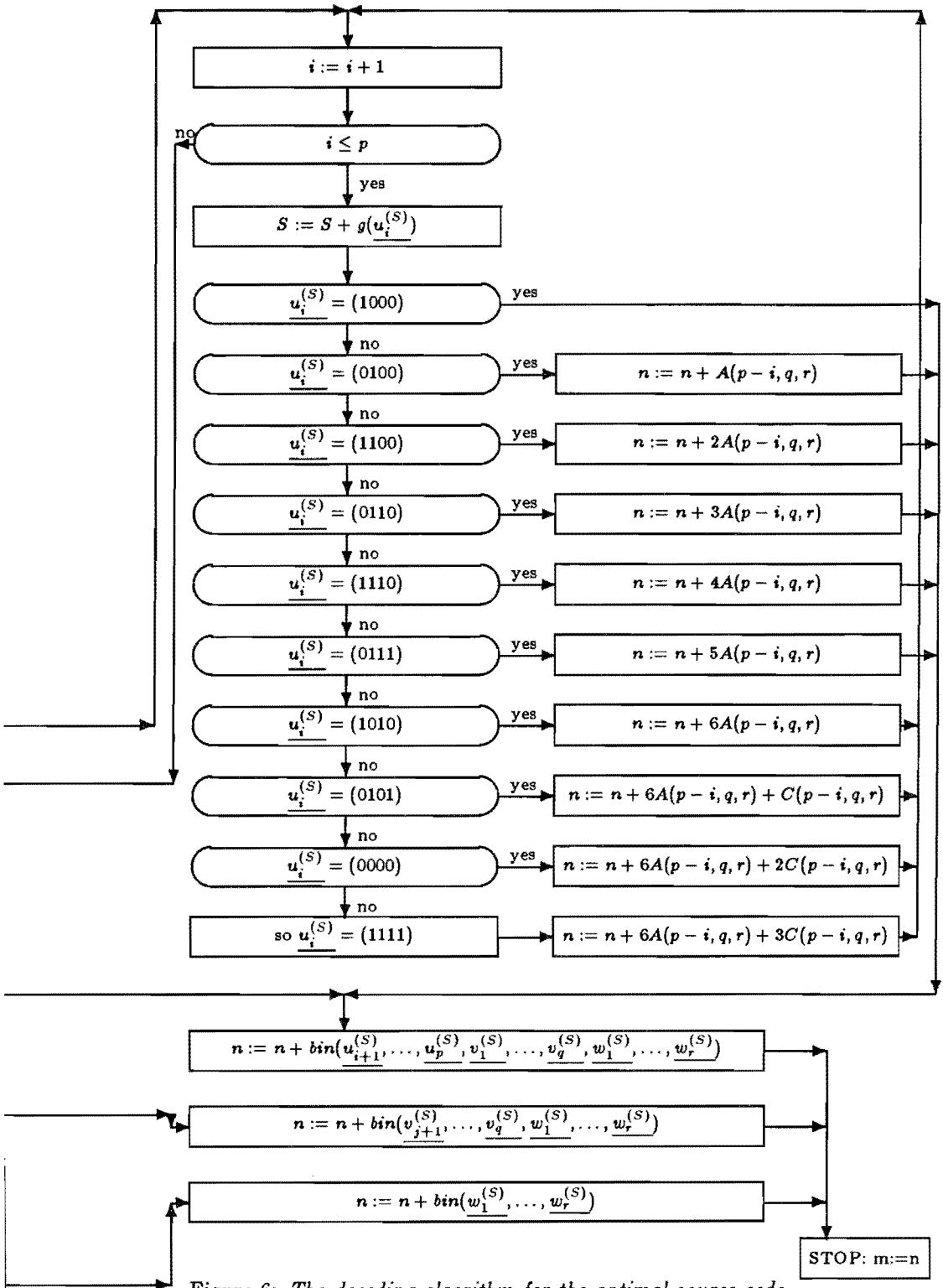


Figure 6: *The decoding algorithm for the optimal source code.*

and the function $g(\underline{x})$ by

- $g(\underline{x})$:= the minimum number of even positions $\underline{x}$ has to be shifted cyclically to the left such that it matches one of the following vectors: (1000), (0100), (1100), (0110), (1110), (0111), (0000), (1111), (1010), (0101), that are the elements of $\mathcal{T} \bigcup \mathcal{U}$.

For a binary vector $\underline{z} = (z_0, z_1)$ of length two, the function $h(\underline{z})$ is defined by

- $h(\underline{z})$:= the minimum number of positions $\underline{z}$ has to be shifted cyclically to the left such that it matches one of the following vectors: (00), (11), (10), that are the elements of $\mathcal{V} \bigcup \mathcal{W}$.

The vector $\underline{x^{(s)}}$ is defined as the result of shifting $\underline{x}$ s positions to the left cyclically.

Note that the numbers $A(.)$,$C(.)$,$M(.)$ and $B(.)$ correspond to the sizes of the sets $\mathcal{F}(.)$,$\mathcal{G}(.)$ and $\mathcal{H}(.)$ defined above. The input to the encoding algorithm is a source message m, $0 \leq m \leq M(p,q,r)-1$, and the output is a channel message word

$$(\underline{u_1}, \ldots, \underline{u_p}, \underline{v_1}, \ldots, \underline{v_q}, \underline{w_1}, \ldots, \underline{w_r}).$$

The input to the source decoding algorithm is a rotation of the channel message word, and the output is the corresponding source message.

We call the encoding and decoding schemes described above optimal, because they use the largest channel message set possible. In this scheme the encoding of source messages costs $4p + 2q + r - \log_2 M(p,q,r) < 2$ bits.

Example 2. In the [48,24] square-cyclic code defined above, $p = 6, q = r = 0$. So $M(p,q,r) = M(6,0,0) = 2^{22} + 2^{10} + 2^5 = 4,195,360$. For example, the source message 1,375,211 corresponds to the channel message

$$(1100, 0100, 1111, 0100, 0001, 1011),$$

and the source message 3,374,855 corresponds to the channel message

$$(0000, 1111, 1010, 0101, 0100, 0111).$$

Example 3. Consider a channel code with dimension $k = 24$ and suppose the corresponding threetuple (p, q, r) of this code equals (4,3,2). So, $M(p, q, r) = M(4, 3, 2) = 4,210,944$. For example, the source message 1,375,211 corresponds to the channel message

$$(1100, 0100, 1111, 1011, 11, 10, 10, 1, 1),$$

and the source message 3,376,658 corresponds to the channel message

$$(0000, 1111, 1010, 0101, 11, 00, 10, 1, 0).$$

If the number M of source messages is at most 2^{k-3}, or the encoding/decoding algorithms described above take too much time, then a simpler source coding scheme can be used. This encoding scheme adds at most three bits to a source message to build a channel message word. The encoding and decoding are as follows:

If $p > 0$:

Encoding: Set the first three bits of $\underline{u_1}$ equal to 1,0,0 and use the remaining $4p + 2q + r - 3$ bits of the channel message word for the $4p + 2q + r - 3$ source message bits.
Decoding: Rotate the channel message word until the first three positions of the channel message word are equal to 1,0,0. The remaining bits are a binary representation of the source message.

If $p = 0$ and $q > 0$:

Encoding: Set $\underline{v_1}$ equal to (10) and use the remaining $2q + r - 2$ bits of the channel message word for the $2q + r - 2$ source message bits.
Decoding: Rotate the channel message word until the first two positions of the channel message word are equal to 1,0. The remaining bits are a binary representation of the source message.

If $p = 0$ and $q = 0$, the channel message word is the binary representation of the source message.

So, in the most general case ($p > 0$), three bits are added to a source message to produce a channel message word.

IV. A canonical generator matrix of a square-cyclic code

In general it is necessary to multiply a part (k bits) of a codeword by a regular matrix to retrieve the corresponding information vector. When a systematic generator matrix for the code is used this is not necessary, the information vector being a part of the codeword. Unfortunately a square-cyclic code does not always have a square-cyclic systematic generator matrix. However, it does have another nicely structured square-cyclic generator matrix, the so-called *canonical form.* When this generator matrix is used, the transformation of codewords into information words does not need the multiplication by a k-by-k regular matrix. In this section we will show how to transform an arbitrary square-cyclic generator matrix into such a canonical form. It should be noted that a square-cyclic code of course also has a systematic generator matrix, but if it is not square-cyclic then there is no nice correspondence between the rotation of a codeword and the rotation of a channel message word.

Let G be a generator matrix of a square-cyclic code of length $4s$, G having the form as given in Section II. The 4-by-4 circulant matrices A_{ij} can be divided into six classes by considering their first rows. The first row of a 4-by-4 circulant matrix is a cyclic shift of one of the following six vectors: (0000), (1111), (1100), (1010), (1000), or (1110). The first four types yield singular matrices, the last two types yield regular matrices whose inverses are also circulant.

Now consider the generator matrix G. In the first four rows $(A_{11}, A_{12}, \ldots, A_{1S})$ at least one 4 by 4 circulant is regular, otherwise G would not have full rank. Without loss of generality, let us suppose that A_{11} is regular. By premultiplying G by the proper

regular matrix we obtain a matrix of the following form:

$$G' = \begin{bmatrix} I & D_{12} & \cdots & \cdots & \cdots & \cdots & \cdots & D_{1S} \\ O & D_{22} & \cdots & \cdots & \cdots & \cdots & \cdots & D_{2S} \\ \vdots & \vdots & & & & & & \\ O & D_{p2} & \cdots & \cdots & \cdots & \cdots & \cdots & D_{pS} \\ O & E_{12} & \cdots & \cdots & \cdots & \cdots & \cdots & E_{1S} \\ \vdots & \vdots & & & & & & \\ O & E_{q2} & \cdots & \cdots & \cdots & \cdots & \cdots & E_{qS} \\ O & F_{12} & \cdots & \cdots & \cdots & \cdots & \cdots & F_{1S} \\ \vdots & \vdots & & & & & & \\ O & F_{r2} & \cdots & \cdots & \cdots & \cdots & \cdots & F_{rS} \end{bmatrix},$$

where $I, O, D_{ij}, E_{ij}, F_{ij}$ are the 4-by-4 identity matrix, all-zero matrices, 4-by-4 circulants, 2-by-4 circulants, and 1-by-4 circulants, respectively. By repeating this process p times for successive four-tuples of rows, after a proper permutation of the columns, we get a generator matrix of the form

$$G'' = \begin{bmatrix} I & O & O & \cdots & O & K_{1\,p+1} & \cdots & \cdots & \cdots & K_{1S} \\ O & I & O & \cdots & O & K_{2\,p+1} & \cdots & \cdots & \cdots & K_{2S} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & & & & \vdots \\ O & O & O & \cdots & I & K_{p\,p+1} & \cdots & \cdots & \cdots & K_{pS} \\ O & O & O & \cdots & O & L_{1\,p+1} & \cdots & \cdots & \cdots & L_{1S} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & & & & \vdots \\ O & O & O & \cdots & O & L_{q\,p+1} & \cdots & \cdots & \cdots & L_{qS} \\ O & O & O & \cdots & O & M_{1\,p+1} & \cdots & \cdots & \cdots & M_{1S} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & & & & \vdots \\ O & O & O & \cdots & O & M_{r\,p+1} & \cdots & \cdots & \cdots & M_{rS} \end{bmatrix},$$

where the matrices K_{ij}, L_{ij}, and M_{ij} are 4-by-4, 2-by-4, and 1-by-4 circulants, respectively.

At least one of the 2-by-4 circulant matrices $L_{1\,p+1}, \ldots, L_{1S}$ has rank two; i.e., it has the form

$$\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

or

$$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix},$$

otherwise the generator matrix G would not have full rank. In an analogous way as is done above, the matrix G'' can be transformed into the matrix $G''' =$

$$\begin{bmatrix}
I & \cdots & O & X_{1\,p+1} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & X_{1S} \\
\vdots & \ddots & \vdots & \vdots & & & & & & & & \vdots \\
O & \cdots & I & X_{p\,p+1} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & X_{pS} \\
OO & \cdots & OO & II & \cdots & OO & Y_{1\,p+q+1} & \cdots & \cdots & \cdots & \cdots & Y_{1S} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & & & & & \vdots \\
OO & \cdots & OO & OO & \cdots & II & Y_{q\,p+q+1} & \cdots & \cdots & \cdots & \cdots & Y_{qS} \\
0000 & \cdots & 0000 & 0000 & \cdots & 0000 & 1111 & \cdots & 0000 & Z_{1\,p+q+r+1} & \cdots & Z_{1S} \\
\vdots & & \vdots & \vdots & & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0000 & \cdots & 0000 & 0000 & \cdots & 0000 & 0000 & \cdots & 1111 & Z_{r\,p+q+r+1} & \cdots & Z_{rS}
\end{bmatrix},$$

where X_{ij}, Y_{ij}, and Z_{ij} are 4-by-4, 2-by-4, and 1-by-4 circulant matrices, respectively. The matrix G''' is called a *canonical* generator matrix of the code C. For a square-cyclic code of length $4s + 1$ the procedure is similar.

We have shown that a square-cyclic generator matrix of a square-cyclic code can be transformed into a matrix which, after proper column permutations, has the above canonical form. By using a canonical generator matrix, it is easy to retrieve a channel message word from a codeword. If $q = r = 0$, then the matrix G''' is systematic, so it is easy to retrieve the information vector $(\underline{u_1}, \ldots, \underline{u_p}) = (\underline{c_1}, \ldots, \underline{c_p})$ from the codeword $(\underline{c_1}, \ldots, \underline{c_S})$, where $\underline{c_i}(i = 1, \ldots, s)$ has length four. If $q \neq 0$, then we have to subtract from the vector $(\underline{c_{p+1}}, \ldots, \underline{c_{p+q}})$ the contribution of the vector $(\underline{c_1}, \ldots, \underline{c_p})$ to obtain the vector $(\underline{v_1}, \underline{v_1}, \underline{v_2}, \underline{v_2}, \ldots, \underline{v_q}, \underline{v_q})$, i.e., two times the information part $(\underline{v_1}, \ldots, \underline{v_q})$. The case $r \neq 0$ is similar.

V. Construction of square-cyclic codes

In this section we will show how square-cyclic codes can be constructed by transformation of quasi-cylic and (shortened) cyclic codes.

In the following we shall frequently apply the possibility to transform any $(f*g)$-by-$(f*h)$ circulant matrix A into an $(f*g)$-by-$(f*h)$ matrix A' consisting of f-by-f circulants, by permuting the rows and columns of A. For example, this can be done by the following permutation of rows and columns:

- *for i=1,2,...,f, j=1,2,...,g, row (i-1)g+j is mapped onto row (j-1)f+i, and for i=1,2,...,f, j =1,2...,h, column (i-1)h+j is mapped onto column (j-1)f+i.*

By applying transformations of this kind, many nicely structured codes, such as (shortened) cyclic and quasi-cyclic codes, can be transformed into square-cyclic codes. Among them are many optimal codes; i.e., they have the maximum minimum distance for a binary linear code of the same length and dimension. Because of their (quasi-)cyclic structure, well-known decoding techniques can be used to decode these codes.

In Subsections A and B we consider the construction of square-cyclic codes from quasi-cyclic and (shortened) cyclic codes, respectively. In the following we shall frequently use the following permutations.

Definition 3. The permutation $P(fg, f)$ maps $(i-1)g+j$ onto $(j-1)f+i$ for $i = 1, 2, \ldots, f$ and $j = 1, 2, \ldots, g$. The permutation $REV(f)$ maps i onto $f+1-i$ for $i = 1, 2, \ldots, f$.

A. Construction of square-cyclic codes from quasi-cyclic codes

A code is called quasi-cyclic if there is some integer v such that every cyclic shift of a codeword by v places is again a codeword. So, after a proper permutation of the columns of a generator matrix of a quasi-cyclic code, we get a generator matrix which is composed of circulant matrices. We have seen that a circulant matrix can be transformed into a matrix composed of smaller circulants. If the sizes of the circulants in a generator matrix of a quasi-cyclic code are proper, then the generator matrix can be transformed into a square-cyclic generator matrix by applying permutations of its rows and columns.

In [1,5,6,10,11] quasi-cyclic codes are constructed whose generator matrices are composed of g-by-g circulants, where g is a multiple of four. Those circulants can thus be transformed into compositions of 4 by 4 circulants by applying the permutation $P(4g, 4)$ to the rows and columns. These constructions give square-cyclic codes with $p \neq 0, q = r = 0$.

The very elegant decoding methods for quasi-cyclic codes can be used in these cases. For example, the decoding method of double-circulant codes described by Karlin [6] gives rise to a very efficient decoding algorithm when decoding is performed by software in a microprocessor.

Example 4. Consider the double-circulant [48,24,12] quadratic residue (QR) code [5]. This code has a generator matrix $G = [I A]$, where I is the 24 by 24 identity matrix and A is the 24 by 24 circulant matrix with top row [5]

$$\underline{a} = (a_0, a_1, \ldots, a_{23}) = (110101011110100101111001).$$

By performing the permutation $P(24, 4)$ onto the rows and columns of I and A, we bring the generator matrix G into square-cyclic form. The resulting generator matrix was given in Example 1.

The very special structure of the [48,24] QR code (self-dual and double-circulant) makes decoding relatively easy. The first step in the decoding process is to apply the inverse of the permutation $P(24, 4)$ to the left and the right half of a received word.

This brings the code back to its original double-circulant form. For decoding of the double-circulant code we use the 'redundant' parity-check matrix

$$H = \begin{bmatrix} I & A \\ A^{-1} & I \end{bmatrix}$$

and the corresponding 'redundant' syndromes. For convenience, we represent vectors $\underline{b} = (\underline{b_1}, \underline{b_2})$, where $\underline{b_1}, \underline{b_2}$ have length 24, by the polynomial pair $(b_1(x), b_2(x))$, where $b_1(x)$ and $b_2(x)$ are polynomials of degree at most 23:

$$\underline{b_i} = (b_{i0}, b_{i1}, \ldots, b_{i23}),\ b_i(x) = \sum_{j=0}^{23} b_{ij} x^j \text{ for } i = 1, 2.$$

An error pattern is then represented by $(e_1(x), e_2(x))$, the corrupted received codeword by $(r_1(x), r_2(x))$, and the syndrome by $(s_1(x), s_2(x)) = (r_1(x) + r_2(x)\overline{a}(x), r_1(x)a(x) + r_2(x))$, where $a(x) := a_0 + a_1 x + a_2 x^2 + \cdots + a_{23} x^{23}$ and $\overline{a}(x) := x^{24} a(x^{-1})$. The following decoding algorithm, which is t-error-correcting, $(11 - t)$-error-detecting for $t \leq 5$, can easily be implemented by software in a microprocessor. In this algorithm one should skip the steps that start with $(t \geq i)$ if the desired correction capacity is less than i. The function $wt(.)$ denotes the Hamming weight function.

Decoding algorithm (t-error-correcting, $(11 - t)$-error-detecting):

1. Input $(r_1(x), r_2(x))$;
2. compute the syndrome

$$\begin{aligned} s_1(x) &= r_1(x) &&+ r_2(x)\overline{a}(x), \\ s_2(x) &= r_1(x)a(x) &&+ r_2(x); \end{aligned}$$

3. if $wt(s_1(x)) \leq t$ then $\hat{e}_1(x) := s_1(x)$ else,
4. if $wt(s_2(x)) \leq t$ then $\hat{e}_1(x) := 0$ else,

$(t \geq 2)$:

5. if for some $j = 0, 1, \ldots, 23$, $wt(s_1(x) + x^j \overline{a}(x)) \leq t - 1$ then $\hat{e}_1(x) := s_1(x) + x^j \overline{a}(x)$ else,

$(t \geq 3)$:

6. if for some $j = 0, 1, \ldots, 23$, $wt(s_2(x) + x^j a(x)) \leq t - 1$ then $\hat{e}_1(x) := x^j$ else,

$(t \geq 4)$:
7. if for some $i, j = 0, 1, \ldots, 23, i \neq j$,
$wt(s_1(x) + (x^i + x^j)\overline{a}(x)) \leq t-2$ then $\hat{e}_1(x) := s_1(x) + (x^i + x^j)\overline{a}(x)$
else,
$(t \geq 5)$:
8. if for some $i, j = 0, 1, \ldots, 23, i \neq j$,
$wt(s_2(x) + (x^i + x^j)a(x)) \leq t-2$ then $\hat{e}_1(x) = x^i + x^j$ else,
9. detection of an error.

What is consequently used in this algorithm is the fact that if the weight of the left or right half of a modified syndrome is small enough, then it is known where the errors can be found [7,Ch.16,Th.19,p.513]. The decoding algorithm above is similar to the one described in Karlin[6].

B. Construction of square-cyclic codes from shortened cyclic codes

In [4] it is shown that generator matrices and parity-check matrices of cyclic codes can be partitioned into circulant matrices. The way this is done is described below.

Let n be an odd integer, and let b be minimal such that n divides $2^b - 1$. Let $\beta \in GF(2^b)$ be a primitive nth root of unity. Let $\{C_i : i \in \mathcal{A}\}$, where $\mathcal{A}$ denotes a set of indices, be the set of cyclotomic cosets modulo n. Now, consider a binary cyclic code C of length n having zeros $\gamma_i = \beta^i$, $i \in \bigcup_{j \in Z} C_j$, where Z is a subset of $\mathcal{A}$.

If we number the positions in a codeword by the nth roots of unity $\beta^0, \beta_1, \ldots, \beta^{n-1}$ such that conjugates of elements occupy adjacent positions in the order of increasing powers of two, the parity-check matrix H of C has the following form:

$$\mathrm{H} = \begin{bmatrix} 1 & \gamma_i^d & \gamma_i^{2d} & \gamma_i^{4d} & \ldots & \gamma_i^e & \gamma_i^{2e} & \gamma_i^{4e} & \ldots & \ldots & \gamma_i^f & \gamma_i^{2f} & \gamma_i^{4f} & \ldots \\ 1 & \gamma_j^d & \gamma_j^{2d} & \gamma_j^{4d} & \ldots & \gamma_j^e & \gamma_j^{2e} & \gamma_j^{4e} & \ldots & \ldots & \gamma_j^f & \gamma_j^{2f} & \gamma_j^{4f} & \ldots \\ \vdots & \vdots & & & & \vdots & & & & \vdots & & & & \\ 1 & \gamma_k^d & \gamma_k^{2d} & \gamma_k^{4d} & \ldots & \gamma_k^e & \gamma_k^{2e} & \gamma_k^{4e} & \ldots & \ldots & \gamma_k^f & \gamma_k^{2f} & \gamma_k^{4f} & \ldots \end{bmatrix},$$

where $d, e, \ldots, f$ are representatives of the cyclotomic cosets modulo n and $i, j, \ldots, k$ are in Z.

Every Galois field has a normal basis [7, page 122]. Let $\alpha, \alpha^2, \alpha^4, \ldots, \alpha^{2^{b-1}}$ be a normal basis of $GF(2^b)$ with respect to $GF(2)$. Now, we map the parity-check matrix H onto its binary image H_{bin} with respect to the abovementioned normal basis. This is done in the following way. Every component in H is replaced by its binary representation (being a column vector of length b) with respect to the normal basis $\alpha, \alpha^2, \alpha^4, \ldots, \alpha^{2^{b-1}}$. From the resulting matrix all rows occurring more than once are deleted. The resulting matrix H_{bin} is a full rank parity-check matrix of the code C, and it is composed of circulant matrices.

$$H_{bin} = \begin{bmatrix} 1 & & & & \\ \vdots & & & & \\ \vdots & A_{gd} & A_{ge} & \ldots\ldots & A_{gf} \\ \vdots & & & & \\ 1 & & & & \\ & & & & \\ 1 & & & & \\ \vdots & & & & \\ \vdots & A_{hd} & A_{he} & \ldots\ldots & A_{hf} \\ \vdots & & & & \\ 1 & & & & \\ & & & & \\ 1 & & & & \\ \vdots & \vdots & \vdots & \vdots\vdots & \vdots \\ 1 & & & & \\ 1 & & & & \\ \vdots & & & & \\ \vdots & A_{id} & A_{ie} & \ldots\ldots & A_{if} \\ \vdots & & & & \\ 1 & & & & \end{bmatrix},$$

in which the rows are numbered in the same way as the columns. The size of a cyclotomic coset modulo n is a divisor of b. The

matrix A_{gd} is a repetition of circulant matrices of row and column size minimum $\{|C_g|, |C_d|\}$ (for a set A, $|A|$ denotes the number of elements in A).

We shall transform a number of parity-check matrices of shortened cyclic codes such that the codes become square-cyclic. A set of rows numbered by the elements of a cyclotomic coset modulo n will be called a row cluster. A set of columns numbered by the elements of a cyclotomic coset modulo n will be called a column cluster.

It is immediately clear that, if we want to construct square-cyclic codes from cyclic codes by the abovementioned method, the number of elements in at least a few cyclotomic cosets modulo n have to be multiples of four. The number of elements in a coset modulo n is a divisor of b , so four has to be a divisor of b.

We now consider all binary cyclic codes of length $n, n \leq 99$, such that $b := min\{a : n \; is \; a \; divisor \; of \; 2^a - 1\}$ is a multiple of four. The lengths, dimensions, and minimum distances of these codes are listed in [8,Appendix D] and [9]. We also consider the binary cyclic codes of length 255.

Table I shows how the parity-check matrices H_{bin} of the abovementioned cyclic codes can be transformed into square-cylic parity-check matrices. Table I is divided into seven parts numbered by the values of b. The second column, with heading 'n_{cycl}', gives the lengths of the cyclic codes we consider. The third column (heading '$n_{sq\ cycl}$') gives the lengths of the square-cyclic codes derived from them. For example, for $b = 8$ we consider the cyclic codes of lengths 51, 85, and 255. The cyclic codes of lengths 51, 85, and 255 are transformed into square-cyclic codes of lengths 49, 81, and 241, respectively. How this is done can be seen in the last three columns. Column four (heading "size cc") gives the sizes of the cyclotomic cosets modulo $(2^b - 1)$ that are of interest for the codes we consider. For example, the cyclotomic cosests modulo $2^8 - 1$ have sizes 1,2,4, and 8. Column five (heading 'rows') and column six (heading 'columns') provide the transformations of the parity-check matrices. If in the sixth column of Table I there is the word 'del' in a row corresponding to a cyclotomic coset size i, it means that all columns in column clusters of size i should be deleted from the parity-check matrix of the above form H_{bin}. If in

b	n_{cycl}	$n_{sq\ cycl}$	size cc	rows	columns
4	15	13	1		
			2		del
			4		
8	51	49	1		
	85	81	2		del
	255	241	4	$P(4,2)$	del
			8	$P(8,4)$	$P(8,4)$
12	35	29	1		
	39	37	2		del
	45	36	3		del
	65	65	4	$REV(4)$	$REV(4)$
	91	85	6	$P(6,2)$	del
			12	$P(12,4)$	$P(12,4)$
20	41	41	1		
	55	45	2		del
	75	73	4		
			10	$P(10,2)$	del
			20	$P(20,4)$	$P(20,4)$
28	87	85	1		
			2		del
			28	$P(28,4)$	$P(28,4)$
36	95	77	1		
			4		
			18	$P(18,2)$	del
			36	$P(36,4)$	$P(36,4)$
48	97	97	1		
			48	$P(48,4)$	$P(48,4)$

Table I: *Construction parameters for square-cyclic codes from cyclic codes*

the fifth (sixth) column of Table I, there is a permutation in a row corresponding to a cyclotomic coset size i, it means that in the parity-check matrices of the above form H_{bin} the rows (columns) in all row (column) clusters of size i should be permuted according to that permutation. For example, consider the cyclic codes of length 85. The corresponding value of b is 8. There exist ten cyclotomic cosets modulo 85 containing eight elements, one containing four elements, none containing two elements and one containing one element. According to Table I, from the parity-check matrices of the cyclic codes of length 85, all columns in column clusters of size four should be deleted. This reduces the length of the codes to 81. Next, in the resulting parity-check matrices the rows and columns in row and column clusters of size eight should be permuted according to the permutation $P(8,4)$. Furthermore, in all parity-check matrices the rows in the row cluster of size four should be permuted according to the permutation $P(4,2)$. The resulting parity-check matrices are square-cyclic parity-check matrices of square-cyclic codes of length 81.

For the decoding of square-cyclic codes constructed in this subsection, the many well-known decoding algorithms for (shortened) cyclic codes can be used. Shortening a square-cyclic code in a proper way again gives a square-cylic code. So by shortening the codes constructed above until their lenghts are squares or squares minus one, we get square-cyclic codes that give completely filled square dot matrices.

Conclusion

We have described the transmission of information by means of two-dimensional matrices of dots on a contrasting background. In the automated manufacturing area, for example, product identification with dot codes is considered to be a good or even better alternative to product identification with bar codes.

The dot code transmission scheme is partitioned into two subschemes, the source coding scheme and the channel coding scheme. Due to the fact that dot matrices can be rotated during transmis-

sion, certain measures in the source and channel coding schemes are taken. To deal with these rotations in the source encoding, we have described two source encoding methods. These source encoding schemes, one in a very simple way and the other in the optimal way, encode the source messages (binary numbers) into channel message words (binary numbers with special structures of somewhat larger length). These channel message words are encoded into codewords in the channel encoder. Because we have to deal with the rotation of codewords and the corruption by errors simultaneously, we use a special class of error-correcting codes, called square-cylcic codes. Noise is caused by printing imperfections, dust particles and reading failures. A large number of well-known codes, such as quasi-cyclic and (shortened) cyclic codes, were transformed into square-cyclic codes. Those codes have the advantage that for them a large class of elegant decoding methods is known. Finally, we want to remark that it is also possible to transform (shortened) cyclic Unequal Error Protection (UEP) codes [2] into square-cyclic UEP codes and that the results in this paper can easily be extended to dot code patterns in the form of regular n-gons, where n = 3,5,6,7...

Acknowledgment

I wish to thank C.P.M.J. Baggen, G.F.M. Beenker, J.W. Brands, C.J.L. van Driel and L.M.G.M. Tolhuizen for their helpful suggestions during the preparation of this paper. Special thanks are due to C.P.M.J. Baggen for the fruitful cooperation that resulted in the discovery of an optimal source coding scheme.

References

[1] C.L. Chen, W.W. Peterson and E.J. Weldon, "Some results on quasi-cyclic codes", *Inform. Contr.*, vol. 15, pp. 407-423,

1969.

[2] W.J. van Gils, "Two topics on linear unequal error protection codes: bounds on their length and cyclic code classes", *IEEE Trans. Inform. Theory*, vol. IT-29, no. 6, pp. 866-876, Nov. 1983.

[3] H.J. Helgert and R.D. Stinaff, "Minimum-distance bounds for binary linear codes", *IEEE Trans. Inform. Theory*, vol. IT-19, no. 3, pp. 344-356, May 1973.

[4] C.W. Hoffner,II and S.M. Reddy, "Circulant bases for cyclic codes", *IEEE Trans. Inform. Theory*, vol. IT-16, no. 4, pp. 511-512, July 1970.

[5] R.A. Jenson, "A double circulant presentation for quadratic residue codes", *IEEE Trans. Inform. Theory*, vol. IT-26, no. 2, pp. 223-227, March 1980.

[6] M. Karlin, "Decoding of circulant codes", *IEEE Trans. Inform. Theory*, vol. IT-16, no.6, pp. 797-802, Nov. 1970.

[7] F.J. MacWilliams and N.J.A. Sloane, *The Theory of Error-Correcting Codes*, Amsterdam: North-Holland 1978.

[8] W.W. Peterson and E.J. Weldon, *Error-Correcting Codes*, 2^{nd} ed., Cambridge,MA: MIT Press 1972.

[9] G. Promhouse and S.E. Tavares, "The minimum distance of all binary cyclic codes of odd lengths from 69 to 99", *IEEE Trans. Inform. Theory*, vol. IT-24, no.4, pp. 438-442, July 1978.

[10] S.E. Tavares, V.K. Bhargava and S.G.S. Shiva, "Some rate-p/(p+1) quasi-cyclic codes", *IEEE Trans. Inform. Theory*, vol. IT-20, no. 1, pp. 133-135, Jan. 1974.

[11] H. van Tilborg, "On quasi-cyclic codes with rate 1/m", *IEEE Trans. Inform. Theory*, vol IT-24, no. 5, pp. 628-630, Sept. 1978.

3.1

A triple modular redundancy technique providing multiple-bit error protection without using extra redundancy

Wil J. van Gils

Abstract

A well-known technique for providing tolerance against single hardware component failures is triplication of the component, called Triple Modular Redundancy(TMR). In this paper a component is taken to be a processor-memory configuration where the memory is organized in a bit-sliced way. If voting is performed bitwise in an orthodox TMR configuration consisting of three of these components, failure of a complete component or failure of bit-slices not on corresponding positions in the memories can be tolerated. We present a TMR technique, not using more redundancy than orthodox TMR, that can tolerate the failure of arbitrary bit-slices (including those on corresponding positions) up to a certain amount. Additionally it can tolerate the failure of arbitrary bit-slices up to a certain amount whenever one component is known to be malfunctioning or whenever one component is disabled. This generalized TMR technique is described for processor-memory configurations processing 4-, 8-, and 16-bit words respectively.

I. Introduction

Consider a configuration of a k-bit processor-memory pair (see Figure 1). We shall call this a module and we shall use the word 'symbol' for a k-bit word. We suppose the memory to be organized in a bit-sliced way.

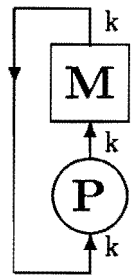


Figure 1:*A k-bit processor-memory configuration; P = processor, M = memory.*

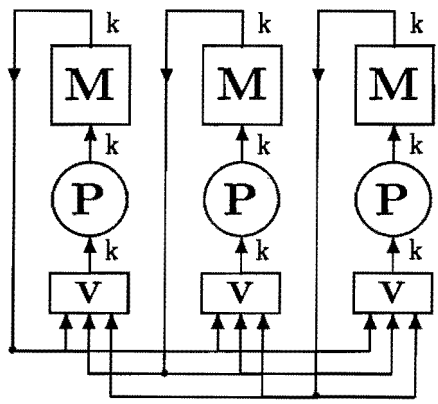


Figure 2: *An orthodox TMR configuration; V = voter.*

A frequently used technique [13] for providing tolerance against

failures of such a module is triplication of the module, called Triple Modular Redundancy (TMR, see Figure 2). The input for the processors is the majority vote over the outputs of the memories. Hence, a so-called symbol error caused by the failure of one module is outvoted. When this voting is done bitwise, multiple bit errors not occurring in corresponding positions in the memories are outvoted as well. In the following we shall refer to this as the orthodox TMR configuration.

In practice, it was observed that bit errors caused by faults in the memories are predominant [3,5,10,12,13(p.287)]. For example, in [5], the Mean Time Between Failure (MTBF) of a memory configuration composed of 64K by 1 RAM chips is reported to be between 49,000 hours and 390 hours for a 32K byte and 4M byte memory configuration, respectively. Per chip, the soft failure rate causing single-cell errors, is reported to be $1. * 10^{-6}h^{-1}$ and the hard failure rate to be $0.27 * 10^{-6}h^{-1}$ [5]. Hard failures cause single cell errors (50 percent), row errors (15.6 percent), column errors (28.1 percent), and row-column errors (6.3 percent) [5]. On the average, a hard failure causes about 72 cells of a memory chip to be in error.

Hence, in large memories, bit errors occur too frequently to repair at any occurence. Therefore, most memory arrays are protected by single-error-correcting, double-error-detecting Hamming codes [13]. In a TMR processor-memory configuration (Figure 2), all single bit errors are outvoted as long as the rest of the system is functioning correctly. However, one kth of all double bit errors leads to system failure in a k-bit configuration. Moreover after a module failure, the permanent bit errors, caused by memory faults, which were outvoted before, become disastrous for the system. Combining the figures given above with the fact that the failure rate of a processor board is in the order of $10^{-4} - 10^{-5}h^{-1}$, we see that upon a real module failure on the average several tens to several thousands of permanent memory cell faults are present, supposing the system was completely fault-free in the beginning. Hence, during repair time (assumed to be done under manual control in several minutes up to several days), the probability of reading a word in memory containing a bit error is very close to one. Such a bit error will bring the orthodox TMR configuration

down, so the coverage factor will not be quite impressive. Hence it is necessary to take additional measures against bit errors.

The problem can be solved by implementing an error-detecting/correcting code in each of the three memory arrays (Figure 3).

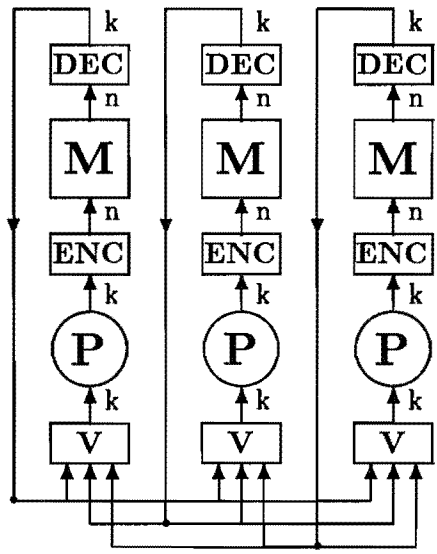


Figure 3: *An orthodox TMR configuration combined with a random bit error-protecting code (of length n and dimension k) to tolerate memory failures.*

However this causes the total memory size to grow by a factor more than three compared to that of a nonredundant module. Exact figures will be given in Section II. A much more elegant and efficient solution to this problem is to combine measures against symbol errors, caused by an arbitrary fault in one module, and bit errors, mainly caused by memory faults. This was shown by Krol, who in [7] introduced the '(N, K) concept' fault-tolerant computer. In [7] this concept was only developed for the case $N = 4$, $K = 2$ and 4-bit symbols. The '(4,2) concept' fault-tolerant computer described in [7] is based on an error-correcting code over the alphabet of 4-bit symbols and is able to correct one of the following error events: single symbol errors, double bit errors, and the combination of a symbol erasure (i.e., a known module failure)

and a single bit error. In the terminology of Krol [7], the TMR configuration is classified as a '(3,1) concept'. In this paper, we describe the so-called generalized TMR configuration, also a '(3,1) concept', in which the combined symbol and bit error-correction idea is used to its full extent.

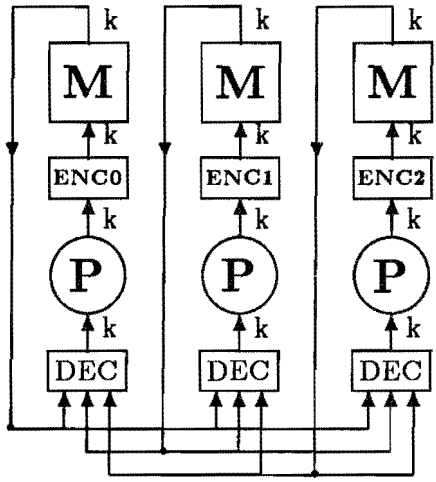


Figure 4: *The generalized TMR configuration; ENC_i = encoder i, DEC = decoder.*

In the generalized TMR configuration either all multiple bit errors up to a certain number or any single symbol error can be tolerated. Furthermore, after a module has failed and is discarded by the other two, bit errors up to a certain number in the remaining modules can still be tolerated. These extra features do not cost extra redundancy with respect to the orthodox TMR technique. The generalized TMR technique is based upon rate one-third error-detecting/correcting codes, specially constructed for this purpose. The general form of the generalized TMR technique is given in Figure 4. The k-bit output of a processor is encoded into a k-bit input for the memory in three different ways. The concatenation of three encoded symbols forms a codeword of the code used. In a READ operation the three decoders each receive the concatenation of three symbols, one from each memory. This possibly corrupted codeword is decoded in the decoders, i.e., symbol and bit errors are detected and corrected, respectively.

The encoder and decoder can easily be integrated into one chip for $k = 4$ and 8. So, this gives only a minor extension of the hardware in a module. The implementation of decoder chips in a (4,2) concept fault-tolerant computer have shown the feasibility of these approaches.

The contents of this paper are as follows. In Section II, we shall indicate how the orthodox TMR technique can be generalized. We shall mention the fault-tolerant properties of our generalized TMR configurations processing 4-, 8-, and 16-bit symbols, respectively. We also compare the amount of memory redundancy needed in the generalized TMR configuration with respect to the classical TMR/memory code configuration. In Section III, we give encoder/decoder pairs that satisfy the results of Section II and we describe a realization of the decoder for the TMR configuration on 8-bit symbols. Section IV describes a possible mode register updating strategy. Many of the ideas in the decoder construction are borrowed from the decoder construction for the (4,2) concept fault-tolerant computer [7,8]. Finally in Section V we construct the codes using Galois field theory and prove the error-correcting properties of the codes.

It should be remarked that some earlier work on high rate codes for combined bit error-correction and symbol error-detection was done for use in memory arrays composed of byte-organized chips [1,2,6]. This paper focuses on new well-structured rate of one-third combined bit and symbol error-correcting codes used within a complete computer configuration.

II. How to generalize TMR

Seen from the point of view of coding theory the orthodox TMR technique is a realization of a $[3k, k]$ binary linear code of length $3k$, dimension k, and generator matrix

$$G = [\, I \; I \; I \,], \tag{1}$$

where I denotes the k-by-k identity matrix. A message $\underline{m} = (m_0, m_1, \ldots, m_{k-1})$ is encoded into $\underline{c} = (\underline{c_0}, \underline{c_1}, \underline{c_2}) := \underline{m}G =$

$(\underline{m}, \underline{m}, \underline{m})$, i.e., it is triplicated. The symbol $\underline{c_i}$ is written in the memory of module i, $i = 0, 1, 2$. Let

$$\underline{e} := (\underline{e_0}, \underline{e_1}, \underline{e_2}) = (e_{00}, e_{01}, \ldots, e_{0k-1}, e_{10}, e_{11}, \ldots, e_{1k-1}, e_{20}, e_{21}, \ldots, e_{2k-1})$$

be the error pattern, then we receive $\underline{r} = \underline{c} + \underline{e}$ and estimate m_i by

$$m_i = \text{majority vote}\{r_{0i}, r_{1i}, r_{2i}\}$$

$$= \text{majority vote}\{m_i + e_{0i}, m_i + e_{1i}, m_i + e_{2i}\}. \tag{2}$$

Hence single symbol errors, i.e., exactly one of the symbols $\underline{e_0}, \underline{e_1}$, or $\underline{e_2}$ is nonzero, can be corrected. We also see that bit errors not occurring on corresponding positions are outvoted.

Instead of the matrix G in (1) let us take the matrix

$$G^* := [\, M_0 \; M_1 \; M_2 \,], \tag{3}$$

where M_0, M_1, and M_2 are non-singular k-by-k matrices, as generator matrix of a $[3k, k]$ code and let us write $\underline{c_i} = \underline{m}M_i$ into the memory of module i, $i = 0, 1, 2$ (Figure 4). This code can also correct single symbol errors, since the matrices M_0, M_1, and M_2 are non-singular. The question posed by us was how to choose M_0, M_1, and M_2 such that the code can also correct multiple bit errors (including those occurring on corresponding positions in symbols) in different symbols and combinations of a symbol erasure and bit errors. Without loss of generality M_0 can be chosen to be equal to I.

By using Galois field theory, we were able to find matrices M_1 and M_2 that give good codes, i.e., a large correction/detection capability for bit errors. This was done by means of theoretical considerations in combination with computer searches. In Section V we shall give the constructions using the theory of Galois fields. Here we shall restrict ourselves to developing everything in terms of vectors, matrices, and matrix-vector multiplications.

The error-correcting/detecting properties of the resulting codes for the generalized TMR configuration are given below. We also compare the generalized TMR configuration with a combination

of the orthodox TMR technique for correcting symbol errors and a binary code for correction/detection of random bit errors in the memories (see Figure 3). If a binary linear code of length n and dimension k is used in this latter TMR/memory code configuration the amount of memory hardware would be $3n/k$ times the amount of memory hardware in the simplex case (Figure 1). In the generalized TMR configuration, symbol and bit error correction are interwoven in the rate one-third code and hence memory hardware is only triplicated with respect to the simplex configuration and is of the same size as in the orthodox TMR configuration (Figure 2).

$k = 4$: Our generalized TMR configuration (Figure 4) processing 4-bit symbols is able to continue functioning correctly when one of the following failures occurs:

- the failure of a single module,
- the failure of up to two arbitrary memory bit-slices in arbitrary modules,
- the erasure of a module (i.e. we know explicitly which module is malfunctioning) and the simultaneous failure of a single memory bit-slice in another module.

To achieve at least the same performance (i.e., being able to continue functioning correctly whenever one of the above failures occurs), the TMR/memory code configuration of Figure 3 would at least require a binary code (of dimension 4) with minimum Hamming distance 2. Then the minimum value for the length of the code equals 5. Using an even-weight [5,4,2] code the decoders in Figure 3 can detect single bit errors and transmit an erasure symbol to the voters such that these will ignore the corresponding vote in the voting process. Hence, the amount of memory hardware is at least $3 \star 5/4 = 3.75$ times the amount of memory hardware of the simplex configuration.

$k = 8$: Our generalized TMR configuration (Figure 4) processing 8-bit symbols is able to continue functioning correctly when one of the following failures occurs:

- the failure of a single module,

- the failure of up to three arbitrary memory bit-slices in arbitrary modules,
- the erasure of a module and the simultaneous failure of up to two arbitrary memory bit-slices in arbitrary modules.

It is also able to detect the failure of four arbitrary memory bit-slices in arbitrary modules.

To have at least the same performance, the TMR/memory code configuration of Figure 3 would at least require a binary code (of dimension 8) with minimum distance 3. Then the minimum value of the length of the code equals 12 (cf. [4]). Hence the amount of memory hardware is at least $3 \star 12/8 = 4.5$ times the amount of memory hardware of the simplex configuration.

$k = 16$: Our generalized TMR configuration (Figure 4) processing 16-bit symbols is able to continue functioning correctly when one of the following failures occurs:

- the failure of a single module,
- the failure of up to five arbitrary memory bit-slices in arbitrary modules,
- the erasure of a module and the simultaneous failure of up to three arbitrary memory bit-slices in arbitrary modules.

It is also able to detect the failure of six arbitrary memory bit-slices in arbitrary modules.

To have at least the same performance the TMR/memory code configuration of Figure 3 at least requires a binary code (of dimension 16) with minimum distance 4. Then the minimum value of the length of the code equals 22 (cf. [4] and [9]). Hence the amount of memory hardware is at least 4.125 times the amount of memory hardware of the simplex configuration.

It should be remarked that, for example, for a generalized TMR configuration containing 16-bit processors, two [3,1] codes on 8-bit symbols can also be used in parallel, one for the lower byte and one for the upper byte.

III. Construction of encoder/decoder pairs

In our generalized TMR configuration a k-bit symbol $\underline{m} \in \mathbf{F} := \{0,1\}^k$ is encoded into a codeword $\underline{c} = (\underline{c_0}, \underline{c_1}, \underline{c_2}) := \underline{m}G$, where

$$G = [\, I \; M \; M^2 \,] \tag{4}$$

and M is a non-singular k-by-k matrix of order 3, i.e., $M \neq I, M^2 \neq I, M^3 = I$. The code C is defined to be the set $\{\underline{c} = \underline{m}G = (\underline{m}, \underline{m}M, \underline{m}M^2) | \underline{m} \in \mathbf{F}\}$ of codewords. The encoder ENCi, $i = 0, 1, 2$ writes $\underline{c_i}$ into the memory of module i. The three identical decoders receive possibly erroneous versions of $(\underline{c_0}, \underline{c_1}, \underline{c_2})$. The matrix H defined by

$$H := \begin{bmatrix} O & M^T & I \\ I & O & M^T \\ M^T & I & O \end{bmatrix} \tag{5}$$

is a 'redundant' (since $(M^T)^2(O\ M^T\ I)+M^T(I\ O\ M^T) = (M^T\ I\ O)$) parity-check matrix of the code C, i.e., for all $\underline{c} \in \mathbf{F}^3$ we have

$$\underline{c} \in C \text{ if and only if } \underline{c}H^T = \underline{0}. \tag{6}$$

We define the syndrome $\underline{s} = (\underline{s_0}, \underline{s_1}, \underline{s_2})$ of a vector $\underline{r} = (\underline{r_0}, \underline{r_1}, \underline{r_2}) \in \mathbf{F}^3$ to be

$$\underline{s} = (\underline{s_0}, \underline{s_1}, \underline{s_2}) := \underline{r}H^T = (\underline{r_0}, \underline{r_1}, \underline{r_2}) \begin{bmatrix} O & I & M \\ M & O & I \\ I & M & O \end{bmatrix}$$

$$= (\underline{r_1}M + \underline{r_2}, \underline{r_0} + \underline{r_2}M, \underline{r_0}M + \underline{r_1}). \tag{7}$$

If for any pair $\underline{e^{(1)}}, \underline{e^{(2)}}$ of different error patterns in a fixed set $\mathcal{U}$ of error patterns it holds that their sum $\underline{e^{(1)}} + \underline{e^{(2)}}$ is not in the code C, or in other words

$$\underline{s^{(1)}} := \underline{e^{(1)}}H^T \neq \underline{e^{(2)}}H^T =: \underline{s^{(2)}}, \tag{8}$$

then the error patterns in $\mathcal{U}$ build up a set of correctable error patterns.

Furthermore if a module is known to be faulty, i.e., from the point of view of coding theory an erasure occurs, then the decoders are set into an erasure mode. Without loss of generality let us suppose the first symbol to be an erasure. Hence the decoder is in erasure mode 0 and it receives

$$(\text{“erasure”}, \underline{c}_1 + \underline{e}_1, \underline{c}_2 + \underline{e}_2). \tag{9}$$

The two symbols $(\underline{c}_1, \underline{c}_2)$ have to satisfy the equation

$$(\underline{c}_1, \underline{c}_2) \begin{bmatrix} M \\ I \end{bmatrix} = \underline{0}. \tag{10}$$

Hence $(\underline{c}_1, \underline{c}_2)$ is a codeword of a $[2k, k]$ binary code with parity-check matrix $[M^T\ I]$. If this code has minimum (Hamming) distance $2t + 1$ then it can correct t bit errors. Hence if we observe an erasure we can correct t bit errors in addition.

For a binary vector $\underline{x}$ we define $wt(\underline{x})$ to be the (Hamming) weight of $\underline{x}$, i.e. the number of ones in the vector $\underline{x}$. Define (*00) to be the set

$$\{(\underline{x}, \underline{0}, \underline{0}) | \underline{x} \in \mathbf{F}, \underline{x} \neq \underline{0}\}. \tag{11}$$

In an analogous way (0*0) and (00*) are defined. Define (ijk) to be the set

$$\{(\underline{x}, \underline{y}, \underline{z}) | \underline{x}, \underline{y}, \underline{z} \in \mathbf{F}, wt(\underline{x}) = i, wt(\underline{y}) = j, wt(\underline{z}) = k\}. \tag{12}$$

The sets denoted by these three-tuples are called *error classes.* Taking an arbitrary error class, then all syndromes of the elements in that error class have some properties in common. For example, all error patterns $\underline{e}$ in (*00) have in common that their syndromes $\underline{s} = (\underline{s}_0, \underline{s}_1, \underline{s}_2) := \underline{e}H^T$ satisfy $\underline{s}_0 = \underline{0}, \underline{s}_1 \neq \underline{0}, \underline{s}_2 \neq \underline{0}$; All error patterns $\underline{e}$ in $(ij0)$ have in common that their syndromes $\underline{s} = \underline{e}H^T$ satisfy $wt(\underline{s}_1) = i$ and $wt(\underline{s}_0 M^2) = j$. We shall use such properties shared by error patterns in an error class in the decoding process.

Let us define the following sets ("+" denotes "union of").

$\mathcal{NE} :=$ (000), no error.

$\mathcal{SS} :=$ ($*00$) + ($0*0$) + ($00*$), the set of all single symbol errors.

$\mathcal{DB}$:= (110) + (101) + (011), the set of all double bit errors that are not single symbol errors.

$\mathcal{TB}$:= (111) + (210) + (120) + (201) + (102) + (021) + (012), the set of all triple bit errors that are not single symbol errors.

$\mathcal{QB}$:= the set of all quadruple bit errors that are not single symbol errors.

$\mathcal{FB}$:= the set of all quintuple bit errors that are not single symbol errors.

$\mathcal{SB}$:= the set of all sextuple bit errors that are not single symbol errors.

We shall now give the matrices M that have the desired error correction capabilities. The construction of these matrices will be explained in Section V.

$k = 4$: If we take for the matrix M in formula (4) the matrix

$$M = \begin{bmatrix} 0110 \\ 0011 \\ 1101 \\ 1010 \end{bmatrix}, \tag{13}$$

then the set $\mathcal{U} = \mathcal{NE} + \mathcal{SS} + \mathcal{DB}$ is a set of correctable error patterns. In erasure mode we use the [8,4] code with generator matrix $[I\ M]$ having a minimum distance of 3.

$k = 8$: If we take for the matrix M in formula (4) the matrix

$$M = \begin{bmatrix} 11001111 \\ 10100111 \\ 11110011 \\ 11101001 \\ 11111100 \\ 01111010 \\ 00111111 \\ 10011110 \end{bmatrix}, \tag{14}$$

then the set $\mathcal{U} = \mathcal{NE} + \mathcal{SS} + \mathcal{DB} + \mathcal{TB}$ is a set of correctable error patterns. Furthermore the error patterns in $\mathcal{QB}$ are detectable. In

erasure mode we are left with a [16,8] code with generator matrix $[I\ M]$ having a minimum distance of 5.

$k = 16$: If we take for the matrix M in formula (4) the matrix

$$M = \begin{bmatrix} 0011011011101000 \\ 0101101101110100 \\ 0000110110111010 \\ 0001011011011101 \\ 1000001101101110 \\ 0100010110110111 \\ 1010000011011011 \\ 1101000101101101 \\ 1110100000110110 \\ 0111010001011011 \\ 1011101000001101 \\ 1101110100010110 \\ 0110111010000011 \\ 1011011101000101 \\ 1101101110100000 \\ 0110110111010001 \end{bmatrix}, \tag{15}$$

then the set $\mathcal{U} = \mathcal{NE} + \mathcal{SS} + \mathcal{DB} + \mathcal{TB} + \mathcal{QB} + \mathcal{FB}$ is a set of correctable error patterns. Furthermore the error patterns in $\mathcal{SB}$ are detectable. In erasure mode we are left with a [32,16] code with generator matrix $[I\ M]$ having a minimum distance of 7.

We shall now describe the main ideas that are used in designing decoders for the generalized TMR configurations by describing the decoder for the case $k = 8$. The decoders for the cases $k = 4$ and $k = 16$, respectively, are constructed according to the same ideas and are respectively less and more complex than the one for the case $k = 8$. Many of the ideas in the decoders of generalized TMR are borrowed from the decoder design for the (4,2) concept fault-tolerant computer [7,8]. Realization on chip of the decoder for the (4,2) concept shows the feasibility of such decoders.

The decoder can operate in 7 different modes, which are:

RM: random mode, i.e., the decoder uses all three outputs of the modules,

EM_i: erasure mode $i, i = 0, 1, 2$, i.e., the decoder considers module i to be malfunctioning and uses the outputs of the other two modules for decoding (duplicated configuration),

SM_i: single mode $i, i = 0, 1, 2$, i.e., the decoder only considers the output of module i (simplex configuration).

Random mode

The elements of a fixed error class have some properties in common. Having a certain property can be translated by satisfying a Boolean expression. The clue is to construct a set of Boolean expressions, one for each error class contained in the set of correctable error patterns $\mathcal{U}$, such that these Boolean expressions are mutually exclusive; i.e., all error patterns in $\mathcal{U}$ satisfy exactly one of these Boolean expressions and all other error patterns outside $\mathcal{U}$ satisfy at most one of these Boolean expressions.

We define the following Boolean variables. For $i = 0, 1, 2$, $j = 0, 1, 2, 3$,

$$\begin{aligned} p_{ij} = 1 &\quad \text{iff} \quad wt(\underline{s_i}) = j, \\ q_{ij} = 1 &\quad \text{iff} \quad wt(\underline{s_i}M^2) = j, \\ RM = 1 &\quad \text{iff} \quad \text{the decoder is running in random mode,} \\ EM_i = 1 &\quad \text{iff} \quad \text{the decoder is running in erasure mode } i, \\ SM_i = 1 &\quad \text{iff} \quad \text{the decoder is running in single mode } i. \end{aligned} \tag{16}$$

Furthermore, we define the following estimates for the message $\underline{m}$:

$$\underline{\hat{m}_0} := \underline{r_0}, \underline{\hat{m}_1} := \underline{r_1}M^2, \underline{\hat{m}_2} := \underline{r_2}M, \underline{\hat{n}_{ij}} := (\underline{r_i} + \underline{u_j})M^{3-i} \tag{17}$$

for $i = 0, 1, 2$ and $j = 0, 1, \ldots, 7$, where $\underline{u_j}$ is the unity vector having a 1 in the j-th position and zeros elsewhere.

	error class	condition	message estimate
$\mathcal{NE}$	(000)	$p_{00}p_{10}$	$\underline{\hat{m}_0}$
$\mathcal{SS}$	(*00)	$p_{00}\overline{p_{10}}$	$\underline{\hat{m}_1}$
	(0*0)	$p_{10}\overline{p_{20}}$	$\underline{\hat{m}_2}$
	(00*)	$p_{20}\overline{p_{00}}$	$\underline{\hat{m}_0}$
$\mathcal{DB}$	(110)	$p_{11}q_{01}$	$\underline{\hat{m}_2}$
	(101)	$p_{01}q_{21}$	$\underline{\hat{m}_1}$
	(011)	$p_{21}q_{11}$	$\underline{\hat{m}_0}$
$\mathcal{TB}$	(111)	$R\ par$	$\underline{\hat{n}_{1j}}$
	(210)	$p_{12}q_{01}$	$\underline{\hat{m}_2}$
	(120)	$p_{11}q_{02}$	$\underline{\hat{m}_2}$
	(201)	$p_{01}q_{22}$	$\underline{\hat{m}_1}$
	(102)	$p_{02}q_{21}$	$\underline{\hat{m}_1}$
	(021)	$p_{22}q_{11}$	$\underline{\hat{m}_0}$
	(012)	$p_{21}q_{12}$	$\underline{\hat{m}_0}$
$\mathcal{QB}$	(310)	$R\ \overline{par}$	error detection flag
	(130)		
	(301)		
	(103)		
	(031)		
	(013)		
	(220)		
	(202)		
	(022)		
	(211)		
	(121)		
	(112)		

Table I: *The correctable/detectable error patterns in random mode for the case $k = 8$ together with the Boolean expression they satisfy and the corresponding estimates for the message.*

In random mode ($RM = 1$) the set $\mathcal{U} = \mathcal{NE} + \mathcal{SS} + \mathcal{DB} + \mathcal{TB}$ is the set of correctable error patterns. Table I gives the boolean expressions which are satisfied by error patterns in the correctable error classes, and the resulting estimates for the message. The first column gives the correctable error classes. The second column

provides the corresponding Boolean expressions satisfied by error patterns in these classes. These Boolean expressions are mutually exclusive. A correctable error pattern in $\mathcal{U}$ satisfies exactly one of these Boolean expressions. The Boolean variables R and *par* are defined as follows.

$$\overline{R} := (p_{00}p_{10} + p_{00}\overline{p_{10}} + p_{10}\overline{p_{20}} + p_{20}\overline{p_{00}} + p_{11}q_{01} + p_{01}q_{21}$$

$$+ p_{21}q_{11} + p_{12}q_{01} + p_{11}q_{02} + p_{01}q_{22} + p_{02}q_{21} + p_{22}q_{11} + + p_{21}q_{12}), \quad (18)$$

$$par := \sum_{i=0}^{2}\sum_{j=0}^{7} r_{ij}. \quad (19)$$

Note the relation of R to the other Boolean expressions in Table I. $R = 1$ if none of the other Boolean expressions equals one. The Boolean variable *par* denotes the parity of the received word. Note that all codewords have even parity, because $I + M + M^2 = O$. The third column of Table I gives the estimates of the message corresponding to the error classes in the first column. They form the output of the decoder. For all correctable error classes, except the class (111), it is easy to find an estimate for the message, because at least one symbol of the received word is correct. For an error in the error class (111) we have to determine a bit error in one of the received symbols. The solution of this problem will be treated later on.

So we use the Boolean expressions in the second column of Table I in our decoder and estimate the message by the estimate given in the third column if the corresponding Boolean expression is satisfied. This means that all error patterns in $\mathcal{U}$ are corrected.

For an error pattern in $\mathcal{U}$ we have $R\,\overline{par} = 0$, for an error pattern $\underline{e}$ in $\mathcal{QB}$ we have $R\,\overline{par} = 1$. So we can detect all error patterns in $\mathcal{QB}$ very easily by evaluating $R\,\overline{par}$.

error class	condition	message estimate
$(E00)$	p_{00}	$\underline{\hat{m}_1}$
$(E10)$	q_{01}	$\underline{\hat{m}_2}$
$(E01)$	p_{01}	$\underline{\hat{m}_1}$
$(E11)$	T_0	$\underline{\hat{n}_{2j}}$
$(E20)$	q_{02}	$\underline{\hat{m}_2}$
$(E02)$	p_{02}	$\underline{\hat{m}_1}$
$(0E0)$	p_{10}	$\underline{\hat{m}_2}$
$(1E0)$	p_{11}	$\underline{\hat{m}_2}$
$(0E1)$	q_{11}	$\underline{\hat{m}_0}$
$(1E1)$	T_1	$\underline{\hat{n}_{0j}}$
$(2E0)$	p_{12}	$\underline{\hat{m}_2}$
$(0E2)$	q_{12}	$\underline{\hat{m}_0}$
$(00E)$	p_{20}	$\underline{\hat{m}_0}$
$(10E)$	q_{21}	$\underline{\hat{m}_1}$
$(01E)$	p_{21}	$\underline{\hat{m}_0}$
$(11E)$	T_2	$\underline{\hat{n}_{1j}}$
$(20E)$	q_{22}	$\underline{\hat{m}_1}$
$(02E)$	p_{22}	$\underline{\hat{m}_0}$

Table II: *The correctable error patterns in the erasure modes for the case $k = 8$, together with the Boolean expressions they satisfy and the corresponding estimates for the message*

Erasure mode

For $k = 8$, in erasure mode, two bit errors can be corrected. An erasure will be denoted by 'E'. For example, (Eij) denotes the set of error patterns with the first symbol being an erasure, the second symbol having weight i, and the third symbol having weight j. Define the Boolean expressions T_0, T_1, and T_2 as

$$\begin{aligned} \overline{T}_0 &:= (p_{00} + q_{01} + p_{01} + q_{02} + p_{02}), \\ \overline{T}_1 &:= (p_{10} + p_{11} + q_{11} + p_{12} + q_{12}), \\ \overline{T}_2 &:= (p_{20} + q_{21} + p_{21} + q_{22} + p_{22}). \end{aligned} \tag{20}$$

Table II is the analogue of Table I for the three erasure modes.

The Boolean expressions in this table can again be used for error correction. The third column gives the estimates for the message. In the cases of the error classes $(E11), (1E1)$, and $(11E)$ we have to determine one bit error in one of the received symbols. For the other error classes the estimate for the message is straightforwardly retrievable from the received word.

error class	condition	two-symbol codeword
(111)	$R\,par = 1$	$(\underline{c_0}, \underline{c_1})$
$(E11)$	$T_0 = 1$	$(\underline{c_1}, \underline{c_2})$
$(1E1)$	$T_1 = 1$	$(\underline{c_2}, \underline{c_0})$
$(11E)$	$T_2 = 1$	$(\underline{c_0}, \underline{c_1})$

Table III: *Special error classes together with corresponding Boolean expression and corresponding two-symbol code word*

As we have seen in the case of an error class of the type $(111), (E11)$, $(1E1)$, or $(11E)$ we can make no straightforward estimate for the message $\underline{m}$. We have to find a bit error in one of the received symbols. In these cases we only consider two symbols of the codeword sent, which we order such that they form a codeword of the code with parity-check matrix $[M^T\ I]$. The four error classes, corresponding conditions, and related two-symbol codewords are given in Table III.

So the problem is to decode an error pattern $(\underline{e_j}, \underline{e_i}) = (\underline{r_j}, \underline{r_i}) - (\underline{c_j}, \underline{c_i})$ with $wt(\underline{e_j}) = wt(\underline{e_i}) = 1$ with respect to the code with parity-check matrix $[M^T\ I]$. The syndrome of $(\underline{e_j}, \underline{e_i})$ is $\underline{s'} := \underline{e_j}M + \underline{e_i}$. We try to find $\underline{e_i}$ by substituting all 8-bit vectors of weight one for $\underline{e_i}$ and checking whether $\underline{e_j}$ has weight one. This gives the following decoding rule, which sets a boolean variable $t_{ip} (i = 0, 1, 2, p = 0, 1, \ldots, 7)$ equal to one if a certain error pattern occurred.

- compute $\underline{s'} = \underline{e_j}M + \underline{e_i}$,
- find an 8-bit vector $\underline{u}$ of weight one such that $wt((\underline{s'} + \underline{u})M^2) = 1$,
- if $\underline{u} = \underline{u_p}$ is the vector found in the previous step then set $t_{ip} = 1$.

decoder	syndrome component used	corresponding two-symbol codeword	code symbol estimated
dec_2	$\underline{s_0}$	$(\underline{c_1}, \underline{c_2})$	$\underline{c_2}$
dec_0	$\underline{s_1}$	$(\underline{c_2}, \underline{c_0})$	$\underline{c_0}$
dec_1	$\underline{s_2}$	$(\underline{c_0}, \underline{c_1})$	$\underline{c_1}$

Table IV: *Three decoders with corresponding syndrome component, two-symbol code word, and code symbol estimated*

We use three decoders for performing the above decoding rule for $i = 0, 1, 2$ respectively. Table IV gives the three decoders together with the syndrome component $\underline{s_i}$ used, the corresponding two-symbol codeword, and the code symbol that is estimated. If $t_{ip} = 1$ then the estimate for the message $\underline{m}$ is $\hat{n}_{ip} = (\underline{r_i} + \underline{u_p})M^{3-i}$. Summarizing, the message $\underline{m}$ is estimated as follows:

$$\begin{array}{ll} \text{if } EM_0T_0t_{2j} = 1 & \text{then } \underline{\hat{m}} = \underline{\hat{n}_{2j}}, \\ \text{if } EM_1T_1t_{0j} = 1 & \text{then } \underline{\hat{m}} = \underline{\hat{n}_{0j}}, \\ \text{if } (RM\ R\ par + EM_2T_2)t_{1j} = 1 & \text{then } \underline{\hat{m}} = \underline{\hat{n}_{1j}}. \end{array} \qquad (21)$$

The above remarks on the decoding strategy lead to the decoder implementation of Figure 5. In this figure the boxes WT compute the weight of the input vector. The box * contains the implementation of the following Boolean expressions:

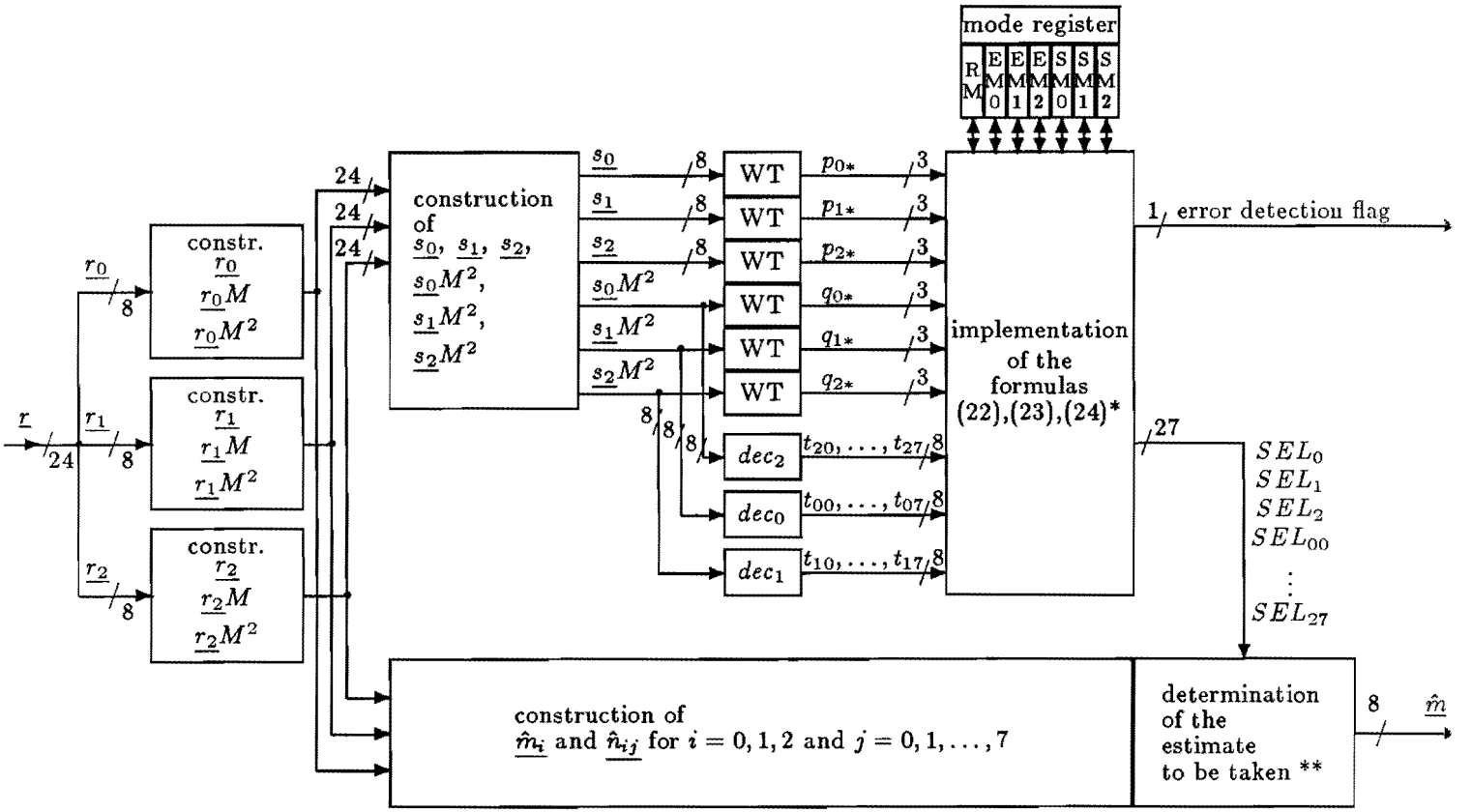


Figure 5: The decoder design for the generalized TMR configuration processing 8-bit words.

$$
\begin{aligned}
SEL_0 &:= RM(p_{00}p_{10} + p_{20}\overline{p_{00}} + p_{21}q_{11} + p_{22}q_{11} + p_{21}q_{12}) \\
&\quad + EM_1(q_{11} + q_{12}) + EM_2(p_{20} + p_{21} + p_{22}) + SM_0, \\
SEL_1 &:= RM(p_{00}\overline{p_{10}} + p_{01}q_{21} + p_{01}q_{22} + p_{02}q_{21}) \\
&\quad + EM_0(p_{00} + p_{01} + p_{02}) + EM_2(q_{21} + q_{22}) + SM_1, \\
SEL_2 &:= RM(p_{10}\overline{p_{20}} + p_{11}q_{01} + p_{12}q_{01} + p_{11}q_{02}) \\
&\quad + EM_0(q_{01} + q_{02}) + EM_2(p_{10} + p_{11} + p_{12}) + SM_2.
\end{aligned} \tag{22}
$$

$$
\begin{aligned}
SEL_{0j} &:= EM_1T_1t_{0j}, \\
SEL_{1j} &:= (RM\ R\ par + EM_2T_2)t_{1j}, \\
SEL_{2j} &:= EM_0T_0t_{2j}, \quad \text{for } j = 0, 1, \ldots, 7, \\
detection flag &:= (\textstyle\prod_{i=0}^{2} \overline{SEL_i})(\prod_{i=0}^{2}\prod_{j=0}^{7} \overline{SEL_{ij}}).
\end{aligned} \tag{23}
$$

These 28 Boolean expressions are mutually exclusive. In box ** it is decided what estimate for the message should be taken.

If $SEL_i = 1$ then $\underline{\hat{m}} = \underline{\hat{m}_i}$,
If $SEL_{ij} = 1$ then $\underline{\hat{m}} = \underline{\hat{n}_{ij}}$,
If detection flag $= 1$ then detection of an uncorrectable error.

Furthermore, box * contains the implementation of the Boolean expressions that are used for the mode register updating. They will be described in the next section.

IV. Mode register updating

The decoder contains a mode register whose contents are changed whenever certain errors are found in the received codeword. For changing the mode register many strategies are possible, depending on the failure characteristics of a module.

A possible strategy could be to switch from random mode to erasure mode when the maximum error correction capacity is reached, i.e., a symbol error not being a single bit error. It seems also wise to switch from erasure mode to single mode if two bit errors in the same symbol occur. Hence, we get the following

formulas for the register updating procedure.

$$\begin{aligned}
EM_0^{out} &:= RM^{in} p_{00}\,\overline{p_{10}}\,\overline{p_{11}} + EM_0^{in}\overline{p_{02}}\,\overline{q_{02}},\\
EM_1^{out} &:= RM^{in} p_{10}\,\overline{p_{20}}\,\overline{p_{21}} + EM_1^{in}\overline{p_{12}}\,\overline{q_{12}},\\
EM_2^{out} &:= RM^{in} p_{20}\,\overline{p_{00}}\,\overline{p_{01}} + EM_2^{in}\overline{p_{22}}\,\overline{q_{22}},\\
RM^{out} &:= RM^{in}(\overline{p_{00}} + p_{10} + p_{11})\\
&\quad (\overline{p_{10}} + p_{20} + p_{21})(\overline{p_{20}} + p_{00} + p_{01}),\\
SM_0^{out} &:= SM_0^{in} + EM_1^{in} q_{12} + EM_2^{in} p_{22},\\
SM_1^{out} &:= SM_1^{in} + EM_0^{in} p_{02} + EM_2^{in} q_{22},\\
SM_2^{out} &:= SM_2^{in} + EM_0^{in} q_{02} + EM_1^{in} p_{12}.
\end{aligned} \tag{24}$$

Switching from single to erasure mode and from erasure to random mode should be done under software control after a faulty module has been replaced by a good one.

V. Construction and properties of the codes

In this section we provide the construction of the codes for generalized TMR. For this construction we used the theory of Galois fields. For an extensive treatment of Galois fields we refer the reader to [9, Ch. 4].

Let $p(x)$ be a primitive polynomial of degree k and let α be a zero of $p(x)$. Then α is a primitive element of the Galois field $GF(2^k) := \{0, 1, \alpha, \alpha^2, \ldots, \alpha^{2^k-2}\}$. The Galois field $GF(2^k)$ is a vector space of dimension k over $GF(2)$, the Galois field containing two elements. The k elements $\alpha^{b(0)}, \alpha^{b(1)}, \ldots, \alpha^{b(k-1)}$ are called a basis of $GF(2^k)$ over $GF(2)$ if no nontrivial solution of the equation

$$\sum_{i=0}^{k-1} a_i \alpha^{b(i)} = 0, \quad a_0, a_1, \ldots, a_{k-1} \in GF(2). \tag{25}$$

exists. In other words the only solution is $a_0 = a_1 = \ldots = a_{k-1} = 0$. When such is the case, any element γ in $GF(2^k)$ can be repre-

sented by a binary k-tuple $(a_0, a_1, \ldots, a_{k-1})$ such that

$$\gamma = \sum_{i=0}^{k-1} a_i \alpha^{b(i)}. \tag{26}$$

The number of ones in $(a_0, a_1, \ldots, a_{k-1})$ is called the weight of γ with respect to the basis $< \alpha^{b(0)}, \alpha^{b(1)}, \ldots, \alpha^{b(k-1)} >$. A Galois field can have many different bases. Some of them have a special form and hence, a name is given to them. The basis $< \alpha^i,\ i = 0, 1, \ldots, k-1 >$ is called the *polynomial basis* corresponding to the polynomial $p(x)$. If, for some j

$$< \alpha^{j2^i},\ i = 0, 1, \ldots, k-1 > \tag{27}$$

forms a basis of $GF(2^k)$ over $GF(2)$ then this basis is called a *normal basis.*

Now define $\beta := \alpha^{(2^k-1)/3}$ (for even k, $2^k - 1$ is always divisible by 3). Then β is a primitive element of $GF(2^2) := \{0, 1, \beta, \beta^2\}$. Consider the [3,1] code of length 3 and dimension 1 over $GF(2^k)$ with generator matrix $[1\ \beta\ \beta^2]$. We map this [3,1] code over $GF(2^k)$ onto a binary $[3k, k]$ code of length $3k$ and dimension k by using the vector representation of $GF(2^k)$ with respect to a given basis $< \alpha^{b(0)}, \alpha^{b(1)}, \ldots, \alpha^{b(k-1)} >$. The generator matrix of the binary $[3k, k]$ code is

$$G^* = [\ I\ M\ M^2\ ],$$

where M is the k by k matrix whose ith row $(i = 0, 1, \ldots, k-1)$ is the binary representation of $\beta\alpha^{b(i)}$ with respect to the basis $< \alpha^{b(0)}, \alpha^{b(1)}, \ldots, \alpha^{b(k-1)} >$.

The minimum distance profile [11] $d(C) = (d(C|0), d(C|1), d(C|2))$ of a [3,1] code C is defined by

$$\begin{aligned} d(C|0) &:= \min\{wt(c_0, c_1, c_2) : (c_1, c_2, c_3) \in C \backslash \{0\}\}, \\ d(C|1) &:= \min\{wt(c_0, c_1), wt(c_0, c_2), wt(c_1, c_2) : (c_0, c_1, c_2) \in C \backslash \{0\}\}, \\ d(C|2) &:= \min\{wt(c_0), wt(c_1), wt(c_2) : (c_0, c_1, c_2) \in C \backslash \{0\}\}. \end{aligned} \tag{28}$$

So for every nonzero codeword $\underline{c} = (c_0, c_1, c_2)$ in C the following

seven relations hold:

$$\begin{aligned} wt(c_0) + wt(c_1) + wt(c_2) &\geq d(C|0); \\ wt(c_i) + wt(c_j) &\geq d(C|1) \text{ for } i,j \in \{0,1,2\}, i \neq j; \\ wt(c_i) &\geq d(C|2) \text{ for } i = 0,1,2. \end{aligned} \tag{29}$$

These relations imply the following property concerning correctable error patterns.

Property I: A set $\mathcal{U}$ of error patterns is a correctable error set if for all pairs $\underline{e^{(1)}}$ and $\underline{e^{(2)}}$ of different elements of $\mathcal{U}$ at least one of the 7 relations of (29) does not hold for their sum $\underline{e^{(1)}} + \underline{e^{(2)}}$.

From the definition of minimum distance profile and relations (29) it is easy to see that the following property must hold.

Property II: $d(C|0) \geq 3 \star d(C|1)/2$.

We will now consider bases $< \alpha^{b(0)}, \alpha^{b(1)}, \ldots, \alpha^{b(k-1)} >$ such that the [3,1] codes over $GF(2^k)$ constructed above have large minimum distance profiles with respect to these bases. We do this for the cases $k = 4, 8$, and 16, respectively.

$k = 4$: The polynomial $p(x) := x^4 + x + 1$ is a primitive polynomial of $GF(2^4)$ over $GF(2)$. Let α be a zero of $p(x)$. The field $GF(2^4)$ has two normal bases, which are $< \alpha^3, \alpha^6, \alpha^{12}, \alpha^9 >$ and $< \alpha^7, \alpha^{14}, \alpha^{13}, \alpha^{11} >$. The element β is defined to be $\beta := \alpha^5$.

The [3,1] code C with generator matrix

$$G = [\ 1\ \beta\ \beta^2\ ] = [\ 1\ \alpha^5\ \alpha^{10}\ ]$$

has minimum distance profile (6,3,1) with respect to the polynomial basis $< \alpha^0, \alpha^1, \alpha^2, \alpha^3 >$ as well as with respect to both normal bases $< \alpha^3, \alpha^6, \alpha^{12}, \alpha^9 >$ and $< \alpha^7, \alpha^{14}, \alpha^{13}, \alpha^{11} >$. It is easily checked that Property I holds for the set $\mathcal{U} = \mathcal{NE} + \mathcal{SS} + \mathcal{DB}$, which therefore is a correctable error set.

Fix one of the three bases mentioned above and call the corresponding code C. If for some $i = 0,1,2$ the ith symbol in every codeword of C is erased, we obtain a [8,4] code $C^{(i)}$ having a minimum bit distance of 3. Hence in erasure mode we can correct single bit errors.

Note that the codes $C^{(0)}, C^{(1)}$, and $C^{(2)}$ are equivalent, because $[\ M\ M^2\ ] = M\ [\ I\ M\ ]$ and $[\ I\ M^2\ ] = M^2\ [\ M\ I\ ]$. By straightforward checking it is easy to see that the [8,4] code with generator matrix $[\ I\ M\ ]$ has a minimum bit distance of 3. Hence by Property II we have that $d(C|0) \geq 3 \star 3/2$, i.e., $d(C|0) \geq 5$, because $d(C|0)$ is an integer. The weight of every codeword in C is even, because $I + M + M^2 = O$. Hence we have that $d(C|0) \geq 6$. According to [4] a [12,4] binary linear code has minimum bit distance at most 6, so $d(C|0) = 6$. This shows that the minimum distance profile of the code C equals (6,3,1).

The generator matrix of formula (4) in which the matrix M of formula (13) is substituted is the binary image of the matrix $[\ 1\ \beta\ \beta^2\ ]$ with respect to the polynomial basis $< 1, \alpha, \alpha^2, \alpha^3 >$.

$k = 8$: The polynomial $p(x) = x^8+x^4+x^3+x^2+1$ is a primitive polynomial of $GF(2^8)$ over $GF(2)$. Let α be a zero of $p(x)$. $GF(2^8)$ has 16 normal bases, that are $N_i =< \alpha^{i2^j},\ j = 0,1,\ldots,7 >$ for $i = 5, 9, 11, 15, 21, 29, 39, 43, 47, 53, 55, 61, 63, 87, 91, 95$. The element β is defined to be $\beta := \alpha^{85}$.

The [3,1] code with generator matrix

$$G = [\ 1\ \beta\ \beta^2\ ] = [\ 1\ \alpha^{85}\ \alpha^{170}\ ]$$

has minimum distance profile (8,4,1) with respect to the normal bases N_i for i in $\mathcal{A} := \{9, 21, 39, 43, 55, 61, 91, 95\}$, and minimum distance profile (8,5,1) with respect to the normal bases N_i for i in $\mathcal{B} := \{5, 11, 15, 29, 47, 53, 63, 87\}$.

Consider the [24,8] binary code C_i obtained from the above [3,1] code with respect to the normal basis $N_i, i \in \mathcal{B}$. By straightforward checking it is easy to see that the [16,8] binary code obtained from it by deleting a symbol in every codeword has minimum bit distance 5. Hence by Property II we have that $d(C_i|0) \geq 3 \star 5/2$, i.e., $d(C_i|0) \geq 8$. The weight of every codeword in C_i is even, because $1 + \beta + \beta^2 = 0$. Furthermore, the minimum distance of a [24,8] binary linear code is not larger than 9 [4]. Hence $d(C_i|0) = 8$. This shows that the minimum distance profile of C_i equals (8,5,1). Therefore by Property I we see that the set $\mathcal{U} = \mathcal{NE} + \mathcal{SS} + \mathcal{DB} + \mathcal{TB}$ forms a correctable error set in random mode. In erasure mode we are left with a [16,8] binary

code with minimum distance 5, so up to two bit errors can be corrected.

The generator matrix of formula (4) with the matrix M of formula (14) substituted in it is the binary image of the matrix $[\ 1\ \beta\ \beta^2\ ]$ with respect to the basis N_5.

$k = 16$: The polynomial $p(x) = x^{16} + x^{12} + x^3 + x + 1$ is a primitive polynomial of $GF(2^{16})$ over $GF(2)$. Let α be a zero of $p(x)$. The Galois field $GF(2^{16})$ has a large number of normal bases. For example, $N_{15} :=< \alpha^{15*2^j}, j = 0,1,\ldots,15 >$ forms a normal basis of $GF(2^{16})$ over $GF(2)$. The element β is defined to be $\beta := \alpha^{21845}$.

The [3,1] code C over $GF(2^{16})$ with generator matrix

$$G = [\ 1\ \beta\ \beta^2\ ] = [\ 1\ \alpha^{21845}\ \alpha^{43690}\ ]$$

has minimum distance profile (12,7,1) with respect to the normal basis N_{15}. By straightforward checking it is easy to see that the [32,16] binary code obtained from it by deleting a symbol in every codeword has minimum distance 7. Hence by Property II we have that $d(C|0) \geq 11$. All weights of the codewords in C are even, so $d(C|0) \geq 12$. Actually $d(C|0) = 12$, because the code C contains a codeword of weight 12. This shows that the minimum distance profile of C equals (12,7,1). Therefore, by Property I we see that $\mathcal{U} = \mathcal{NE} + \mathcal{SS} + \mathcal{DB} + \mathcal{TB} + \mathcal{QB} + \mathcal{FB}$ forms a correctable error set in random mode. Furthermore, all sextuple bit errors in $\mathcal{SB}$ can be detected. In erasure mode we are left with a [32,16] binary code with a minimum bit distance of 7, so up to three bit errors can be corrected.

The generator matrix of formula (4) in which the matrix M of formula (15) is substituted is the binary image of the matrix $[\ 1\ \beta\ \beta^2\ ]$ with respect to the basis N_{15}.

Furthermore, it should be remarked that the [3,1] code with generator matrix $[\ 1\ \beta\ \beta^2\ ]$ also has minimum distance profile (12,7,1) with respect to several other normal bases of $GF(2^{16})$ over $GF(2)$.

Conclusion

In this paper a generalization of the TMR technique is discussed. The orthodox TMR technique applies a repetition code to correct single symbol errors. In systems with large memories, bit errors are predominant. The TMR technique does not give enough resistance against these errors. Hence, bit error-correcting codes are necessary to survive a module failure. But this causes the total memory size to grow by a factor four to four-and-a-half with respect to the single nonredundant configuration.

However, in our generalization we apply special rate one-third error-correcting codes that are able to correct symbol errors as well as multiple bit errors. Furthermore, whenever one module has failed we are left with a duplex configuration in which the punctured code still provides multiple bit error-correction capability. So this generalized TMR technique extends the orthodox TMR technique with a good resistance against multiple bit errors and the combination of a symbol erasure and bit errors without needing extra redundancy.

The idea of combined symbol and bit error-correction in processor-memory configurations originates from the '(N, K) concept' fault-tolerant computer described by Krol[7].

Acknowledgment

I would like to thank my colleague Th. Krol for many fruitful discussions on the (N, K) concept fault-tolerant computer.

References

[1] C.L. Chen, "Error-correcting codes with byte error-detection

capability", *IEEE Trans. Comput.*, vol. C-32, no. 7, pp. 615-621, July 1983.

[2] L.A. Dunning, M.R. Varanasi, "Code constructions for error control in byte organized memory systems", *IEEE Trans. Comput.*, vol. C-32, no. 6, pp. 535-542, June 1983.

[3] R.M.F. Goodman, R.J. McEliece, "Lifetime analyses of error-control coded semiconductor RAM systems", *IEE Proc.*, vol. 129, part E, no. 3, pp. 81-85, May 1982.

[4] H.J. Helgert, R.D. Stinaff, "Minimum-distance bounds for binary linear codes", *IEEE Trans. Inform. Theory*, vol. IT-19, no.3,pp. 344-356, May 1973.

[5] *Intel Memory Design Handbook*, pp. 4-13 - 4-33, Jan. 1981.

[6] S. Kaneda, "A class of odd-weight-column SEC-DED-SbED codes for memory system applications", *IEEE Trans. Comput.*, vol. C-33, no. 8, pp. 737-739, August 1984.

[7] Th. Krol, "The '(4,2) concept' fault-tolerant computer", *IEEE Fault Tolerant Computing Symposium* 12, Santa Monica, CA, June 1982, pp. 49-54.

[8] Th. Krol, B.J. Vonk, U.S. patent pending.

[9] F.J. MacWilliams, N.J.A. Sloane, *The Theory of Error-Correcting Codes*, Amsterdam: North-Holland 1978.

[10] R.J. McEliece, "The reliability of computer memories", *Scientific American*, pp. 68-73, Jan. 1985.

[11] P. Piret, "Binary codes for compound channels", *IEEE Trans. Inform. Theory*, vol. IT-31, no.3, pp. 436-440, May 1985.

[12] D.B. Sarrazin, M. Malek, "Fault-tolerant semiconductor memories", *Computer*, vol. 17, no. 8, pp. 49-56, August 1984.

[13] D.P. Siewiorek, R.S. Swarz, *The Theory and Practice of Reliable System Design*, Bedford, MA: Digital 1982.

3.2

An error-control coding system for storage of 16-bit words in memory arrays composed of three 9-bit wide units

Wil J. van Gils

Abstract

Error-correcting codes are widely used to improve the reliability of computer memories. The shift of VLSI technology towards higher levels of integration has resulted in multiple-bit-per-card and multiple-bit-per-chip memory structures. This paper describes codes for storing 16-bit words in a memory array consisting of three 9-bit wide memory units, a unit being a single card or a single chip. These codes are able to correct single bit errors, to detect up to four bit errors and to detect the failure of a complete memory unit. The codes have an elegant structure which makes fast decoding possible by simple means.

I. Introduction

Single-error-correcting, double-error-detecting (SEC-DED) binary codes are widely used to increase the reliability of computer memories having a one-bit-per-chip or one-bit-per-card structure [1,4,9]. However, the shift of VLSI technology towards higher levels of integration has resulted in multiple-bit-per-card and multiple-bit-per-chip memory structures [1,5,9]. Frequently occurring error events in such memory arrays are single cell failures due to impingement of atomic alpha particles. These cause transient single bit errors. Less frequent are permanent errors due to single cell, row, column, row-column or complete chip failures. These can produce single bit errors, but may also cause multiple bit errors in a single chip output.

Codes are therefore needed which correct/detect not only bit errors, but also errors caused by the failure of a complete chip or card [1,5]. For example IBM [9] produces 2K*9, 8K*9 and 32K*9 memory chips, which allow for a parity-check in each 8-bit word stored in it. This paper describes the construction and use of a class of [27,16] binary linear codes that encode 16 data bits into 27 code bits, which are stored in three 9-bit wide memory units. In [5], a similar code is described. It can correct single bit errors, detect double bit errors, and detect the failure of a complete chip. However, this code is not optimal and its lack of structure requires a rather complex decoder.

We have constructed a class of [27,16] codes which can correct single bit errors, detect up to four bit errors and detect single memory chip failures. The codes constructed are optimal in the sense that there does not exist any [27,16] code having better correction/detection properties. Our coding schemes also include simpler decoders using less hardware than the one described in [5].

In Section II we describe the construction and the properties of the codes. The decoders are described in Section III.

II. Construction and properties of the codes

Let α be a root of the primitive polynomial $x^8 + x^4 + x^3 + x^2 + 1$. Hence, α is a primitive element of the Galois field $GF(2^8)$. Define β to be equal to α^{85}, $\beta := \alpha^{85}$. The finite field $GF(2^8)$ has sixteen normal bases, namely

$$\mathcal{N}_b := \{\alpha^{b2^i} | i = 0, 1, \ldots, 7\}$$

for $b \in \mathcal{B} := \{5, 9, 11, 15, 21, 29, 39, 43, 47, 53, 55, 61, 63, 87, 91, 95\}$. For each of these normal bases $\mathcal{N}_b$, we define the 8 by 8 binary matrix $M_b = \{m_{ij}^{(b)}\}$ with $0 \leq i \leq 7, 0 \leq j \leq 7$, by

$$\beta\alpha^{b2^i} = \sum_{j=0}^{7} m_{ij}^{(b)} \alpha^{b2^j}, \; i = 0, 1, \ldots, 7.$$

This means that the ith row of M_b is the binary representation of $\beta\alpha^{b2^i}$ with respect to the basis $\mathcal{N}_b$. The matrix M_b is a primitive element of the field $GF(4)$, so that $M_b^3 = I$ and $I + M_b + M_b^2 = O$, where I denotes the identity matrix and O denotes the all-zero matrix. Furthermore, it can be readily seen that the row $(i+1)$ mod 7 of M_b^2 is equal to the ith row of M_b $(i = 0, 1, \ldots, 7)$. In [2] these matrices M_b were used to construct codes for the generalized Triple Modular Redundancy scheme. Here we shall use them to construct $[3 \star 9, 16]$ codes.

Let $\underline{p}^T(A)$ for a binary matrix A denote the column vector of row parities of A, i.e. $p(A)_i = \sum_j a_{ij}$. Define $C_b, b \in B$ to be the binary linear $[3 \star 9, 16]$ code with generator matrix

$$G_b := \begin{bmatrix} I & \underline{p}^T(I) & O & \underline{p}^T(O) & M_b & \underline{p}^T(M_b) \\ O & \underline{p}^T(O) & I & \underline{p}^T(I) & M_b^2 & \underline{p}^T(M_b^2) \end{bmatrix}.$$

We consider all codewords $\underline{c}$ in such a code to be composed of three symbols of nine bits: $\underline{c} = (\underline{c_1}, \underline{c_2}, \underline{c_3})$, where $\underline{c_1}, \underline{c_2}$ and $\underline{c_3}$ all have length nine.

To measure the error-correcting capacity of such codes we need the definition of the minimum distance profile of a code [2,8].

We denote by $wt(\underline{x})$ the Hamming weight of a binary vector $\underline{x}$, i.e., the number of components in $\underline{x}$ equal to 1. For a codeword $\underline{c} = (\underline{c_1}, \underline{c_2}, \underline{c_3})$, consisting of three symbols $\underline{c_1}, \underline{c_2}$ and $\underline{c_3}$ of equal size, *the weight profile* $\underline{wp}(\underline{c}) = (wp_0(\underline{c}), wp_1(\underline{c}), wp_2(\underline{c}))$ is defined by

$$\begin{aligned} wp_0(\underline{c}) &= wt(\underline{c}), \\ wp_1(\underline{c}) &= min\{wt(\underline{c_1}, \underline{c_2}), wt(\underline{c_1}, \underline{c_3}), wt(\underline{c_2}, \underline{c_3})\}, \\ wp_2(\underline{c}) &= min\{wt(\underline{c_1}), wt(\underline{c_2}), wt(\underline{c_3})\}. \end{aligned}$$

For a code C, consisting of such codewords, the *minimum distance profile* $\underline{d}(C) = (d(C|0), d(C|1), d(C|2))$ is defined by

$$d(C|i) = min\{wp_i(\underline{c}) : \underline{c} \in C \backslash \{\underline{0}\}\}, \quad i = 0, 1, 2.$$

In general for a code C, the set $\mathcal{T}$ of error patterns is correctable and the set $\mathcal{U}$ of error patterns is detectable if for all pairs $\underline{x} \in \mathcal{T} \cup \mathcal{U}$ and $\underline{y} \in \mathcal{T}$ their sum $\underline{x} + \underline{y}$ is not in C. In terms of the minimum distance profile this means that $\underline{wp}(\underline{x} + \underline{y})$ is not componentwise larger than or equal to $\underline{d}(C)$, or in other words there is at least one index i such that $wp_i(\underline{x} + \underline{y}) < d(C|i)$.

In [5], a [27,16] binary linear code is constructed which contains codewords of three symbols of nine bits, having minimum distance profile (4,2,0). This minimum distance profile guarantees the correction of single bit errors and the detection of single (9-bit) symbol errors and double bit errors. We shall now show that the $[3 \star 9, 16]$ codes constructed above have minimum distance profile (6,2,0), so that they guarantee correction of single bit errors, detection of single (9-bit) symbol errors and detection of up to four bit errors. We therefore need the following two lemmas.

Lemma 1. For $b \in \mathcal{A}_5 := \{5, 11, 15, 29, 47, 53, 63, 87\}$, the [16,8] binary linear code with generator matrix $[I\ M_b]$ has minimum bit distance 5.

Proof. Because of the special structure and the mutual relation between M_b and M_b^2, this can be demonstrated in a straightforward manner by merely checking the weight of ten codewords.

□

Lemma 2. For $b \in \mathcal{A}_4 := \{9, 21, 39, 43, 55, 61, 91, 95\}$, the [16,8] binary linear code with generator matrix $[I\ M_b]$ has minimum bit

distance 4, and the codewords of weight 4 have one component equal to one in the first eight positions and three components equal to one in the last eight positions, or vice versa.

Proof. Like the proof of Lemma 1, this can be demonstrated merely by checking ten codewords of the code.

□

Theorem 3. The $[3 \star 9, 16]$ codes $C_b, b \in \mathcal{A}_4 \cup \mathcal{A}_5$, have minimum distance profile (6,2,0).

Proof. Fix $b \in \mathcal{A}_4 \cup \mathcal{A}_5$, and define $M := M_b$, $C := C_b$ and $G := G_b$. From Lemmas 1 and 2, it follows that the [18,8] code with generator matrix $[I \ \underline{p}^T(I) \mid M \ \underline{p}^T(M)]$ has minimum bit distance 6. Since $[I \ M^2] = M^2[M \ I]$, the code with generator matrix $[I \ \underline{p}^T(I) \mid M^2 \ \underline{p}^T(M^2)]$ also has minimum bit distance 6.

Now let $\underline{c} = (\underline{c_1}, \underline{c_2}, \underline{c_3}) = (\underline{m_1}, \underline{m_2})G$ be a codeword of code C, where $\underline{c_1}, \underline{c_2}$, and $\underline{c_3}$ are binary vectors of length 9 and $\underline{m_1}$ and $\underline{m_2}$ are binary vectors of length 8. We distinguish three cases:

A. $\underline{m_1} \neq \underline{0}, \underline{m_2} = \underline{0}$. Then $\underline{c} = \underline{m_1}[I \ \underline{p}^T(I) \mid O \ \underline{p}^T(O) \mid M \ \underline{p}^T(M)]$. Hence $wt(\underline{c}) \geq 6$.

B. $\underline{m_1} = \underline{0}, \underline{m_2} \neq \underline{0}$. Then $\underline{c} = \underline{m_2}[O \ \underline{p}^T(O) \mid I \ \underline{p}^T(I) \mid M^2 \ \underline{p}^T(M^2)]$. Hence $wt(\underline{c}) \geq 6$.

C. $\underline{m_1} \neq \underline{0}, \underline{m_2} \neq \underline{0}$.

C1. If $\underline{c_3} \neq \underline{0}$, then $wt(\underline{c_i}) \geq 2$ for $i = 1, 2, 3$. So $wt(\underline{c}) \geq 6$.

C2. If $\underline{c_3} = \underline{0}$, then $\underline{m_1} = \underline{m_2}M$, and hence
$\underline{c} = (\underline{m_2}M \ \underline{p}^T(\underline{m_2}M) \mid \underline{m_2} \ \underline{p}^T(\underline{m_2}) \mid \underline{0} \ 0)$.
$wt(\underline{c}) = wt(\underline{m_2}[M, \ \underline{p}^T(M) \mid I \ \underline{p}^T(I)]) \geq 6$.

From these observations it follows that $d(C|0) = 6$. Furthermore it is easy to see that $d(C|1) = 2$ and $d(C|2) = 0$. Hence, the code C has minimum distance profile (6,2,0).

□

This minimum distance profile (6,2,0) implies that single bit errors are correctable and double, triple and quadruple bit errors and single symbol errors are detectable. To see this, define $\mathcal{T}$ to be the set of single bit errors and $\mathcal{U}$ to be the set of double, triple and quadruple bit errors and single symbol errors. Now it is easy to check that for any $\underline{x} \in \mathcal{T}$ and any $\underline{y} \in \mathcal{T} \cup \mathcal{U}$ there is at least one index $i \in \{0,1,2\}$ such that $wp_i(\underline{x}+\underline{y}|i) < d(C|i)$.

The $[3\star 9,16]$ binary linear codes with minimum distance profile (6,2,0) are optimal in the sense that any $[3 \star 9,16]$ binary linear code has a minimum distance profile (a,b,c) that satisfies $a \leq 6, b \leq 2, c = 0$. This is because the maximum minimum distance of a [27,16] binary linear code is 6, and the maximum minimum distance of a [18,16] binary linear code is 2 [3,6]. Furthermore, because of their very regular structure, the constructed codes C_b have very efficient decoders.

The 11 by 27 matrix

$$H_b = \left[\begin{array}{cccccc} 11111111 & 1 & 00000000 & 0 & 00000000 & 0 \\ 00000000 & 0 & 11111111 & 1 & 00000000 & 0 \\ 00000000 & 0 & 00000000 & 0 & 11111111 & 1 \\ & 0 & & 0 & & 0 \\ & 0 & & 0 & & 0 \\ M_b^T & . & (M_b^2)^T & . & I & . \\ & . & & . & & . \\ & 0 & & 0 & & 0 \end{array}\right]$$

is a parity-check matrix of the code C_b. This parity-check matrix can be used for syndrome decoding of the code. The 27 syndromes of single bit errors are used for correction. The remaining 2020 non-zero syndromes are used for detection of, among others, all single symbol errors and double, triple, and quadruple bit errors. The decoder can be implemented as is done in [5], using 27 AND-gates with 11 inputs. However, an implementation using less hardware is described in the next section.

III. Encoder and decoder implementation

In this section we describe the encoder and decoder implementation of the $[3 \star 9, 16]$ code C_9. A generator matrix G of this code is given in Figure 1. A parity-check matrix H of this code is given in Figure 2.

$$\left[\begin{array}{lll}
100000001 & 000000000 & 000010111 \\
010000001 & 000000000 & 110001010 \\
001000001 & 000000000 & 110000101 \\
000100001 & 000000000 & 011100010 \\
000010001 & 000000000 & 101100001 \\
000001001 & 000000000 & 010111000 \\
000000101 & 000000000 & 001011001 \\
000000011 & 000000000 & 000101110 \\
 & & \\
000000000 & 100000001 & 100010110 \\
000000000 & 010000001 & 100001011 \\
000000000 & 001000001 & 111000100 \\
000000000 & 000100001 & 011000011 \\
000000000 & 000010001 & 101110000 \\
000000000 & 000001001 & 010110001 \\
000000000 & 000000101 & 001011100 \\
000000000 & 000000011 & 000101101
\end{array}\right]$$

Figure 1: *Generator matrix of the code* C_9.

From Figure 2 we see that the parity-check matrix H has an elegantly structured form. We take advantage of this structured form in the decoder design. Define $\underline{s} = \underline{s}(\underline{r}) = (s_1, s_2, \ldots, s_{11}) = \underline{r}H^T$ to be the syndrome of an output vector $\underline{r}$ of the memory array. The syndromes $\{\underline{s} = \underline{e}H^T | wt(\underline{e}) = 1\}$ are used for single bit error correction. The remaining nonzero syndromes are used for error detection.

$$\left[\begin{array}{ccc}
111111111 & 000000000 & 000000000 \\
000000000 & 111111111 & 000000000 \\
000000000 & 000000000 & 111111111 \\
 & & \\
011010000 & 111010000 & 100000000 \\
011101000 & 001101000 & 010000000 \\
000110100 & 001110100 & 001000000 \\
000111010 & 000011010 & 000100000 \\
100001100 & 100011100 & 000010000 \\
010001110 & 010000110 & 000001000 \\
101000010 & 101000110 & 000000100 \\
110100010 & 110100000 & 000000010
\end{array}\right]$$

Figure 2: *Parity-check matrix of the code* C_9.

Define the signals A_0, A_1, A_2 and A_3 as follows:

$$\begin{aligned}
A_0 &= \overline{s_1}\,\overline{s_2}\,\overline{s_3}, \\
A_1 &= s_1\,\overline{s_2}\,\overline{s_3}, \\
A_2 &= \overline{s_1}\,s_2\,\overline{s_3}, \\
A_3 &= \overline{s_1}\,\overline{s_2}\,s_3,
\end{aligned}$$

where $\overline{x}$ denotes the inverse of x, $\overline{x} = 1 + x$. These can be implemented with NOR-gates with three inputs and inverters. If no error occurs, then $A_0 = 1$. If a single bit error occurs, then A_1, A_2 or A_3 indicate the symbol in which it occurs: $A_i = 1$ if this single bit error is in symbol i.

We now consider the vectors $\underline{u_j}, \underline{v_j}, \underline{w_j}$ of length 8 being the j^{th} columns of $M_b^T, (M_b^2)^T$ and I respectively. Since $M_b^2 = I + M_b$, it holds that $\underline{u_j}$ and $\underline{v_j}$ differ only in one position and their difference is exactly $\underline{w_j}$. We use this property in the construction of the decoder. For $j = 1$, for example,

$$\begin{aligned}
\underline{u_1} &= (00001011), \\
\underline{v_1} &= (10001011), \\
\underline{w_1} &= (10000000).
\end{aligned}$$

We now define the following functions of $\underline{s}$:

$$
\begin{aligned}
X_1 &:= (s_5 + s_6 + s_7 + s_9) = \overline{(\overline{s_5}\,\overline{s_6}\,\overline{s_7}\,\overline{s_9})},\\
Y_1 &:= (\overline{s_8} + \overline{s_{10}} + \overline{s_{11}}) = \overline{(s_8\, s_{10}\, s_{11})},\\
Z_1 &:= (s_8 + s_{10} + s_{11}) = \overline{(\overline{s_8}\,\overline{s_{10}}\,\overline{s_{11}})},\\
B_1 &:= \overline{(s_4 + X_1 + Y_1)},\\
C_1 &:= \overline{(\overline{s_4} + X_1 + Y_1)},\\
D_1 &:= \overline{(\overline{s_4} + X_1 + Z_1)},\\
SE_1 &:= A_1B_1 = s_1\,\overline{s_2}\,\overline{s_3}\,\overline{s_4}\,\overline{s_5}\,\overline{s_6}\,\overline{s_7}\,s_8\,\overline{s_9}\,s_{10}\,s_{11},\\
SE_{10} &:= A_2C_1 = \overline{s_1}\,s_2\,\overline{s_3}\,s_4\,\overline{s_5}\,\overline{s_6}\,\overline{s_7}\,s_8\,\overline{s_9}\,s_{10}\,s_{11},\\
SE_{19} &:= A_3D_1 = \overline{s_1}\,\overline{s_2}\,s_3\,s_4\,\overline{s_5}\,\overline{s_6}\,\overline{s_7}\,\overline{s_8}\,\overline{s_9}\,\overline{s_{10}}\,\overline{s_{11}}.
\end{aligned}
$$

Then it can be readily seen that,

if a bit error occurs at position 1, then $SE_1 = 1$,
if a bit error occurs at position 10, then $SE_{10} = 1$,
if a bit error occurs at position 19, then $SE_{19} = 1$.

An implementation of the above formulae B_1, C_1 and D_1 is given in Figure 3. The inputs of the box are the 8 bits $s_4, s_5, \ldots, s_{11}$ of the syndrome vector. The outputs of the box are the three bits B_1, C_1 and D_1. The contents of the box consist of three NAND gates and three NOR gates and are two gates deep. This box is called BOX 1. Other implementations with other gates are possible, we consider only one possibility. The signals $X_2, \ldots, X_8$, $Y_2, \ldots, Y_8$, $Z_2, \ldots, Z_8$, $B_2, \ldots, B_8$, $C_2, \ldots, C_8$, $D_2, \ldots, D_8$ and the signals $SE_2, \ldots, SE_8$, $SE_{11}, \ldots, SE_{17}$, $SE_{20}, \ldots, SE_{26}$ are defined similarly. Implementation of boxes BOX 2,...,BOX 8 also proceeds in the same way as BOX 1: For $j = 1, 2, \ldots, 8$, BOX j has inputs $s_4, s_5, \ldots, s_{11}$, outputs B_j, C_j, D_j, and contains three NAND gates and three NOR gates and is two gates deep.

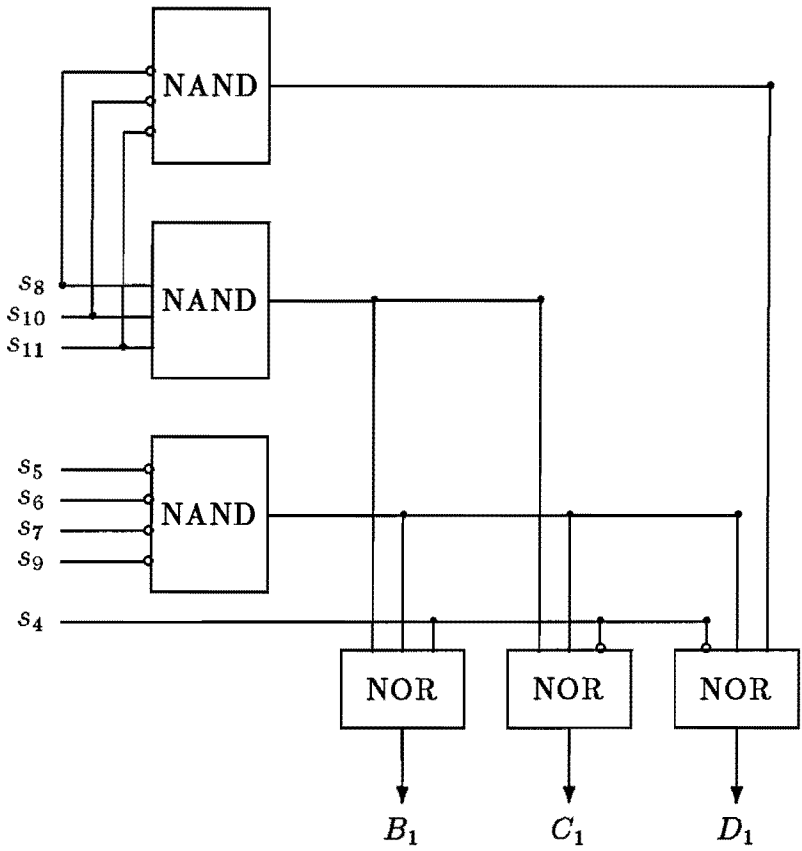


Figure 3: *Generation of the signals* B_1, C_1 *and* D_1.

The full decoder design is shown in Figure 4. Single bit errors are corrected because a single bit error in position j necessarily means that

$$SE_j = 1 \text{ and } SE_i = 0 \text{ for } i \neq j.$$

Double, triple and quadruple bit errors and single symbol errors are detected because their syndromes are non-zero and not equal to a syndrome of a single bit error, so they cause $\underline{s} \neq \underline{0}$ and $SE_j = 0$ for all $j = 1, \ldots, 27$.

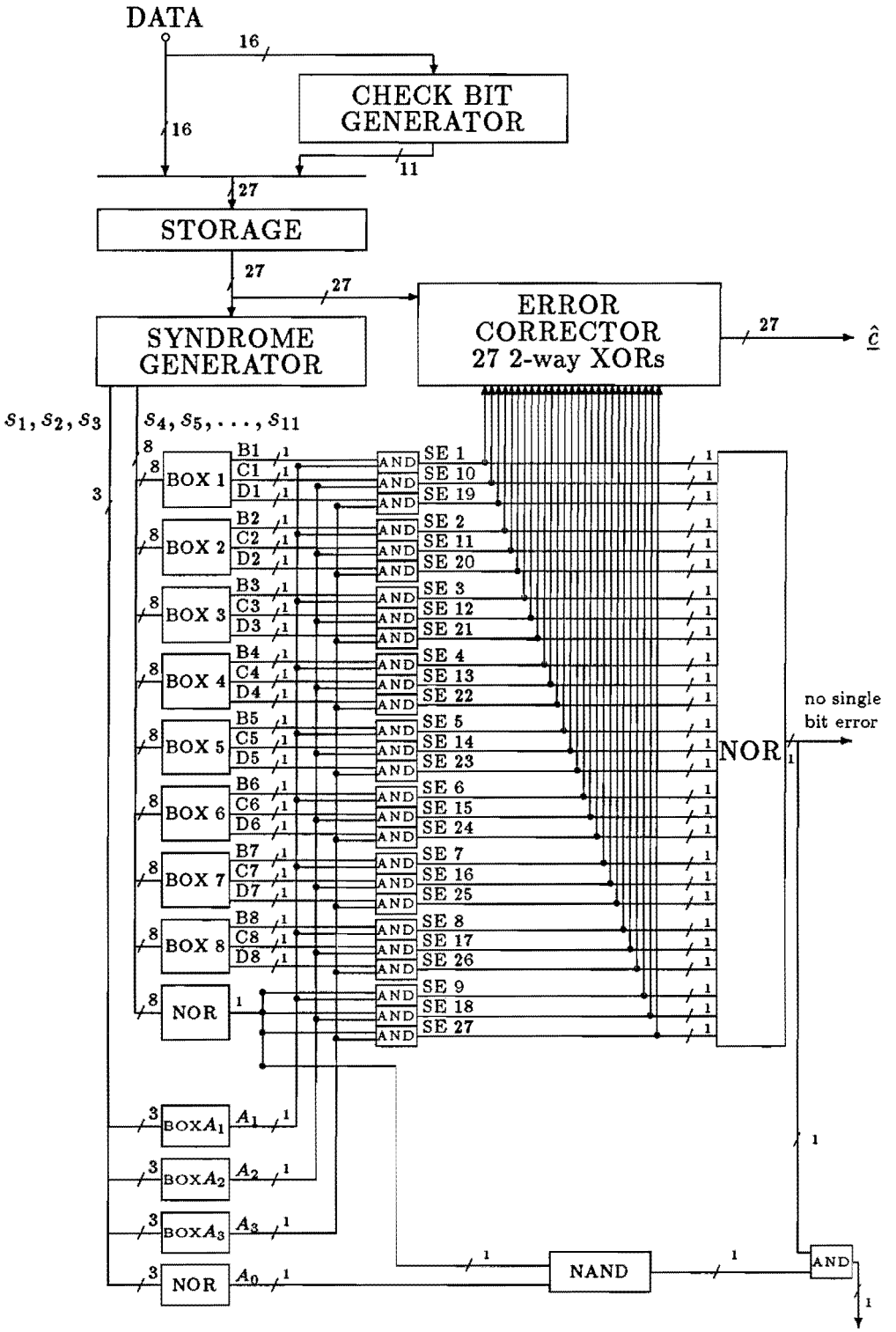


Figure 4: *Encoder/decoder design for the code C_9.*

References

[1] C.L. Chen, 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, March 1984.

[2] W.J. van Gils, "A triple modular redundancy technique providing multiple bit error protection without using extra redundancy", *IEEE Trans. Comput.*, vol. C-35, no.7, pp. 623-631, July 1986.

[3] H.J. Helgert, R.D. Stinaff, "Minimum-distance bounds for binary linear codes", *IEEE Trans. Inform. Theory*, vol. IT-19, no.3, pp. 344-356, May 1973.

[4] M.Y. Hsiao, "A class of optimal minimum odd-weight-column SEC-DED codes", *IBM J. Res. Develop.*, vol. 14, no. 4, pp. 395-401, July 1970.

[5] IBM, European Patent Application publication no. 0100825.

[6] F.J. MacWilliams, N.J.A. Sloane, *The Theory of Error-Correcting Codes*, Amsterdam: North-Holland, 1978.

[7] W.W. Peterson, E.J. Weldon jr., *Error-Correcting Codes*, 2nd ed., Cambridge, MA: MIT Press, 1972.

[8] P. Piret, "Binary codes for compound channels", *IEEE Trans. Inform. Theory*, vol. IT-31, no. 3, pp. 436-440, May 1985.

[9] D.B. Sarrazin, M. Malek, "Fault-tolerant semiconductor memories", *IEEE Computer*, vol. 17, no. 8, pp. 49-56, August 1984.

3.3

On combined symbol and bit error-control [4,2] codes over $\{0,1\}^8$ to be used in the (4,2) concept fault-tolerant computer

Wil J. van Gils Jean-Paul Boly

Abstract

This paper describes the construction, properties and decoding of four nonequivalent [4,2] codes over the alphabet $\{0,1\}^8$ that are able to correct the following error patterns:

- error patterns containing one nonzero byte,
- error patterns containing up to three nonzero bits,
- error patterns containing one byte erasure and at most one nonzero bit.

In addition all error patterns containing one byte erasure and two nonzero bits can be detected. These codes can be used in the (4,2) concept fault-tolerant totally interactive consistent computer ((4,2) FTTICC) and in memory systems composed of 8-bit wide chips or cards, where byte as well as bit errors occur.

I. Introduction

The construction, properties and decoding of four nonequivalent codes of length four and dimension two over the alphabet $\{0,1\}^8$ are described. These codes can be used in the (4,2) concept fault-tolerant totally interactive consistent computer ((4,2) FTTICC) [7,8,9] or in memory systems composed of byte-wide units such as chips or cards. In these systems byte errors as well as bit errors can occur, and both should be dealt with. The next paragraph briefly describes the basic internal functioning of the (4,2) concept computer. A more extensive description can be found in the papers by Krol [7,8]. The connection of the (4,2) concept computer to the outside world, especially interactively consistent input, is described by Krol and van Gils [9] and Krol [8]. The schematic

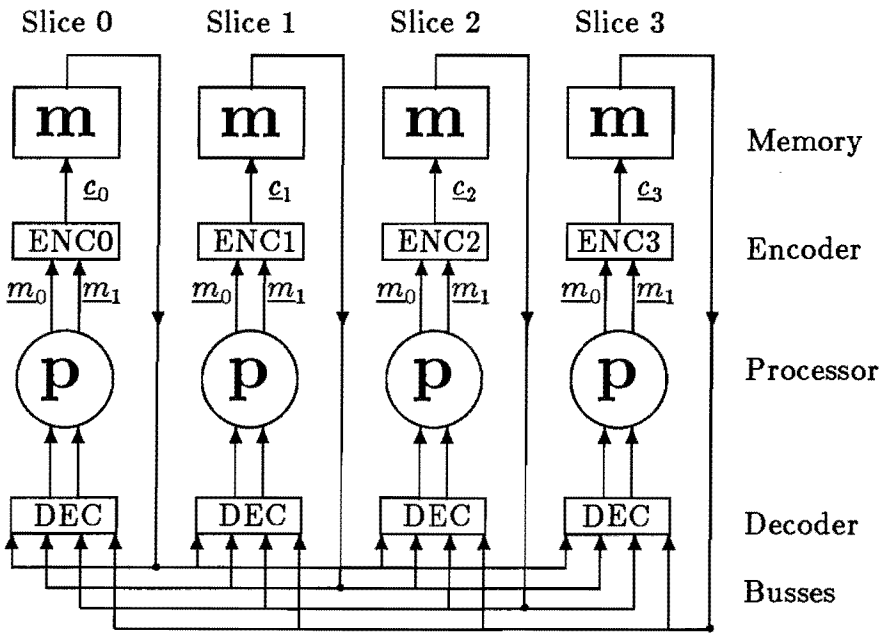


Figure 1: *The schematic architecture of the (4,2)-concept fault-tolerant computer.*

architecture of a (4,2) concept computer for a 16-bit processor is given in Figure 1. The (4,2) concept computer contains four processors doing exactly the same job synchronously. During a write action a processor does not, as usual, write a two-byte word (m_0, m_1) into its memory, but the two-byte word passes through an encoder circuit with a one-byte output. This one-byte output is stored in its memory. The encoder circuits are mutually different for the four processors, such that the concatenation (c_0, c_1, c_2, c_3) of their outputs is a codeword of a specially designed code of length four and dimension two over the alphabet $\{0,1\}^8$. During a read action, the four processors activate their memories and a possibly corrupted codeword is put onto the bus. This four-byte memory output is the input for four identical decoders, one attached to each processor. A decoder tries to remove the errors in its input. Its two-byte output is the input for the corresponding processor. A configuration of processor, encoder, memory, bus, decoder and necessary control logic is called a slice. The (4,2) concept computer thus contains four slices.

The constructed [4,2] codes over $\{0,1\}^8$ are able to correct the following four-byte (i.e. 32-bit) error patterns:

- error patterns containing one nonzero byte,
- error patterns containing up to three nonzero bits,
- error patterns containing one byte erasure and at most one nonzero bit.

In addition error patterns containing one byte erasure and two nonzero bits can be detected. In the (4,2) concept computer this error-control capacity is used in the following way. Any arbitrary failure of a piece of hardware within a slice causes at most one byte of a codeword to be in error. If the (error-prone) memories are composed of one-bit wide memory chips, the simultaneous failure of up to three memory chips can cause at most three bit errors. These kinds of errors can be corrected in the decoder, so these hardware failures have no influence on the correct behaviour of the computer as a whole. Furthermore, whenever the decoder detects a byte error in symbol i, then it immediately switches to a special mode, called erasure mode i. In that mode it considers

the ith symbol of its input to be an erasure. In erasure mode, a single bit error in one of the nonerased symbols can still be corrected, a double bit error can be detected. So during repair or while waiting for repair of a faulty slice, the computer system can still function correctly in the presence of single bit errors. These error masking capabilities of the error-control code make the (4,2) concept a very effective method for increasing the reliablity and availability of a single computer system [7,8,13].

The general (N,K) concept, which is a well-structured measure for reliability and availability improvement of computer systems, was described by Krol [7,8]. In references [7,8,10], [4,2] codes over $\{0,1\}^4$, i.e., an alphabet of 4-bit symbols, were constructed. These codes can correct one of the following error patterns: single 4-bit symbol errors, up to two bit errors, and single 4-bit symbol erasures in combination with a single bit error. Those codes were implemented in the (4,2) concept control computer in the Philips SOPHO-S telephone switch [13]. Reference [4] describes the (3,1) concept with combined symbol and bit error-control codes over alphabets of 4-, 8- and 16-bit symbols respectively. Reference [5] shows the application and construction of combined symbol and bit error-control [27,16] binary codes for error-control in memory systems constructed from 9-bit wide chips or cards. In reference [2], the general concept of combined symbol and bit error-control codes and the theory behind it are described. The present paper focusses on newly constructed [4,2] codes over $\{0,1\}^8$ having very nice properties.

We measure the error-control capacity of codes for combined control of symbol as well as bit errors by the so-called minimum distance profile [1,2,4,5,11], whose definition and properties will be given in Section II. This measure was introduced by Piret [11] under the name of compound distance profile. Section III describes the construction and the nice mathematical properties of four nonequivalent [4,2] codes over $\{0,1\}^8$. In Section IV we describe a decoder outline for these codes when they are used in the (4,2) concept fault-tolerant computer.

II. Definition and properties of the minimum distance profile

In this section we shall define the minimum distance profile of a code and we shall indicate the error detection/correction capacities induced by this minimum distance profile. The concept of (compound) distance profile was introduced by Piret [11]. The definitions and properties in this section partly coincide with those in reference [11].

Consider a linear $[n, k]$ code C of length n and dimension k over the Galois field $\mathbf{F} := GF(2^m)$ having minimum Hamming distance S. The elements of the Galois field $\mathbf{F}$ are called symbols. They can be represented by binary m-vectors with respect to a basis of $GF(2^m)$ over $GF(2)$ [12,Chapter 4,Section 8]. The weight $wt(x)$ of a symbol x of $\mathbf{F}$ with respect to the chosen vector representation is defined as the number of ones in its vector representation $\underline{x} = (x_0, x_1, \ldots, x_{m-1})$. The *symbol weight* of a vector of symbols is defined as the number of nonzero symbol positions in that vector. The *bit weight* $wt(\underline{y})$ of a vector $\underline{y} = (y_0, y_1, \ldots, y_{n-1})$ of symbols of $\mathbf{F}$ is defined as the sum of the weights of its symbol components,

$$wt(\underline{y}) = \sum_{i=0}^{n-1} wt(y_i).$$

Analogously we distinguish between the *(minimum) symbol* and *(minimum) bit distance* of a code.

For a vector $\underline{c} = (c_0, c_1, \ldots, c_{n-1}) \in \mathbf{F}^n$ let $wt(\underline{c}|j)$ denote the bit weight of the vector of length $n - j$ over $\mathbf{F}$ that is obtained by deleting the j heaviest (i.e., those having the largest weights) symbols from $\underline{c}$. The *weight profile* of an n-vector $\underline{c} = (c_0, c_1, \ldots, c_{n-1})$ of symbols of $\mathbf{F}$ is given by the n-vector

$$\begin{aligned} \underline{wp}(\underline{c}) &= (wp_0(\underline{c}), wp_1(\underline{c}), \ldots, wp_{n-1}(\underline{c})) \\ &:= (wt(\underline{c}|0), wt(\underline{c}|1), \ldots, wt(\underline{c}|n-1)). \end{aligned}$$

The *minimum distance profile* $\underline{d}(C)$ of the code C is given by the S-vector

$$\underline{d}(C) := (d_0(C), d_1(C), \ldots, d_{S-1}(C)),$$

where its components are defined by

$$d_j(C) := minimum\{wt(\underline{c}|j) : \underline{c} \in C, \underline{c} \neq \underline{0}\},\ j = 0, 1, ..., S-1.$$

In other words, if for fixed j one takes $n-j$ arbitrary symbol components of an arbitrary nonzero codeword in the code C, then the sum of the weights of these components is at least $d_j(C)$. There also exists a codeword in the code C having $n-j$ components whose weights add up to exactly $d_j(C)$. Hence, $d_j(C)$ is the minimum of the minimum bit distances of the $\binom{n}{j}$ codes obtained by puncturing C in j symbol positions. Note that the length of the weight profile is equal to the length n of the code C and that the length of the minimum distance profile is equal to the minimum symbol distance S of the code C.

A finite set $\mathcal{A} := \{(a_1, b_1), (a_2, b_2), \ldots, (a_{|\mathcal{A}|}, b_{|\mathcal{A}|})\}$ of pairs of natural numbers is called a *list of pairs* if all first components $a_i, i = 1, \ldots, |\mathcal{A}|$ of the pairs in $\mathcal{A}$ are mutually different. We say that a vector $\underline{x}$ is covered by $\mathcal{A}$, $\mathcal{A}$ being a list of pairs, if there exists a pair (a, b) in $\mathcal{A}$ such that $wt(\underline{x}|a) \leq b$.

Let $\mathcal{T}$ and $\mathcal{U}$ be two lists of pairs. The linear code C is said to be *$\mathcal{T}$-correcting and $\mathcal{U}$-detecting* if it corrects all errors $\underline{e}$ covered by $\mathcal{T}$ and if it detects all errors $\underline{e}$ which are covered by $\mathcal{U}$ and which are not covered by $\mathcal{T}$. If $\mathcal{U} = \emptyset$, then C is called *$\mathcal{T}$-correcting*. If $\mathcal{T} = \emptyset$, then C is called *$\mathcal{U}$-detecting*. For example, a {(1,0),(0,2)}-correcting and {(0,3)}-detecting code corrects all single symbol errors and all double bit errors, and detects all triple bit errors. It is immediately clear that C is $\mathcal{T}$-correcting and $\mathcal{U}$-detecting if and only if

1. none of its cosets contains more than one vector covered by $\mathcal{T}$,

   and

2. if a coset does contain a vector covered by $\mathcal{T}$ then it does not contain a vector that is covered by $\mathcal{U}$ and is not covered by $\mathcal{T}$.

This is equivalent with saying that the difference of two different words of which one is covered by $\mathcal{T}$ and the other is covered by $\mathcal{T}$ or $\mathcal{U}$, is never a codeword. The following property is a direct consequence of these remarks.

Theorem 0. The code C is $\mathcal{T}$-correcting and $\mathcal{U}$-detecting if and only if, for any pair (s,t) in $\mathcal{T}$ and any pair (u,v) in the union of $\mathcal{T}$ and $\mathcal{U}$, the component $d_{s+u}(C)$ of $\underline{d}(C)$ satisfies the inequality

$$d_{s+u}(C) \geq t + v + 1.$$

Proof. Sufficiency: Suppose for all $(s,t) \in \mathcal{T}$ and $(u,v) \in \mathcal{T} \cup \mathcal{U}$ we have that $d_{s+u}(C) \geq t + v + 1$. We have to show that C is $\mathcal{T}$-correcting and $\mathcal{U}$-detecting. So let $\underline{e_1}$ and $\underline{e_2}$ be two different error patterns covered by $\mathcal{T}$ and $\mathcal{T} \cup \mathcal{U}$ respectively. This means that there exist pairs $(s_1,t_1) \in \mathcal{T}$ and $(s_2,t_2) \in \mathcal{T} \cup \mathcal{U}$ such that $wt(\underline{e_1}|s_1) \leq t_1$ and $wt(\underline{e_2}|s_2) \leq t_2$. So, trivially we have that $wt(\underline{e_1}+\underline{e_2}|s_1+s_2) \leq t_1+t_2 < d_{s_1+s_2}(C)$, which means that $\underline{e_1}+\underline{e_2}$ is not a nonzero codeword of C. Hence no difference of two different words, of which one is covered by $\mathcal{T}$ and the other is covered by $\mathcal{T}$ or $\mathcal{U}$, is a codeword of C.
Necessity: Suppose the code C is $\mathcal{T}$-correcting and $\mathcal{U}$-detecting. Let (s,t) and (u,v) be pairs covered by $\mathcal{T}$ and $\mathcal{T} \cup \mathcal{U}$ respectively. Let $\underline{c}$ be a nonzero codeword of C. Assume that $wt(\underline{c}|s+u) \leq t+v$. This means that the vector $\underline{c}$ is equal to a sum $\underline{e_1}+\underline{e_2}$ of two vectors $\underline{e_1}$ and $\underline{e_2}$ that satisfy the following two inequalities,

$$wt(\underline{e_1}|s) \leq t \text{ and } wt(\underline{e_2}|u) \leq v.$$

Hence the vector $\underline{e_1}$ is covered by $\mathcal{T}$ and the vector $\underline{e_2}$ is covered by $\mathcal{T} \cup \mathcal{U}$. In other words, $\underline{e_1}$ is a correctable error pattern and $\underline{e_2}$ is a correctable or detectable error pattern, while their sum $\underline{e_1}+\underline{e_2}$ is a codeword of C, a contradiction. Hence for all nonzero codewords $\underline{c}$ in C we have that $wt(\underline{c}|s+u) \geq t+v+1$ and consequently we have that $d_{s+u}(C) \geq t+v+1$. This completes the proof.

□

If in every codeword of a code C, e fixed symbol positions ($e \leq S-1$) are erased then we obtain the set of codewords of an $[n-e,k]$ code C' over $GF(2^m)$ with minimum distance profile

$\underline{d}(C') = (d_0(C'), \ldots, d_{S'}(C'))$, where S' denotes the minimum symbol distance of C' and where $d_j(C')$ satisfies the inequality

$$d_j(C') \geq d_{j+e}(C).$$

For example the [4,2] codes over $GF(2^4)$ constructed by Krol and Vonk [5,6,10] have minimum distance profile (5,3,1) and so by Theorem 0 they are {(1,0),(0,2)}-correcting. The punctured [3,2] codes over $GF(2^4)$ have minimum distance profile (3,1) and are {(0,1)}-correcting. The [4,2] codes constructed in this paper have minimum distance profile (7,4,1) and so by Theorem 0 they are {(1,0),(0,3)}-correcting. The punctured [3,2] codes have minimum distance profile (4,1) and are {(0,1)}-correcting and {(0,2)}-detecting. For other examples, see Boly [1], Boly and van Gils [2], van Gils [4,5] and Piret [11].

The following bound turns out to be very useful in finding the minimum distance profile of a code.

Theorem 1. For all $j = 1, ..., S-1$ we have that

a. $wt(\underline{c}|j) \geq (n-j)wt(\underline{c}|j+1)/(n-j-1)$ for all $\underline{c} \in C$,

and

b. $d_j(C) \geq (n-j)d_{j+1}(C)/(n-j-1)$.

Proof By definition, the average bit weight per symbol, $wt(\underline{c}|j)/(n-j)$, is a nonincreasing function of j.

□

In Boly and van Gils [2] the concept of the minimum distance profile of a code is elaborated to a wider class of codes than the ones considered in this paper.

III. Construction and properties of the codes

In this section we describe the four nonequivalent [4,2] codes over $GF(2^8)$ that have minimum distance profile (7,4,1) with respect to a normal basis of $GF(2^8)$ over $GF(2)$. For two of them we can easily prove that the minimum distance profile equals (7,4,1). For the other two the minimum distance profile was calculated with the help of a computer. In fact these four codes are the only nonequivalent codes over $GF(2^8)$ that have minimum distance profile (7,4,1) with respect to a normal basis of $GF(2^8)$ over $GF(2)$.

For an element $\gamma \in GF(2^m)$ and a basis $B = \{\beta_0, \beta_1, \ldots, \beta_{m-1}\}$ of $GF(2^m)$ over $GF(2)$, $\underline{\gamma(B)} = (\gamma_0, \gamma_1, \ldots, \gamma_{m-1})$ will denote the vector representation of γ with respect to B, i.e.

$$\gamma = \sum_{i=0}^{m-1} \gamma_i \beta_i, \ \gamma_i \in GF(2), i = 0, 1, ..., m-1.$$

For a vector $\underline{x} = (x_0, x_1, \ldots, x_{n-1}) \in GF(2^m)^n$, $\underline{x}(B)$ is defined by $\underline{x}(B) := (\underline{x_0(B)}, \underline{x_1(B)}, \ldots, \underline{x_{n-1}(B)})$. For a linear $[n,k]$ code C over $GF(2^m)$, $\underline{C(B)}$ denotes the binary linear $[nm, km]$ code $\{\underline{c(B)} : \underline{c} \in C\}$.

Now let α be a root of $x^8+x^4+x^3+x^2+1$. Then α is a primitive element of $GF(2^8)$. Define β to be α^{17} ,so β is a primitive element of $GF(2^4)$ satisfying the equation $\beta^4 + \beta + 1 = 0$. Define C to be the [4,2] code over $GF(2^8)$ with generator matrix

$$G_C = \begin{bmatrix} \beta^3 & \beta^6 & \beta^{12} & \beta^9 \\ \beta^6 & \beta^{12} & \beta^9 & \beta^3 \end{bmatrix}.$$

Then the matrix

$$H_C = \begin{bmatrix} \beta^{14} & \beta^{13} & \beta^{11} & \beta^7 \\ \beta^{13} & \beta^{11} & \beta^7 & \beta^{14} \end{bmatrix}$$

is a parity-check matrix of C. Note that $\{\beta^3, \beta^6, \beta^{12}, \beta^9\}$, being the elements in the matrix G_C, and $\{\beta^7, \beta^{14}, \beta^{13}, \beta^{11}\}$, being the

elements in the matrix H_C, are complementary normal bases of $GF(2^4)$ over $GF(2)$. The code C has symbol distance 3 and the binary subfield subcode of C consists of the zero codeword only. Let $\mathcal{N}(i)$ denote the normal basis $\{\alpha^{i2^k} : k = 0, 1, \ldots, 7\}$ of $GF(2^8)$ over $GF(2)$. The field $GF(2^8)$ has 16 normal bases, namely $\mathcal{N}(i)$ for $i \in \mathcal{NB} := \{5, 9, 11, 15, 21, 29, 39, 43, 47, 53, 55, 61, 63, 87, 91, 95\}$.

The codes $C(\mathcal{N}(i))$ and $C^{\perp}(\mathcal{N}(i))$, $i \in \mathcal{NB}$, have the property that the simultaneous cyclic shifts of the four bytes over one byte position and of the eight bits within a byte over one bit position in a codeword again give a codeword of the code. This property is shown in the next theorem.

Theorem 2. For the codes $C(\mathcal{N}(j))$ (respectively $C^{\perp}(\mathcal{N}(j))$), $j \in \mathcal{NB}$, we have that for every codeword

$$\underline{c} = (c_{00}, c_{01}, \ldots, c_{07}, c_{10}, c_{11}, \ldots, c_{17}, c_{20}, c_{21}, \ldots, c_{27}, c_{30}, c_{31}, \ldots, c_{37})$$

in $C(\mathcal{N}(j))$ (respectively $C^{\perp}(\mathcal{N}(j))$) also its shift

$$(c_{37}, c_{30}, \ldots, c_{36}, c_{07}, c_{01}, \ldots, c_{06}, c_{17}, c_{10}, \ldots, c_{16}, c_{27}, c_{20}, \ldots, c_{26})$$

is a codeword of $C(\mathcal{N}(j))$ (respectively $C^{\perp}(\mathcal{N}(j))$).

Proof. For a codeword $\underline{c} \in C(\mathcal{N}(j))$ we have that

$$\sum_{u=0}^{3} \sum_{v=0}^{7} c_{uv} (\beta^{14})^{i2^u} \alpha^{j2^v} = 0 \text{ for } i = 1, 2.$$

By squaring these equations we get that

$$\sum_{u=0}^{3} \sum_{v=0}^{7} c_{uv} (\beta^{14})^{i2^{u+1}} \alpha^{j2^{v+1}} = 0 \text{ for } i = 1, 2.$$

This proves the theorem for $C(\mathcal{N}(j))$. The proof for $C^{\perp}(\mathcal{N}(j))$ is similar.

□

We shall now show that $C_1 := C(\mathcal{N}(11))$ has minimum distance profile (7,4,1). We need the following lemmas.

Lemma 3 (Davenport [3]). For integer m let η be an element of

$GF(2^{2^m})$. Then $\{\eta^{2^j} : j = 0, 1, ..., 2^m - 1\}$ is a normal basis of $GF(2^{2^m})$ over $GF(2)$ if and only if

$$Tr(\eta) := \sum_{i=0}^{2^m-1} \eta^{2^i} = 1.$$

Let $CC(i) := \{\alpha^i, \alpha^{2i}, \alpha^{4i}, \ldots, \alpha^{128i}\}$ denote the conjugate classes of $GF(2^8)\backslash\{0\}$. For these conjugate classes we have the following lemma.

Lemma 4. Let η_1 and η_2 be distinct elements of a conjugate class $CC(i)$ of $GF(2^8)\backslash\{0\}$. If $\eta_1\eta_2^{-1} \in GF(2^4)$, then $CC(i) \subset GF(2^4)$.

Proof. For distinct elements η_1, η_2 of $CC(i)$ there are integers a and b, $0 \leq a, b \leq 7, a \neq b$ such that $\eta_1 = \alpha^{i2^a}, \eta_2 = \alpha^{i2^b}$. If $\eta_1\eta_2^{-1}$ is in $GF(2^4)$, then $i(2^a - 2^b) = 0 \bmod 17$. This is only possible if i is a multiple of 17. But then $CC(i)$ is a subset of $GF(2^4)$.

□

Table I gives the representation of $GF(2^8) = GF(2)(\alpha)$ with respect to the normal basis $\mathcal{N}(11)$. From this table it can be seen that the elements in $GF(2^8)$ of weight two with respect to $\mathcal{N}(11)$ are exactly the elements of $CC(1), CC(13), CC(119)$ and $CC(127)$. Note that $CC(119)$ is equal to $\{\beta^7, \beta^{14}, \beta^{13}, \beta^{11}\}$. For these conjugate classes we have the following lemma.

Lemma 5. For $\eta_1 \in \mathcal{N}(11)$ and $\eta_2 \in CC(1) \cup CC(13) \cup CC(119) \cup CC(127)$ we have that $\eta_1\eta_2^{-1} \notin GF(2^4)$.

Proof. It can easily be checked that $(i2^a - 11) \neq 0 \bmod 17$ for $i = 1, 13, 127$ and $a = 0, 1, \ldots, 7$. Furthermore, $CC(119) \subset GF(2^4)$ and $\mathcal{N}(11) \cap GF(2^4) = \emptyset$.

□

Now we can prove our main theorem.

Theorem 6. The code $C_1 = C(\mathcal{N}(11))$ has minimum distance profile $(d_0, d_1, d_2) = (7, 4, 1)$.

i	bin.rep.	i	bin.rep.	i	bin.rep.	i	bin.rep.	i	bin.rep.
0	11111111	51	01110111	102	10111011	153	11101110	204	11011101
1	11000000	52	01010000	103	10101001	154	10001010	205	10110011
2	01100000	53	01000101	104	00101000	155	11011001	206	11010100
3	10110111	54	11101011	105	01000111	156	01010111	207	01001100
4	00110000	55	11101100	106	10100010	157	01101010	208	00010100
5	11110010	56	11000011	107	10011110	158	10000011	209	00101100
6	11011011	57	10101011	108	11110101	159	00100110	210	10100011
7	00011110	58	01100001	109	11110100	160	10010111	211	00011100
8	00011000	59	00110101	110	01110110	161	00001010	212	01010001
9	10110101	60	10010100	111	11010010	162	00110100	213	11000111
10	01111001	61	11000001	112	11100001	163	00010110	214	01001111
11	10000000	62	11100010	113	11011000	164	01100101	215	11011111
12	11101101	63	00010011	114	11010101	165	11010001	216	11111010
13	01000001	64	00000011	115	11001110	166	00101010	217	10100110
14	00001111	65	11001011	116	10110000	167	00001110	218	01111010
15	01010010	66	11010110	117	00011111	168	11010000	219	01001011
16	00001100	67	00000101	118	10011010	169	10101000	220	00111011
17	10011001	68	01100110	119	00010001	170	10101010	221	01000100
18	11011010	69	00011010	120	01001010	171	11100011	222	01101001
19	10011100	70	11111100	121	10111001	172	10010001	223	01001000
20	10111100	71	00001011	122	11100000	173	10100111	224	11110000
21	10000110	72	10110110	123	01011010	174	11111000	225	10010010
22	01000000	73	10110010	124	01110001	175	11101111	226	01101100
23	10001101	74	01010110	125	11111101	176	00001000	227	00101110
24	11110110	75	11101000	126	10001001	177	01111101	228	11101010
25	00111111	76	00100111	127	00010010	178	01100100	229	01101110
26	10100000	77	00010101	128	10000001	179	01010011	230	01100111
27	11010111	78	10101110	129	01101111	180	10001110	231	10011000
28	10000111	79	00000111	130	11100101	181	00111101	232	01011000
29	11000010	80	00101111	131	00111100	182	11101001	233	00111000
30	00101001	81	01101000	132	01101011	183	10100101	234	10001111
31	11000101	82	11001010	133	00000001	184	10110001	235	10111111
32	00000110	83	01010100	134	10000010	185	10011101	236	01001101
33	10101101	84	10100001	135	10100100	186	00111110	237	10010110
34	11001100	85	01010101	136	00110011	187	00100010	238	10001000
35	11111001	86	00100011	137	00111001	188	01110011	239	10010000
36	01101101	87	11110001	138	00001101	189	10110100	240	00100101
37	10101100	88	00010000	139	00011011	190	11111011	241	01011100
38	01001110	89	11001000	140	01111110	191	00100100	242	11011100
39	01011101	90	00011101	141	10101111	192	11011110	243	00110001
40	01011110	91	11010011	142	10000101	193	01111000	244	01110000
41	10010101	92	01100011	143	10001011	194	00000010	245	01111111
42	01000011	93	01111100	144	01011011	195	01001001	246	00101101
43	01000110	94	11100110	145	11110011	196	01110010	247	00100001
44	00100000	95	11110111	146	01011001	197	00110110	248	10111000
45	00111010	96	10111101	147	10111010	198	01011111	249	01100010
46	11000110	97	00000100	148	00101011	199	00010111	250	11111110
47	11001101	98	11100100	149	10001100	200	11100111	251	01000010
48	01111011	99	10111110	150	01110100	201	01110101	252	11000100
49	11001001	100	11001111	151	10011011	202	00011001	253	10000100
50	10011111	101	00110010	152	10010011	203	00110111	254	00001001

Table I: *The binary representation (bin.rep) of α^i with respect to the basis $\mathcal{N}(11)$*

Proof. $d_2 = 1$: Clearly $d_2 = 1$, because C has symbol distance 3 and C is linear over $GF(2^m)$.

$d_1 = 4$: Because the minimum bit distance of a [24,16] binary linear code is at most 4 [6], we have that $d_1 \leq 4$. Let C' be the code over $GF(2^8)$ obtained by puncturing C in the first symbol. The code C' has parity-check matrix

$$H' = \begin{bmatrix} 1 & \beta^6 & \beta^8 \end{bmatrix}.$$

By Theorem 2 we have that d_1 is equal to the minimum bit distance of $C'(\mathcal{N}(11))$. We split the problem by first considering codewords of C' having symbol weight two, and then considering those having symbol weight three.

* **A.1**: Let α^i and α^j be the two distinct nonzero symbols of a codeword $\underline{c'}$ in $C'(\mathcal{N}(11))$ of symbol weight two. Hence, $\alpha^{i-j} \in GF(2^4)$. By Lemmas 4 and 5 it follows that $\underline{c'}$ has a bit weight of at least 4.

* **A.2:** Let $(\gamma_1, \gamma_2, \gamma_3)$ be a codeword of $C'(\mathcal{N}(11))$ having symbol weight three. We have that $\gamma_1 + \gamma_2\beta^6 + \gamma_3\beta^8 = 0$ and so

$$Tr(\gamma_1) + Tr(\gamma_2\beta^6) + Tr(\gamma_3\beta^8) = 0.$$

We want to show that γ_1, γ_2 and γ_3 cannot all three be members of $\mathcal{N}(11)$. Assume $\gamma_1, \gamma_2, \gamma_3$ are three elements of $\mathcal{N}(11)$. Using Lemma 3 it is easy to check that

$$\begin{aligned} Tr(\gamma_1) &= 1, \\ Tr(\gamma_2\beta^6) &= 1 \quad \text{iff} \quad (\gamma_2 = \alpha^{11\,2^i} \text{ for } i = 3 \text{ or } 7), \\ Tr(\gamma_3\beta^8) &= 1 \quad \text{iff} \quad (\gamma_3 = \alpha^{11\,2^i} \text{ for } i = 0, 1, 4 \text{ or } 5). \end{aligned}$$

By Lemma 4 and the fact that $(\gamma_1, \gamma_2, \gamma_3) \in C'$ we have that $\gamma_2 \neq \gamma_3$. We have to check whether $\gamma_2\beta^6 + \gamma_3\beta^8$ can be an element of $\mathcal{N}(11)$. Due to the above properties and the fact that $\beta \in GF(2^4)$, the 64 possiblilities for the pair (γ_2, γ_3) reduce to the following 13:

$$(\alpha^{11\,2^i}, \alpha^{11\,2^j}) \text{ for } (i,j) \in$$
$$(\{3\} \star \{2,6,7\}) \cup (\{0\} \star \{1,4,5\}) \cup (\{1\} \star \{0,4,5\}) \cup (\{2\} \star \{0,1,4,5\}).$$

None of these possibilities make $\gamma_2\beta^6 + \gamma_3\beta^8$ into an element of $\mathcal{N}(11)$. Hence $(\gamma_1, \gamma_2, \gamma_3)$ has bit weight at least 4.
From A.1 and A.2 we may conclude that $d_1 \geq 4$.

$d_0 = 7$: By Theorem 1 we have that $d_0 \geq 4d_1/3$, and hence $d_0 \geq 6$.

* **B.1:** Let $\underline{c}$ be a codeword of $C(\mathcal{N}(11))$ of symbol weight 3. Due to Theorem 2 we may take the first symbol equal to zero, that is $\underline{c} = (0, \gamma_1, \gamma_2, \gamma_3) = \theta(0, 1, \beta^{14}, \beta^2)$ for some $\theta \in GF(2^8)$. Assume that $\underline{c}$ has bit weight 6. Then by part A. of this proof it follows that γ_1, γ_2 and γ_3 should all have bit weight two, i.e. they must be elements of $\mathcal{V} = CC(1) \cup CC(13) \cup CC(119) \cup CC(127)$.

* **B.1.1:** Assume that one of $\gamma_1, \gamma_2, \gamma_3$ is in $CC(119)$. Because $CC(119) \subset GF(2^4)$, this means that $\theta \in GF(2^4)$ and hence all three $\gamma_1, \gamma_2, \gamma_3$ are elements of $GF(2^4)$. They are also elements of $\mathcal{V}$, so all three are elements of $\mathcal{V} \cap GF(2^4) = CC(119)$. Hence there are integers a and b, $0 \leq a, b \leq 3$ such that $\theta = \beta^{7\,2^a}$ and $\gamma_2 = \beta^{7\,2^b}$. From $\gamma_2 = \theta\beta^{14}$ it follows that $(2^a + 2) = 2^b \bmod 15$. The unique solution is $a = 1, b = 2$. But then $\gamma_3 = \beta^{14}\beta^2 = \beta \notin CC(119)$. This is in contradiction with $\gamma_3 \in CC(119)$. So none of the elements γ_1, γ_2 or γ_3 is in $CC(119)$.

* **B.1.2:** Assume that γ_1 is in $CC(1)$, so $\gamma_1 = \alpha^{2^k}$ for some $k \in \{0, 1, \ldots, 7\}$. Hence $\gamma_2 = \alpha^{2^k - 17}$. Furthermore by B.1.1 and Lemma 4 we have that γ_2 is an element of $CC(13) \cup CC(127)$. There are two choices for k such that $\gamma_2 \in CC(13) \cup CC(127)$, namely $k = 0$ and $k = 4$. But, if $k = 0$ then $\gamma_3 = \alpha^{35} \notin CC(1) \cup CC(13) \cup CC(127)$ and if $k = 4$ then $\gamma_3 = \alpha^{50} \notin CC(1) \cup CC(13) \cup CC(127)$. From this we may conclude that $\gamma_1 \notin CC(1)$.

* **B.1.3:** Analogously as in B.1.2 we can show that $\gamma_1 \notin CC(13)$ and $\gamma_1 \notin CC(127)$.

From B.1 and the fact that $d_0 \geq 6$ we may conclude that a codeword of $C(\mathcal{N}(11))$ having symbol weight 3, has a bit weight of at least 7.

* **B.2:** Let $\underline{c} = (\gamma_0, \gamma_1, \gamma_2, \gamma_3)$ be a codeword of $C(\mathcal{N}(11))$ of symbol weight 4. Assume that it has bit weight 6. Then without loss of generality (Theorem 2) we can have the following two possibilities for the weights of $\gamma_0, \gamma_1, \gamma_2$ and γ_3: γ_0 and γ_1 have bit weight 1 and γ_2 and γ_3 have bit weight 2, or γ_0 and γ_2 have bit weight 1 and γ_1 and γ_3 have bit weight 2.

* **B.2.1:** Assume γ_0 and γ_1 have bit weight 1 and γ_2 and γ_3 have bit weight 2. Because $\underline{c} = (\gamma_0, \gamma_1, \gamma_2, \gamma_3) \in C$, we have $\underline{c}H_C^T = \underline{0}$,

i.e.,

$$(I): \begin{cases} \gamma_2 = \gamma_0\beta^{11} + \gamma_1\beta^{14} \\ \gamma_3 = \gamma_0\beta^{9} + \gamma_1\beta^{2}. \end{cases}$$

Because the bit weights of γ_2 and γ_3 are two we have that $Tr(\gamma_2) = Tr(\gamma_3) = 0$ and so combining this with the equations in (I) we get

$$(II): \begin{cases} Tr(\gamma_0\beta^{11}) + Tr(\gamma_1\beta^{14}) = 0 \\ Tr(\gamma_0\beta^{9}) + Tr(\gamma_1\beta^{2}) = 0. \end{cases}$$

The solutions to (II) are $(\gamma_0, \gamma_1) = (\alpha^{11\,2^i}, \alpha^{11\,2^j})$, $(i,j) \in (\{1,5\} \star \{3,7\}) \cup (\{2,3,6,7\} \star \{0,1,4,5\})$. By straightforward checking it can be seen that none of these pairs substituted in (I) gives a γ_2 of bit weight 2 (because $\beta \in GF(2^4)$, we only have to do 10 checks). This is in contradiction with our assumption that γ_2 had bit weight 2.

* **B.2.2:** The case that γ_0 and γ_2 have bit weight 1 and γ_1 and γ_3 have bit weight 2 is similar to B.2.1.

From B.2 and the fact that $d_0 \geq 6$ we may conclude that a codeword of $C(\mathcal{N}(11))$ having symbol weight 4, has a bit weight of at least 7.

From B.1 and B.2 we conclude that $d_0 \geq 7$. Furthermore $(0, \alpha^{88}, \alpha^{71}, \alpha^{122})$ is a codeword of $C(\mathcal{N}(11))$ with weight profile (7,4,1,0), so we have that $C(\mathcal{N}(11))$ has minimum distance profile (7,4,1).

□

In the same way as in Theorem 6 we can show the following result.

Theorem 7. The code $C_2 = C^{\perp}(\mathcal{N}(43))$ has minimum distance profile (7,4,1).

Also the codes $C(\mathcal{N}(47))$ and $C^{\perp}(\mathcal{N}(95))$ have minimum distance profile (7,4,1). In fact $C(\mathcal{N}(47))$ is equivalent to $C_1 = C(\mathcal{N}(11))$ and $C^{\perp}(\mathcal{N}(95))$ is equivalent to $C_2 = C^{\perp}(\mathcal{N}(43))$ [1,2]. Two combined symbol-and-bit error-control codes are called equivalent if one can be obtained from the other by applying a permutation on the symbols and (possibly mutually different) permutations on the bits within a symbol [1,2].

We have done a computer search to find all [4,2] codes over $GF(2^8)$ that have a minimum distance profile of at least (7,4,1)

with respect to a normal basis of $GF(2^8)$ over $GF(2)$. By using the tables in [6] we see that a [4,2,3] code over $GF(2^8)$ has a minimum distance profile (d_0, d_1, d_2) with $d_0 \leq 8$, $d_1 \leq 4$ and $d_2 = 1$. The computer search yielded four nonequivalent [4,2] codes over $GF(2^8)$ with a minimum distance profile of (7,4,1). Two of them have already been mentioned, C_1 and C_2. The other two are the codes C_3 and C_4 which are the binary images with respect to the normal basis $\mathcal{N}(43)$ of the codes with parity-check matrices

$$H_3 = \begin{bmatrix} 1 & 0 & \beta & \beta^2 \\ 0 & 1 & \beta^6 & \beta \end{bmatrix}$$

and

$$H_4 = \begin{bmatrix} 1 & 0 & \beta & \beta^3 \\ 0 & 1 & \beta^2 & \beta^{12} \end{bmatrix}$$

respectively. A number of theorems about equivalences between combined symbol-and-bit error-control codes used in this search can be found in [1,2].

IV. Decoder outline

We describe the decoder for the code C_3 to be used in a (4,2) concept fault-tolerant computer. The decoders for the other codes are similar. The code C_3 has generator matrix

$$G = \begin{bmatrix} M & M^6 & I & O \\ M^2 & M & O & I \end{bmatrix}$$

and a 'redundant' parity-check matrix (redundant because the rank of the matrix equals half the size of the matrix)

$$H = \begin{bmatrix} O & I & (M^T)^6 & M^T \\ I & O & M^T & (M^T)^2 \\ (M^T)^6 & M^T & O & I \\ M^T & (M^T)^2 & I & O \end{bmatrix},$$

where

$$M = \begin{bmatrix} 11001011 \\ 11110011 \\ 11010010 \\ 11101100 \\ 10111100 \\ 00111111 \\ 00101101 \\ 11001110 \end{bmatrix}$$

is the 8 by 8 binary matrix whose ith row is the binary representation of $\beta\alpha^{43\cdot 2^i}$ with respect to the basis $\mathcal{N}(43)$ $(i = 0, 1, \ldots, 7)$. The (redundant) syndrome of a corrupted codeword $\underline{r} = (\underline{r}_0, \underline{r}_1, \underline{r}_2, \underline{r}_3) = (\underline{c}_0, \underline{c}_1, \underline{c}_2, \underline{c}_3) + (\underline{e}_0, \underline{e}_1, \underline{e}_2, \underline{e}_3)$ is defined by $\underline{s} = \underline{r}H^T$.

To describe the decoding principles we need to define a number of sets. The set $(\star 000)$ is defined by

$$(\star 000) := \{(\underline{x}, \underline{0}, \underline{0}, \underline{0}) : \underline{x} \in \mathbf{F}, \underline{x} \neq \underline{0}\},$$

where $\underline{0}$ denotes the 8-bits allzero vector. In an analogous way $(0 \star 00), (00 \star 0)$ and $(000\star)$ are defined. The union of these four sets builds the set of all single symbol errors. The sets $(ijkm), i, j, k, m \in \{0, 1, 2\}$ are defined by

$$(ijkm) := \{(\underline{u}, \underline{v}, \underline{w}, \underline{x}) : \underline{u}, \underline{v}, \underline{w}, \underline{x} \in \mathbf{F},$$

$$wt(\underline{u}) = i, wt(\underline{v}) = j, wt(\underline{w}) = k, wt(\underline{x}) = m\}.$$

The union of the sets $(ijkm)$ for which $i + j + k + m$ is a fixed constant c builds the set of all error patterns of bit weight c. An erasure will be denoted by 'E'. The sets $(Ejkm)$, $j, k, m \in \{0, 1\}$ are defined by

$$(Ejkm) := \{(\underline{u}, \underline{v}, \underline{w}, \underline{x}) : \underline{u}, \underline{v}, \underline{w}, \underline{x} \in \mathbf{F},$$

$$wt(\underline{v}) = j, wt(\underline{w}) = k, wt(\underline{x}) = m\}.$$

Note that in this set no restriction on $\underline{u}$ is made because it is considered to be an erasure. In an analogous way $(iEjk), (ijEk)$ and $(ijkE)$ are defined. The sets defined above are called *error classes.*

For the code C_3 the set of correctable error patterns is the union of the error classes given in the first column of Table II. For a fixed error class in the set $\mathcal{T}$ of correctable error patterns, the syndromes of its elements have some property in common. For example, all error patterns $\underline{e}$ in the error class $(\star 000)$ have in common that their syndromes $\underline{s} = (\underline{s_0}, \underline{s_1}, \underline{s_2}, \underline{s_3})$ satisfy $\underline{s_0} = \underline{0}, \underline{s_1} \neq \underline{0}, \underline{s_2} \neq \underline{0}, \underline{s_3} \neq \underline{0}$. All error patterns $\underline{e}$ in the error class $(i0j0)$ have in common that their syndromes satisfy $wt(\underline{s_0}M^{-6}) = j$ and $wt(\underline{s_2}M^{-6}) = i$. We translate these properties shared by elements in an error class into boolean expressions that have to be satisfied and use them in the decoding process. The second column of Table II gives these boolean expressions. To any error class there corresponds one expression such that all error patterns in that error class satisfy the expression and do not satisfy any of the other expressions in Table II. For an error class containing elements of which at least two symbols are zero, the boolean expression is very easy. If it is estimated that an error pattern is in such an error class, then the estimation of the message is easy too, because at least two symbols of the corrupted codeword determine the corresponding message uniquely (the code is maximum 'symbol' distance separable). For error classes containing elements of which three symbols have weight one, it is somewhat more complicated. In these situations we have to determine the bit error in one of the symbols. When this is done we have two correct codeword symbols, so we can estimate the message.

To construct the above-mentioned boolean expressions we need the definition of the boolean variables given in Table III. By $\underline{u_j}$ we denote the binary vector of length 8 having a one in the jth position and zeros elsewhere. Table IV gives possible estimates for the message. The estimate $\underline{\hat{m}_{ij}}$ is derived from the received symbols $\underline{r_i}$ and $\underline{r_j}$. The estimate $\underline{\hat{n}_{ij}}$ is derived from two received symbols in which one of them has one bit flipped. The last column of Table II gives the estimates of the message for all error classes. Note that for the error classes with three symbols of weight one, the estimate depends on the estimate of the position of the bit error in one of the symbols. For example, an error pattern $\underline{e}$ in (1100) satisfies the boolean expression $RMf_{01}f_{11} = 1$ and the message is estimated by $\underline{\hat{m}_{23}}$. An error pattern $\underline{e}$ in $(10E0)$ satisfies

the boolean expression $EM_2h_{21} = 1$ and the message is estimated by $\underline{\hat{m}_{13}}$. An error pattern $\underline{e}$ in (1101) satisfies exactly one of the boolean expressions $RMt_{3j} = 1, j = 0, \ldots, 7$ and can be estimated by $\underline{\hat{n}_{3j}}$.

From Table II we can see that in fact the decoder should perform the rules given in Table V. The decoder also has a mode register, containing the values of RM, EM_i, and SM_{ij} for $i, j = 0, 1, 2, 3, i \neq j$. After a decoding step the mode register should be updated. One of the possible strategies could be to switch from random mode to erasure mode when a single symbol error occurs that is not a single bit error. Other switching steps strongly depend on the error statistics of the entire system and will not be discussed here. Note that in single mode, that is only two symbols of a codeword are observed, error detection is no longer possible. In the decoder other kinds of registers could be implemented, as for example registers storing the positions and frequencies of bit errors [10]. A global decoder design implementing the rules given in Table V is given in Figure 2.

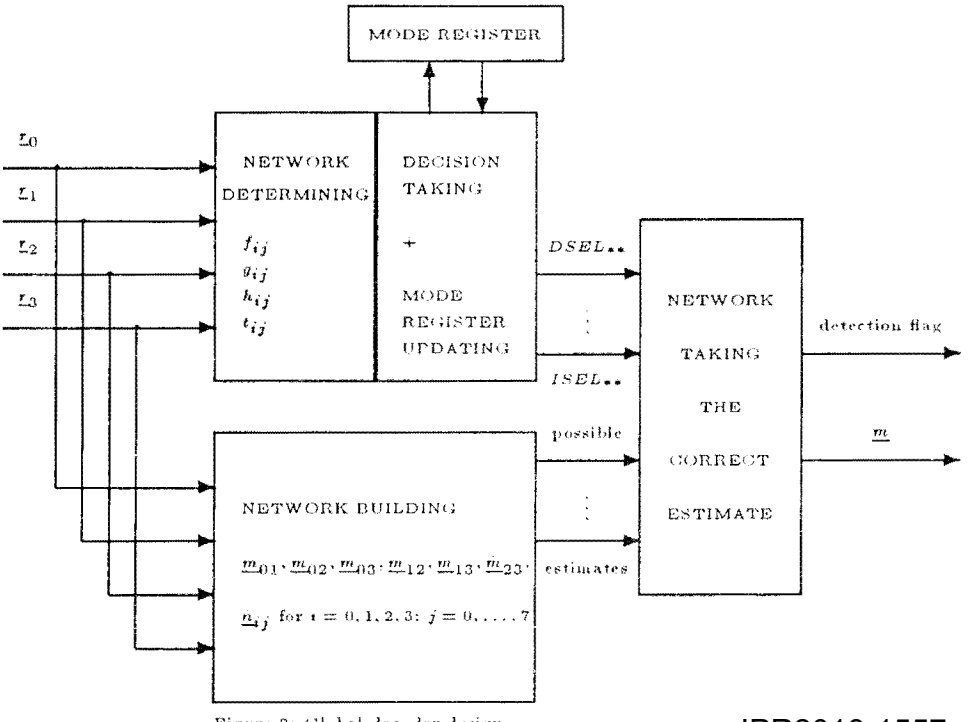


Figure 2: Global decoder design

error class	boolean expression	estimate
0000	$RMf_{00}f_{10}$	$\underline{\hat{m}_{23}}$
$\star$000	$RMf_{00}\overline{f_{10}}\overline{f_{20}}\overline{f_{30}}$	$\underline{\hat{m}_{23}}$
0$\star$00	$RM\overline{f_{00}}f_{10}\overline{f_{20}}\overline{f_{30}}$	$\underline{\hat{m}_{23}}$
00$\star$0	$RM\overline{f_{00}}\overline{f_{10}}f_{20}\overline{f_{30}}$	$\underline{\hat{m}_{01}}$
000$\star$	$RM\overline{f_{00}}\overline{f_{10}}\overline{f_{20}}f_{30}$	$\underline{\hat{m}_{01}}$
1100	$RMf_{01}f_{11}$	$\underline{\hat{m}_{23}}$
1010	$RMh_{01}h_{21}$	$\underline{\hat{m}_{13}}$
1001	$RMg_{01}g_{31}$	$\underline{\hat{m}_{12}}$
0110	$RMg_{11}g_{21}$	$\underline{\hat{m}_{03}}$
0101	$RMh_{11}h_{31}$	$\underline{\hat{m}_{02}}$
0011	$RMf_{21}f_{31}$	$\underline{\hat{m}_{01}}$
2100	$RMf_{01}f_{12}$	$\underline{\hat{m}_{23}}$
2010	$RMh_{01}h_{22}$	$\underline{\hat{m}_{13}}$
2001	$RMg_{01}g_{32}$	$\underline{\hat{m}_{12}}$
1200	$RMf_{02}f_{11}$	$\underline{\hat{m}_{23}}$
0210	$RMg_{11}g_{22}$	$\underline{\hat{m}_{03}}$
0201	$RMh_{11}h_{32}$	$\underline{\hat{m}_{02}}$
1020	$RMh_{02}h_{21}$	$\underline{\hat{m}_{13}}$
0120	$RMg_{12}g_{21}$	$\underline{\hat{m}_{03}}$
0021	$RMf_{21}f_{32}$	$\underline{\hat{m}_{01}}$
1002	$RMg_{01}g_{30}$	$\underline{\hat{m}_{12}}$
0102	$RMh_{12}h_{31}$	$\underline{\hat{m}_{02}}$
0012	$RMf_{22}f_{31}$	$\underline{\hat{m}_{01}}$
1110	RMt_{2j}	$\underline{\hat{n}_{2j}}$
1101	RMt_{3j}	$\underline{\hat{n}_{3j}}$
1011	RMt_{0j}	$\underline{\hat{n}_{0j}}$
0111	RMt_{1j}	$\underline{\hat{n}_{1j}}$
E000	EM_0f_{00}	$\underline{\hat{m}_{23}}$
E100	EM_0f_{01}	$\underline{\hat{m}_{23}}$
E010	EM_0h_{01}	$\underline{\hat{m}_{13}}$
E001	EM_0g_{01}	$\underline{\hat{m}_{12}}$
0E00	EM_1f_{10}	$\underline{\hat{m}_{23}}$
1E00	EM_1f_{11}	$\underline{\hat{m}_{23}}$
0E10	EM_1g_{11}	$\underline{\hat{m}_{03}}$
0E01	EM_1h_{11}	$\underline{\hat{m}_{02}}$
00E0	EM_2f_{20}	$\underline{\hat{m}_{01}}$
10E0	EM_2h_{21}	$\underline{\hat{m}_{13}}$
01E0	EM_2g_{21}	$\underline{\hat{m}_{03}}$
00E1	EM_2f_{21}	$\underline{\hat{m}_{01}}$
000E	EM_3f_{30}	$\underline{\hat{m}_{01}}$
100E	EM_3g_{31}	$\underline{\hat{m}_{12}}$
010E	EM_3h_{31}	$\underline{\hat{m}_{02}}$
001E	EM_3f_{31}	$\underline{\hat{m}_{01}}$

Table II: *The correctable error classes, the corresponding boolean expressions and the corresponding estimates.*

$$
\begin{array}{lllll}
f_{ij} = 1 & \Leftrightarrow & wt(\underline{s_i}) = j & i = 0,1,2,3, & j = 0,1,2. \\
g_{ij} = 1 & \Leftrightarrow & wt(\underline{s_i}M^{-1}) = j & i = 0,1,2,3, & j = 1,2. \\
h_{ij} = 1 & \Leftrightarrow & wt(\underline{s_i}M^{-2}) = j & i = 1,3, & j = 1,2. \\
h_{ij} = 1 & \Leftrightarrow & wt(\underline{s_i}M^{-6}) = j & i = 0,2, & j = 1,2.
\end{array}
$$

$$
\begin{array}{lllll}
t_{0j} = 1 & \Leftrightarrow & wt(\underline{s_2} + \underline{u_j}M^6) = 1 & \text{and} & wt(\underline{s_3} + \underline{u_j}M) = 1, \\
t_{1j} = 1 & \Leftrightarrow & wt(\underline{s_2} + \underline{u_j}M) = 1 & \text{and} & wt(\underline{s_3} + \underline{u_j}M^2) = 1, \\
t_{2j} = 1 & \Leftrightarrow & wt(\underline{s_0} + \underline{u_j}M^6) = 1 & \text{and} & wt(\underline{s_1} + \underline{u_j}M) = 1, \\
t_{3j} = 1 & \Leftrightarrow & wt(\underline{s_0} + \underline{u_j}M) = 1 & \text{and} & wt(\underline{s_1} + \underline{u_j}M^2) = 1, \\
 & & & & \text{for } j = 0,1,\ldots,7.
\end{array}
$$

$RM = 1$ if and only if the decoder is running in random mode, i.e., all outputs of the four slices are considered in the decoding process.

$EM_i = 1$ if and only if the decoder is running in erasure mode i, i.e., slice i is considered to be malfunctioning (producing an erasure).

$SM_{ij} = 1$ if and only if the decoder is running in single mode, i.e., only the slices i and j are considered to be functioning correctly.

Table III: *Definition of boolean variables.*

$$\underline{\hat{m}_{01}} := (\underline{r_0}, \underline{r_1}) \begin{bmatrix} M & M^6 \\ M^2 & M \end{bmatrix}$$

$$\underline{\hat{m}_{02}} := (\underline{r_0}, \underline{r_2}) \begin{bmatrix} O & M^{13} \\ I & M^{14} \end{bmatrix}$$

$$\underline{\hat{m}_{03}} := (\underline{r_0}, \underline{r_3}) \begin{bmatrix} M^{14} & O \\ M & I \end{bmatrix}$$

$$\underline{\hat{m}_{12}} := (\underline{r_1}, \underline{r_2}) \begin{bmatrix} O & M^{14} \\ I & M^5 \end{bmatrix}$$

$$\underline{\hat{m}_{13}} := (\underline{r_1}, \underline{r_3}) \begin{bmatrix} M^9 & O \\ M^{10} & I \end{bmatrix}$$

$$\underline{\hat{m}_{23}} := (\underline{r_2}, \underline{r_3})$$

$$\underline{\hat{n}_{0j}} := (\underline{r_0} + \underline{u_j}, \underline{r_1}) \begin{bmatrix} M & M^6 \\ M^2 & M \end{bmatrix} \quad j = 0, 1, \ldots, 7$$

$$\underline{\hat{n}_{1j}} := (\underline{r_0}, \underline{r_1} + \underline{u_j}) \begin{bmatrix} M & M^6 \\ M^2 & M \end{bmatrix} \quad j = 0, 1, \ldots, 7$$

$$\underline{\hat{n}_{2j}} := (\underline{r_2} + \underline{u_j}, \underline{r_3}) \qquad j = 0, 1, \ldots, 7$$

$$\underline{\hat{n}_{3j}} := (\underline{r_2}, \underline{r_3} + \underline{u_j}) \qquad j = 0, 1, \ldots, 7$$

Table IV: *Estimates for the message.*

* Estimate $\underline{\hat{m}} = \underline{\hat{m}_{01}}$ if

$$DSEL_{01} := \left\{ \begin{array}{l} RM(\overline{f_{00}}\,\overline{f_{10}}f_{20}\overline{f_{30}} + \overline{f_{00}}\,\overline{f_{10}}\,\overline{f_{20}}f_{30} \\ \quad + f_{21}f_{31} + f_{21}f_{32} + f_{22}f_{31}) \\ + EM_2(f_{20} + f_{21}) + EM_3(f_{30} + f_{31}) + SM_{01} \end{array} \right\} = 1.$$

* Estimate $\underline{\hat{m}} = \underline{\hat{m}_{02}}$ if

$$DSEL_{02} := \left\{ \begin{array}{l} RM(h_{11}h_{31} + h_{11}h_{32} + h_{12}h_{31}) \\ + EM_1h_{11} + EM_3h_{31} + SM_{02} \end{array} \right\} = 1.$$

* Estimate $\underline{\hat{m}} = \underline{\hat{m}_{03}}$ if

$$DSEL_{03} := \left\{ \begin{array}{l} RM(g_{11}g_{21} + g_{11}g_{22} + g_{12}g_{21}) \\ + EM_1g_{11} + EM_2g_{21} + SM_{03} \end{array} \right\} = 1.$$

* Estimate $\underline{\hat{m}} = \underline{\hat{m}_{12}}$ if

$$DSEL_{12} := \left\{ \begin{array}{l} RM(g_{01}g_{31} + g_{01}g_{32} + g_{02}g_{31}) \\ + EM_0g_{01} + EM_3g_{31} + SM_{12} \end{array} \right\} = 1.$$

* Estimate $\underline{\hat{m}} = \underline{\hat{m}_{13}}$ if

$$DSEL_{13} := \left\{ \begin{array}{l} RM(h_{01}h_{21} + h_{01}h_{22} + h_{02}h_{21}) \\ + EM_0h_{01} + EM_2h_{21} + SM_{13} \end{array} \right\} = 1.$$

* Estimate $\underline{\hat{m}} = \underline{\hat{m}_{23}}$ if

$$DSEL_{23} := \left\{ \begin{array}{l} RM(f_{00}f_{10} + f_{00}\overline{f_{10}}\,\overline{f_{20}}\,\overline{f_{30}} + \overline{f_{00}}f_{10}\overline{f_{20}}\,\overline{f_{30}} \\ \quad + f_{01}f_{11} + f_{01}f_{12} + f_{02}f_{11}) \\ + EM_0(f_{00} + f_{01}) + EM_1(f_{10} + f_{11}) + SM_{23} \end{array} \right\} = 1.$$

* Estimate $\underline{\hat{m}} = \underline{\hat{n}_{ij}}$ if

$$ISEL_{ij} := RMt_{ij} = 1 \quad i = 0,1,2,3,\ j = 0,1,\ldots,7.$$

$$\text{detection flag} := \overline{(\textstyle\sum_{i,j} DSEL_{ij} \ + \ \sum_{i,j} ISEL_{ij})}$$

$$\overline{(RMf_{10}f_{20} + EM_0f_{00} + EM_1f_{10} + EM_2f_{20} + EM_3f_{30})}.$$

Table V: *Estimates for the message and the conditions therefor.*

References

[1] J.P. Boly, *On Combined Symbol-and-Digit Error-Control Codes*, M.Sc. Thesis Eindhoven University of Technology, November 1986.

[2] J.P. Boly and W.J. van Gils, "Codes for combined symbol and digit error-control", submitted for publication to *IEEE Trans. Inform. Theory*

[3] H. Davenport, "Bases for finite fields", *J. London Math. Soc.*, vol. 43, pp. 21-39, 1968.

[4] W.J. van Gils, "A triple modular redundancy technique providing multiple-bit error protection without using extra redundancy", *IEEE Trans. on Comput.*, vol. C-35, no. 7, pp. 623-631, July 1986.

[5] W.J. van Gils, "An error-control coding system for storage of 16-bit words in memory arrays composed of three 9-bit wide units", *Philips J. of Res.*, vol. 41, no. 4, pp.391-399, 1986.

[6] H.J. Helgert and R.D. Stinaff, "Minimum-distance bounds for binary linear codes", *IEEE Trans. Inform. Theory*, vol. IT-19, pp. 344-356, May 1973.

[7] T. Krol, "The '(4,2)-concept' fault-tolerant computer", *Dig. 12th Int. Symp. Fault-Tolerant Computing*, pp. 49-54, Santa Monica CA, June 1982.

[8] T. Krol, "(N,K) concept fault-tolerance", *IEEE Trans. Comput.* , vol. C-35, no.4, pp. 339-349, April 1986.

[9] T. Krol and W.J. van Gils, "The input/output architecture of the (4,2) concept fault-tolerant computer", *Proc. 15th Intern. Symp. Fault-Tolerant Computing*, pp. 254-259, Ann Arbor MI, June 1985.

[10] T. Krol and B.J. Vonk, U.S. patent pending.

[11] P. Piret, "Binary codes for compound channels", *IEEE Trans. Inform. Theory*, vol. IT-31, no. 3, pp. 436-440, May 1985.

[12] F.J. MacWilliams and N.J.A. Sloane, *The Theory of Error-Correcting Codes*, Amsterdam: North-Holland ,1978.

[13] *SOPHOMATION : The total approach to information management*, Special issue of *Philips Telecomm Rev.*, vol. 43, no. 2, June 1985.

3.4

Codes for combined symbol and digit error-control

Jean-Paul Boly Wil J. van Gils

Abstract

This paper describes an approach towards error-control coding for systems in which digit as well as symbol errors and erasures can occur, a symbol being a position-fixed group of digits. Codes that can deal with both types of errors and erasures simultaneously are constructed. A theoretical basis for the determination of the error-control capacity of such codes is given.

I. Introduction

This paper describes an approach towards error-control coding for systems in which digit as well as symbol errors can occur, where a symbol is a position-fixed group of digits. Examples of such systems are the (N, K)-concept fault-tolerant computers [6,8,12,13,14], compound channels [23,24] and memory systems composed of m-bit wide chips where m is larger than one [2,4,5,7,11,26,29].

Consider a linear code C of composite length $n = Nm$ and dimension k over the Galois field $GF(q)$. The vectors $\underline{x} \in (GF(q))^{Nm}$ are considered to be composed of N elements of $(GF(q))^m$:

$$\underline{x} = (\underline{x_0}, \underline{x_1}, \ldots, \underline{x_{N-1}}), \quad \underline{x_i} \in (GF(q))^m, \quad i = 0, 1, \ldots, N-1. \quad (1)$$

The components $\underline{x_i}$ of $\underline{x}$ are called the *symbols* of $\underline{x}$. The components $x_{ij} \in GF(q)$ of $\underline{x_i} = (x_{i0}, x_{i1}, \ldots, x_{i(m-1)})$ are called the *digits* of $\underline{x}$. We are interested in codes that can detect and correct digit errors and erasures, symbol errors and erasures and combinations of these. These codes will be called combined Symbol and Digit Error-Control (SDEC) codes. For example, in the $(4, 2)$-concept and $(3, 1)$-concept (generalized Triple Modular Redundancy) fault-tolerant computers [6,8,12,13], these codes are used for correction of single symbol errors, multiple bit errors and the combination of a single symbol erasure and bit errors. References [2,4,5,11,26,29] provide constructions of single-bit error-correcting, single-byte error-detecting, double-bit error-detecting (SEC-BED-DED) codes for use in memory systems composed of byte-wide units. Piret [23,24] considers the use of SDEC codes on a memoryless compound channel, being an approximation of the two-state Gilbert channel.

In Section II we introduce the concept of the minimum distance profile of a code, a good measure for determining the error-control capacity of an SDEC code. In Section III a number of constructions of SDEC codes are given. The parameters of these SDEC codes are summarized in the Tables IV-IX. Earlier constructions of SDEC codes were published by Krol [12,13], Piret [23,24], van Gils [6,7] and van Gils and Boly [8].

II. Definition and Properties of the Minimum Distance Profile

In this section we shall define the minimum distance profile of a code and we shall indicate the error detection/correction capacities induced by it. The original definition was introduced by Piret [23].

Consider a linear code C of length $n = Nm$ and dimension k over $GF(q)$. The vectors $\underline{x}$ in $(GF(q))^{Nm}$ are considered to be partitioned into N blocks (symbols) of m q-ary digits each, i.e.

$$\underline{x} = (\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_{N-1}), \quad \underline{x}_i = (x_{i0}, x_{i1}, \ldots, x_{i(m-1)}) \in (GF(q))^m. \tag{2}$$

The parameters N, n, k and m are called *symbol code length*, *digit code length*, *digit code dimension* and *symbol size* respectively. The *symbol weight* $sw(\underline{x})$ of $\underline{x} \in (GF(q))^{Nm}$ is defined to be the number of nonzero symbols in $\underline{x}$:

$$sw(\underline{x}) := |\{i : \underline{x}_i \neq \underline{0},\ i = 0, 1, \ldots, N-1\}|. \tag{3}$$

The *symbol distance* $sd(\underline{x}, \underline{y})$ between two elements $\underline{x}$ and $\underline{y}$ in $(GF(q))^{Nm}$ is defined to be the symbol weight of their difference $\underline{x} - \underline{y}$:

$$sd(\underline{x}, \underline{y}) := sw(\underline{x} - \underline{y}). \tag{4}$$

The *weight* of an element $\underline{x}_i$ in $(GF(q))^m$ is defined to be the number of nonzero digits in $\underline{x}_i$. The *digit weight* $dw(\underline{x})$ of a vector $\underline{x} \in (GF(q))^{Nm}$ is defined to be the sum of the weights of its symbols, or equivalently, the number of nonzero digits in $\underline{x}$:

$$dw(\underline{x}) := |\{(i,j) : x_{ij} \neq 0,\ i = 0, 1, \ldots, N-1;\ j = 0, 1, \ldots, m-1\}|. \tag{5}$$

The *digit distance* $dd(\underline{x}, \underline{y})$ between two elements $\underline{x}$ and $\underline{y}$ from $(GF(q))^{Nm}$ is defined to be the digit weight of their differences $\underline{x} - \underline{y}$:

$$dd(\underline{x}, \underline{y}) := dw(\underline{x} - \underline{y}). \tag{6}$$

The *minimum symbol weight* $sw(C)$ and the *minimum digit weight* $dw(C)$ of the code C are the respective minima of the weights defined above over all nonzero codewords in C. The *minimum symbol distance* $sd(C)$ and the *minimum digit distance* $dd(C)$ of the code C are the respective minima of the distances defined above over all pairs of mutually different codewords in C.

For a vector $\underline{x} = (\underline{x_0}, \ldots, \underline{x_{N-1}})$ let $dw(\underline{x}|j)$ denote the digit weight of the vector of symbol length $N - j$ that is obtained by deleting the j heaviest (i.e. those having the largest weights) symbols from $\underline{x}$. The *weight profile* of a vector $\underline{x} = (\underline{x_0}, \ldots, \underline{x_{N-1}})$ is given by the N-vector

$$\underline{wp}(\underline{x}) := (dw(\underline{x}|0), dw(\underline{x}|1), \ldots, dw(\underline{x}|N-1)). \tag{7}$$

For example the vector (1000,1010,1011) in $((GF(2))^4)^3$ has weight profile $(6,3,1)$. For the code C the *minimum distance profile* $\underline{mdp}(C)$ is defined by

$$\underline{mdp}(C) = (d_0(C), d_1(C), \ldots, d_{sd(C)-1}(C)), \tag{8}$$

where

$$d_j(C) := min\{dw(\underline{c}|j) \ : \ \underline{c} \in C, \ \underline{c} \neq \underline{0}\},$$
$$j = 0, 1, \ldots, sd(C) - 1. \tag{9}$$

In other words, if for fixed j one takes $N - j$ arbitrary symbol components of an arbitrary nonzero codeword in the code C, then the sum of the weights of these components is at least $d_j(C)$. There also exists a codeword in the code C having $N - j$ symbol components whose weights add up to exactly $d_j(C)$. Hence, $d_j(C)$ is the minimum of the minimum digit distances of the $\begin{pmatrix} N \\ j \end{pmatrix}$ codes obtained by puncturing C in j symbol positions. Note that the length of the weight profile is equal to the symbol length N of the code C and that the length of the minimum distance profile is equal to the minimum symbol distance $sd(C)$ of the code C. For example the binary code with parameters $N = 3$, $m = 4$, $k = 2$ and generator matrix

$$G = \begin{bmatrix} 0000 & 1110 & 1111 \\ 1000 & 1000 & 1110 \end{bmatrix}$$

has $\underline{mdp}(C) = (4,2)$.

If the code C is such that the digit dimension k is a multiple of the symbol size m and furthermore if $K := k/m$ fixed symbol positions in any codeword can be seen as information symbols (i.e. $k = Km$ information digits), then the parameter K is called the *symbol code dimension* of C. In this case an SDEC code C of symbol length N, symbol dimension K and symbol size m is denoted by the triplet $\langle N, K, m\rangle$. If we look at the digit length $n = Nm$ and digit dimension k we denote an SDEC code by the triplet $\langle\langle n, k, m\rangle\rangle$.

Now we shall describe the error-control capacity of an SDEC code induced by its minimum distance profile. To do this we need the following definitions. A finite set $\mathcal{A} = \{(a_1, b_1), \ldots, (a_{|\mathcal{A}|}, b_{|\mathcal{A}|})\}$ of pairs of natural numbers is called a *list of pairs* if all first components a_i, $i = 1, ..., |\mathcal{A}|$ of the pairs in $\mathcal{A}$ are mutually different. We say that a vector $\underline{x}$ is *covered* by $\mathcal{A}$, $\mathcal{A}$ being a list of pairs, if there exists a pair (a, b) in $\mathcal{A}$ such that $dw(\underline{x}|a) \leq b$.

Let $\mathcal{T}$ and $\mathcal{U}$ be two lists of pairs. The code C is said to be *$\mathcal{T}$-correcting and $\mathcal{U}$-detecting* if it corrects all errors $\underline{e}$ covered by $\mathcal{T}$ and if it detects all errors $\underline{e}$ which are covered by $\mathcal{U}$ and which are not covered by $\mathcal{T}$. If $\mathcal{U} = \emptyset$, then C is called *$\mathcal{T}$-correcting*, if $\mathcal{T} = \emptyset$ then C is called *$\mathcal{U}$-detecting*. It is immediately clear that C is $\mathcal{T}$-correcting and $\mathcal{U}$-detecting if and only if

1. none of its cosets contains more than one vector covered by $\mathcal{T}$,

2. in case a coset contains a vector covered by $\mathcal{T}$, then it does not contain another vector covered by $\mathcal{U}$.

This is equivalent to saying that none of the differences between two distinct vectors, of which one is covered by $\mathcal{T}$ and the other by $\mathcal{T}$ or $\mathcal{U}$, is a codeword of C. The following theorem is a direct consequence of these remarks.

Theorem 1. The code C is $\mathcal{T}$-correcting and $\mathcal{U}$-detecting if and only if for every pair (s, t) in $\mathcal{T}$ and every pair (u, v) in the union of $\mathcal{T}$ and $\mathcal{U}$ the component $d_{s+u}(C)$ of $\underline{mdp}(C)$ satisfies the inequality

$$d_{s+u}(C) \geq t + v + 1. \tag{10}$$

Proof. Sufficiency: Suppose for all $(s,t) \in \mathcal{T}$ and all $(u,v) \in \mathcal{T} \cup \mathcal{U}$ we have that $d_{s+u}(C) \geq t+v+1$. We have to show that C is $\mathcal{T}$-correcting and $\mathcal{U}$-detecting. So let $\underline{e_1}$ and $\underline{e_2}$ be two different error patterns covered by $\mathcal{T}$ and $\mathcal{T} \cup \mathcal{U}$ respectively. This means that there exist pairs $(s_1,t_1) \in \mathcal{T}$ and $(s_2,t_2) \in \mathcal{T} \cup \mathcal{U}$ such that $dw(\underline{e_1}|s_1) \leq t_1$ and $dw(\underline{e_2}|s_2) \leq t_2$. So, trivially we have

$$dw(\underline{e_1} - \underline{e_2}|s_1 + s_2) \leq t_1 + t_2 < d_{s_1+s_2}(C), \tag{11}$$

and it follows that $\underline{e_1} - \underline{e_2}$ is not a nonzero codeword of C. Hence no difference of two different vectors, of which one is covered by $\mathcal{T}$ and the other is covered by $\mathcal{T}$ or $\mathcal{U}$, is a codeword of C.
Necessity: Suppose the code C is $\mathcal{T}$-correcting and $\mathcal{U}$-detecting. Let (s,t) and (u,v) be pairs in $\mathcal{T}$ and $\mathcal{T} \cup \mathcal{U}$ respectively. Let $\underline{c}$ be a nonzero codeword of C. Assume that $dw(\underline{c}|s+u) \leq t+v$. Then the vector $\underline{c}$ can be written as the difference $\underline{e_1} - \underline{e_2}$ of two vectors $\underline{e_1}$ and $\underline{e_2}$ that satisfy the following two inequalities,

$$dw(\underline{e_1}|s) \leq t \text{ and } dw(\underline{e_2}|u) \leq v. \tag{12}$$

Hence the vector $\underline{e_1}$ is covered by $\mathcal{T}$ and the vector $\underline{e_2}$ is covered by $\mathcal{T} \cup \mathcal{U}$. In other words $\underline{e_1}$ is a correctable error pattern and $\underline{e_2}$ is a correctable or detectable error pattern, while their difference $\underline{e_1} - \underline{e_2}$ is a codeword of C, a contradiction. Hence for all nonzero codewords $\underline{c}$ in C we have that $dw(\underline{c}|s+u) \geq t+v+1$ and consequently we have that $d_{s+u}(C) \geq t+v+1$. This completes the proof.

□

If in any codeword of an $\langle\langle n,k,m\rangle\rangle$ code C e fixed symbol positions ($e \leq sd(C)-1$) are erased then we obtain an $\langle\langle n-em,k,m\rangle\rangle$ code C' with minimum distance profile $\underline{mdp}(C') = (d_0(C'),\ldots,d_{sd(C')-1}(C'))$, where $sd(C') \geq sd(C)-e$ and where $d_j(C')$ satisfies the inequality

$$d_j(C') \geq d_{j+e}(C). \tag{13}$$

For example, the binary $\langle 4,2,4\rangle$ codes constructed by Krol and Vonk [12,14] for the (4,2)-concept fault-tolerant computer have

minimum distance profile $(5,3,1)$ and so they are $\{(1,0),(0,2)\}$-correcting. The punctured $\langle 3,2,4\rangle$ codes, obtained by puncturing one fixed symbol position in all codewords, have minimum distance profile $(3,1)$ and hence they are $\{(0,1)\}$-correcting. Therefore a combination of a symbol erasure and a single bit error in a codeword of the $\langle 4,2,4\rangle$ code can be corrected. In van Gils [6], binary $\langle 3,1,4\rangle$, $\langle 3,1,8\rangle$ and $\langle 3,1,16\rangle$ codes were constructed with minimum distance profiles $(6,3,1)$, $(8,5,1)$ and $(12,7,1)$ respectively, to be used in a generalized Triple Modular Redundant computer. In van Gils [7], binary $\langle\langle 27,16,9\rangle\rangle$ codes were constructed with minimum distance profile $(6,2)$ to be used as single-bit error-correcting, single-byte error-detecting, quadruple-bit error-detecting codes in memory systems composed of banks of three 9-bit wide units. In van Gils and Boly [8] the construction, properties and decoding of $\langle 4,2,8\rangle$ codes with minimum distance profile $(7,4,1)$ are discussed.

An SDEC code with a certain minimum distance profile can be used in many ways. For example in Section III we will construct a binary $\langle 5,2,8\rangle$ SDEC code with minimum distance profile $(10,7,4,1)$. The five ways (cf. Theorem 1) in which this code can be used are given in Table I.

	T-correcting $T=$	U-detecting $U=$
a.	$\{(1,1),(0,4)\}$	$\{(1,2),(0,5)\}$
b.	$\{(1,0),(0,3)\}$	$\{(2,0),(1,3),(0,6)\}$
c.	$\{(0,2)\}$	$\{(2,1),(1,4),(0,7)\}$
d.	$\{(0,1)\}$	$\{(2,2),(1,5),(0,8)\}$
e.	$\emptyset$	$\{(3,0),(2,3),(1,6),(0,9)\}$

Table I: *Error correction/detection properties of an SDEC code with minimum distance profile* (10,7,4,1).

The following bound turns out to be very useful for finding the minimum distance profile of a code.

Theorem 2. For an SDEC code C of symbol length N we have

$$dw(\underline{c}|j) \geq \frac{N-j}{N-j-1} dw(\underline{c}|j+1), \underline{c} \in C, \quad j = 0, 1, \ldots, N-2, \quad (14)$$

and

$$d_j(C) \geq \frac{N-j}{N-j-1} d_{j+1}(C), \quad j = 0, 1, \ldots, sd(C) - 2. \quad (15)$$

Proof. By definition, the average digit weight per symbol, $\frac{1}{N-j} dw(\underline{c}|j)$, is a nonincreasing function of j.

□

Also the tables of Helgert and Stinaff [10] and the updated version given by Verhoeff [30] provide upper bounds on the components of the minimum distance profile.

An $\langle\langle n, k, m\rangle\rangle$ SDEC code C is called *i-optimal*, $0 \leq i \leq sd(C) - 1$, if no $\langle\langle n, k, m\rangle\rangle$ code C' exists such that

$$sd(C') \geq sd(C), \quad d_i(C') > d_i(C) \text{ and}$$

$$d_j(C') \geq d_j(C) \text{ for all } j = 0, 1, \ldots, sd(C) - 1. \quad (16)$$

An SDEC code is called *optimal* whenever it is i-optimal for all $i = 0, 1, \ldots, sd(C) - 1$.

III. SDEC Codes

In general we construct a linear $\langle N, K, m\rangle$ SDEC code over $GF(q)$ by choosing a linear $[N, K]$ code over $GF(q^m)$ and mapping this code onto its q-ary image with respect to a certain basis of $GF(q^m)$ over $GF(q)$. To be more specific, let q be a prime power and let the ordered set $B = \langle \beta_0, \beta_1, \ldots, \beta_{m-1}\rangle$ be a basis of $GF(q^m)$ over $GF(q)$. For the vector $\underline{x} = (x_0, x_1, \ldots, x_{N-1}) \in (GF(q^m))^N$ the *q-ary image* of $\underline{x}$ with respect to the basis B is defined to be the vector $\underline{x(B)} \in (GF(q))^{mN}$ where

$$\underline{x(B)} = (\underline{x_0(B)}, \ldots, \underline{x_{N-1}(B)}) \text{ and } \underline{x_i(B)} = (x_{i0}, x_{i1}, \ldots, x_{i(m-1)}),$$

$$x_i = \sum_{j=0}^{m-1} x_{ij}\beta_j, \quad i = 0, 1, \ldots, N-1. \tag{17}$$

For a linear $[N, K]$ code C over $GF(q^m)$ we define the code $C(B)$ as

$$C(B) := \{\underline{c(B)} \ : \ \underline{c} \in C\}. \tag{18}$$

The code $C(B)$ is called the *q-ary image* of C with respect to the basis B. Clearly, $C(B)$ is a linear code of length Nm and of dimension Km over $GF(q)$.

If the K by N matrix G over $GF(q^m)$ with rows

$$\underline{g_j} = (g_{j0}, g_{j1}, \ldots, g_{j(N-1)}), \ 0 \le j \le K-1$$

is a generator matrix for the code C, then the Km by Nm matrix $G(B)$ over $GF(q)$ whose rows are equal to

$$(\underline{(\beta_i g_{j0})(B)}, \underline{(\beta_i g_{j1})(B)}, \ldots, \underline{(\beta_i g_{j(N-1)})(B)}),$$

$$i = 0, 1, \ldots, m-1, \ j = 0, 1, \ldots, K-1,$$

is a generator matrix of $C(B)$. Analogously, if the $(N-K)$ by N matrix H over $GF(q^m)$ with jth column

$$\underline{h_j} = (h_{0j}, h_{1j}, \ldots, h_{(N-K-1)j}), \ j = 0, 1, \ldots, N-1$$

is a parity check matrix of C, then the $(N-K)m$ by Nm matrix $H(B)$ over $GF(q)$ whose $(i+jm)$th column is equal to

$$(\underline{(\beta_i h_{0j})(B)}, \underline{(\beta_i h_{1j})(B)}, \ldots, \underline{(\beta_i h_{(N-K-1)j})(B)})^T,$$

$$i = 0, 1, \ldots, m-1, \ j = 0, 1, \ldots, N-1,$$

is a parity check matrix for $C(B)$.

The minimum distance profile of the SDEC code $C(B)$ depends on the choice of the code C over $GF(q^m)$ and on the choice of the basis B of $GF(q^m)$ over $GF(q)$. A way to construct good SDEC codes is to take an $[N, K]$ code C over $GF(q^m)$ of maximal minimum symbol distance (for $N \le q^m + 1$ this is an MDS code) and to determine a basis B of $GF(q^m)$ over $GF(q)$ such that $C(B)$ has a good minimum distance profile. However, a Galois field has a large number of bases and different bases may yield different

minimum distance profiles. In the next subsection we look at the equivalence of SDEC codes which will have to be defined in a special way because of the special symbol and digit structure of the codes. We provide some theorems that state which bases of $GF(q^m)$ over $GF(q)$ yield q-ary images that are equivalent SDEC codes. In subsections B-G we consider constructions of SDEC codes.

A. Equivalence of SDEC Codes

For any integer t let S_t denote the group of all the permutations of the set $\{0, 1, \ldots, t-1\}$.

Definition 1. For permutations $\pi \in S_N$ and $\sigma_0, \sigma_1, \ldots, \sigma_{N-1} \in S_m$, we define the permutation $(\pi; \sigma_0, \sigma_1, \ldots, \sigma_{N-1})$ of $\{0, 1, \ldots, N-1\} \times \{0, 1, \ldots, m-1\}$ by

$$(\pi; \sigma_0, \sigma_1, \ldots, \sigma_{N-1})(i, j) := (\pi(i), \sigma_i(j)),$$

$$i = 0, 1, \ldots, N-1, \quad j = 0, 1, , \ldots, m-1. \tag{19}$$

Obviously the set of all such permutations constitutes a group. We denote this group by $S_N[S_m]$, i.e.

$$S_N[S_m] := \{(\pi; \sigma_0, \sigma_1, \ldots, \sigma_{N-1}) \mid \pi \in S_N;\ \sigma_0, \ldots, \sigma_{N-1} \in S_m\}. \tag{20}$$

Applying a permutation $(\pi; \sigma_0, \sigma_1, \ldots, \sigma_{N-1}) \in S_N[S_m]$ on the coordinates of a vector

$$(\underline{x_0}, \ldots, \underline{x_{N-1}}) = (x_{00}, \ldots, x_{0(m-1)}, \ldots, x_{(N-1)0}, \ldots, x_{(N-1)(m-1)})$$

yields the vector

$$(\underline{\tilde{x}_0}, \ldots, \underline{\tilde{x}_{N-1}}) = (\tilde{x}_{00}, \ldots, \tilde{x}_{0(m-1)}, \ldots, \tilde{x}_{(N-1)0}, \ldots, \tilde{x}_{(N-1)(m-1)}),$$

where $\tilde{x}_{\pi(i)\sigma_i(j)} = x_{ij}$, $i = 0, 1, \ldots, N-1$, $j = 0, 1, \ldots, m-1$.

Definition 2. Two $\langle\langle n, k, m \rangle\rangle$ SDEC codes C_1 and C_2 over the same alphabet are said to be *equivalent* if there exists a permutation $(\pi; \sigma_0, \sigma_1, \ldots, \sigma_{N-1}) \in S_N[S_m]$ that maps the codewords of

C_1 onto the codewords of C_2.

Definition 3. The group of permutations in $S_N[S_m]$ that map an SDEC code onto itself is called the *automorphism group* of that code.

It is clear that equivalent SDEC codes have identical minimum distance profiles.

In the following we give some theorems that say, given a linear code over $GF(q^m)$, which bases give equivalent q-ary SDEC codes. The next two theorems can be shown straightforwardly.

Theorem 3. For a linear code C over $GF(q^m)$, a basis B of $GF(q^m)$ over $GF(q)$ and a nonzero element ξ of $GF(q^m)$, the SDEC codes $C(B)$ and $C(\xi B)$ are the same.

Theorem 4. For a basis $B = \langle \beta_0, \ldots, \beta_{m-1} \rangle$ of $GF(q^m)$ over $GF(q)$ and an integer t, the (ordered) set $B^{q^t} := \langle \beta_0^{q^t}, \ldots, \beta_{m-1}^{q^t} \rangle$ is also a basis of $GF(q^m)$ over $GF(q)$. If C is a linear code over $GF(q^m)$ with a parity check matrix $H = [\underline{h_0}^T, \ldots, \underline{h_{N-1}}^T]$ such that there exists an integer t and a permutation $\pi \in S_N$ with $\underline{h_{\pi(i)}} = \underline{h_i}^{q^t} := (h_{i0}^{q^t}, \ldots, h_{i(N-K-1)}^{q^t})$, then the q-ary SDEC codes $C(B)$ and $C(B^{q^t})$ are equivalent.

For example, cyclic codes over $GF(q^m)$ satisfy the conditions of Theorem 4 for all integers t.

Theorem 5. Let C be a linear code of length N over $GF(q^m)$ with a generator matrix solely consisting of elements of $GF(q^t)$, $GF(q^t)$ being a subfield of $GF(q^m)$. Let $A = \langle \alpha_0, \alpha_1, \ldots, \alpha_{m-1} \rangle$ and $B = \langle \beta_0, \beta_1, \ldots, \beta_{m-1} \rangle$ be bases of $GF(q^m)$ over $GF(q)$, let $\Lambda = \langle \lambda_0, \lambda_1, \ldots, \lambda_{t-1} \rangle$ be a basis of $GF(q^t)$ over $GF(q)$, and let τ and σ be permutations in S_m such that

$$\sigma(\underline{(\alpha_j \lambda_h)(A)}) = \underline{(\beta_{\tau(j)} \lambda_h)(B)}, \tag{21}$$

for $j = 0, 1, \ldots, m-1$, $h = 0, 1, \ldots, t-1$. Then the SDEC code $C(A)$ is mapped onto the SDEC code $C(B)$ by the permutation $(1; \sigma, \ldots, \sigma)$ where 1 denotes the identity.

Proof. Because formula (21) holds for all basis elements of Λ, it

is true for all $\xi \in GF(q^t)$, i.e.

$$\sigma(\underline{(\alpha_j \xi)(A)}) = \underline{(\beta_{\tau(j)} \xi)(B)}, \quad j = 0, 1, \ldots, m-1. \tag{22}$$

Because the code C has a generator matrix G solely consisting of elements of $GF(q^t)$ we have that $(1; \sigma, \ldots, \sigma)$ maps the rows of the generator matrix $G(A)$ of $C(A)$ onto the rows of the generator matrix $G(B)$ of $C(B)$.

□

Let $\nu_a \in S_m$ be defined by

$$\nu_a(i) := (i + a) \bmod m. \tag{23}$$

For $m = 2t$, where t is even, we define the permutation $\omega \in S_m$ by

$$\omega(i) := (i + \delta_i t) \bmod m, \tag{24}$$

where

$$\delta_i := \begin{cases} 0 \text{ if } i \text{ is even} \\ 1 \text{ if } i \text{ is odd.} \end{cases} \tag{25}$$

It is easy to check that for all integers a we have

$$\omega \nu_a = \nu_{a+\delta_a t} \omega. \tag{26}$$

Corollary 6. Let $m = 2t$ where t is even, let $A = \langle \alpha, \alpha^q, \ldots, \alpha^{q^{m-1}} \rangle$ and $B = \langle \beta, \beta^q, \ldots, \beta^{q^{m-1}} \rangle$ be normal bases of $GF(q^m)$ over $GF(q)$ and let $\Lambda = \langle \lambda, \lambda^q, \ldots, \lambda^{q^{t-1}} \rangle$ be a normal basis of $GF(q^t)$ over $GF(q)$. If a and b are integers such that

$$\nu_a \omega(\underline{(\alpha \lambda^{q^i})(A)}) = \underline{(\beta \lambda^{q^{i+b}})(B)}, \quad i = 0, 1, \ldots, t-1, \tag{27}$$

and C is a linear code over $GF(q^m)$ with a generator matrix solely consisting of elements of $GF(q^t)$, then the code $C(A)$ is mapped onto the code $C(B)$ by the permutation $(1; \nu_{a-b}\omega, \ldots, \nu_{a-b}\omega)$.

Proof. It is easy to see that for a normal basis Γ of $GF(q^m)$ over $GF(q)$ we have

$$\nu_j(\underline{\eta(\Gamma)}) = \underline{(\eta^{q^j})(\Gamma)}, \quad j = 0, 1, \ldots, m-1, \quad \eta \in GF(q^m).$$

Hence, for $i = 0, 1, \ldots, t-1$ and $j = 0, 1, \ldots, m-1$ we have

$$\nu_{a-b}\omega(\underline{(\alpha^{q^j}\lambda^{q^i})(A)}) = \nu_{a-b}\omega\nu_j(\underline{(\alpha\lambda^{q^{i-j}})(A)}) =$$

$$\nu_{a-b+j+\delta_{jt}}\omega(\underline{(\alpha\lambda^{q^{i-j}})(A)}) = \nu_{-b+j+\delta_{jt}}(\underline{(\beta\lambda^{q^{i-j+b}})(B)}) =$$

$$\underline{(\beta^{q^{-b+j+\delta_{jt}}}\lambda^{q^{i+\delta_{jt}}})(B)} = \underline{(\beta^{q^{-b+j+\delta_{jt}}}\lambda^{q^i})(B)}.$$

With Theorem 5 the assertion follows.

□

Corollary 7. Let α be a primitive element of $GF(2^8)$ that satisfies $\alpha^8 = \alpha^4 + \alpha^3 + \alpha^2 + 1$ and let $\mathcal{N}(i)$ denote the normal basis of $GF(2^8)$ over $GF(2)$ containing α^i. For a linear code C over $GF(2^8)$ with a generator matrix solely consisting of elements of $GF(2^4)$, the SDEC codes $C(\mathcal{N}(i))$ and $C(\mathcal{N}(j))$ are pairwise equivalent for

$$(i,j) \in \{(5,87),(43,95),(11,47),(39,55),$$

$$(53,63),(9,91),(15,29),(21,61)\}.$$

Proof. Define ξ to be α^{17}. Then ξ is in $GF(2^4)$ and $\langle \xi^3, \xi^6, \xi^{12}, \xi^9 \rangle$ is a normal basis of $GF(2^4)$ over $GF(2)$. Table II gives the binary images of the elements $\alpha^i\xi^{3\cdot 2^j}$ with respect to the normal bases $\mathcal{N}(i)$ for $j = 0, 1, 2, 3$ and $i = 5, 87, 43, 95, 11, 47, 39, 55$. By straightforward checking we can see that

$$\omega(\underline{(\alpha^5\xi^{3\cdot 2^j})(\mathcal{N}(5))}) = \underline{(\alpha^{87}\xi^{3\cdot 2^{j+1}})(\mathcal{N}(87))}.$$

$$\omega(\underline{(\alpha^{43}\xi^{3\cdot 2^j})(\mathcal{N}(43))}) = \underline{(\alpha^{95}\xi^{3\cdot 2^{j+2}})(\mathcal{N}(95))}.$$

$$\omega(\underline{(\alpha^{11}\xi^{3\cdot 2^j})(\mathcal{N}(11))}) = \underline{(\alpha^{47}\xi^{3\cdot 2^{j+3}})(\mathcal{N}(47))}.$$

$$\omega(\underline{(\alpha^{39}\xi^{3\cdot 2^j})(\mathcal{N}(39))}) = \underline{(\alpha^{55}\xi^{3\cdot 2^{j+2}})(\mathcal{N}(55))}.$$

Applying Corollary 6 gives the assertion for the pairs

$$(i,j) \in S_1 := \{(5,87),(43,95),(11,47),(39,55)\}.$$

Now, consider the dual code $C^\perp$ of C. Clearly, $C^\perp$ also has a generator matrix solely consisting of elements of $GF(2^4)$. So, the

i	$j=0$	$j=1$	$j=2$	$j=3$
5	0101 1010	0101 0111	0001 0101	1001 1000
87	1000 1001	0000 1111	0101 0111	0101 0001
43	1111 1101	1001 1010	1101 0001	0011 0110
95	1001 0101	0110 0011	1111 1101	1000 1011
11	1110 0010	1101 1000	1101 1111	0110 0101
47	1000 1101	1101 1111	0111 0100	1010 0110
39	0010 1110	1111 1010	0100 0110	0001 0010
55	0100 0110	0000 0011	0110 1010	1010 1111

Table II: *Binary images.*

codes $C^{\perp}(\mathcal{N}(i))$ and $C^{\perp}(\mathcal{N}(j))$ are equivalent for the pairs $(i,j) \in S_1$ as defined above. Furthermore, for a linear code C over $GF(q^m)$ and complementary bases A and B of $GF(q^m)$ over $GF(q)$ we have that $(C(A))^{\perp} = C^{\perp}(B)$. Together with the fact that $\mathcal{N}(i)$, $i = 63, 95, 15, 55, 47, 61, 91, 87$ are the complementary bases of $\mathcal{N}(j)$, $j = 5, 9, 11, 21, 29, 39, 43, 53$ respectively, the remaining part of the theorem follows.

□

This Corollary can be used to show that several of the $[4,2]$ codes over $GF(2^8)$ constructed in [8] are equivalent (for more details see [1]).

If n is a divisor of m, let $T_{m,n} : GF(q^m) \to GF(q^n)$ denote the *trace* of $GF(q^m)$ into $GF(q^n)$ defined by

$$T_{m,n}(\eta) := \sum_{i=0}^{\frac{m}{n}-1} \eta^{q^{in}}, \quad \eta \in GF(q^m). \tag{28}$$

Corollary 8. Let m and q be even integers and let $A = \langle \alpha, \alpha^q, \ldots, \alpha^{q^{m-1}} \rangle$ and $B = \langle \beta, \beta^q, \ldots, \beta^{q^{m-1}} \rangle$ be complementary normal bases of $GF(q^m)$ over $GF(q)$. If C is a linear code over $GF(q^m)$ with a generator matrix solely containing elements of $GF(q^2)$, then the SDEC codes $C(A)$ and $C(B)$ are equivalent.

Proof. The Galois field $GF(q^2)$ has a primitive element θ such

that $\langle \theta, \theta^q \rangle$ is a normal basis of $GF(q^2)$ over $GF(q)$ [3]. The sum $\theta + \theta^q$ is a nonzero element of $GF(q)$. Therefore we may take θ such that $\theta + \theta^q = 1$. Then, because A and B are complementary bases, we have

$$T_{m,1}(\alpha^{q^i}\beta^{q^j}\theta) + T_{m,1}(\alpha^{q^i}\beta^{q^j}\theta^q) = T_{m,1}(\alpha^{q^i}\beta^{q^j}) = \delta_{ij},$$

$$i, j = 0, 1, , \ldots, m-1. \tag{29}$$

Furthermore

$$T_{m,1}(\alpha^{q^i}\beta^{q^j}\theta^{q^h}) = (T_{m,1}(\alpha^{q^i}\beta^{q^j}\theta^{q^h}))^{q^{m-j-i}} =$$

$$T_{m,1}(\alpha^{q^{m-j}}\beta^{q^{m-i}}\theta^{q^{h+m-j-i}}),\ i, j = 0, 1, \ldots, m-1, h = 0, 1. \tag{30}$$

Combining (29) and (30) gives the equation

$$T_{m,1}(\alpha^{q^i}\beta^{q^j}\theta^{q^h}) = T_{m,1}(\alpha^{q^{m-j}}\beta^{q^{m-i}}\theta^{q^h}),$$

$$i, j = 0, 1, \ldots, m-1,\ h = 0, 1. \tag{31}$$

Now let the permutation $\rho \in S_m$ be defined by

$$\rho(i) := (m - i)\ mod\ m,\ \ i = 0, 1, \ldots, m-1. \tag{32}$$

Recall that for complementary bases $\Lambda = \langle \lambda_0, \lambda_1, \ldots, \lambda_{m-1} \rangle$ and $M = \langle \mu_0, \mu_1, \ldots, \mu_{m-1} \rangle$ of $GF(q^m)$ over $GF(q)$, we have

$$\underline{\xi(\Lambda)} = (T_{m,1}(\xi\mu_0), T_{m,1}(\xi\mu_1), \ldots, T_{m,1}(\xi\mu_{m-1})),\ \ \xi \in GF(q^m) \tag{33}$$

(see [16, p. 118]). Combining the above facts we see that for all $i = 0, 1, \ldots, m-1$ and $h = 0, 1$

$$\rho(\underline{(\alpha^{q^i}\theta^{q^h})(A)}) = \rho((T_{m,1}(\alpha^{q^i}\theta^{q^h}\beta), \ldots, T_{m,1}(\alpha^{q^i}\theta^{q^h}\beta^{q^{m-1}}))) =$$

$$\rho((T_{m,1}(\alpha\theta^{q^h}\beta^{q^{m-i}}), T_{m,1}(\alpha^{q^{m-1}}\theta^{q^h}\beta^{q^{m-i}}), \ldots, T_{m,1}(\alpha^q\theta^{q^h}\beta^{q^{m-i}}))) =$$

$$= (T_{m,1}(\alpha\theta^{q^h}\beta^{q^{m-i}}), (T_{m,1}(\alpha^q\theta^{q^h}\beta^{q^{m-i}}), \ldots, T_{m,1}(\alpha^{q^{m-1}}\theta^{q^h}\beta^{q^{m-i}})) =$$

$$\underline{(\beta^{q^{m-i}}\theta^{q^h})(B)}.$$

Now by Theorem 5 we see that the code $C(A)$ is mapped onto the code $C(B)$ by the permutation $(1; \rho, \ldots, \rho)$.

□

Corollary 9. For a linear code C over $GF(2^8)$ that has a generator matrix solely consisting of elements of $GF(2^2)$, the codes

$$C(\mathcal{N}(i)), \quad i \in S_1 := \{9, 21, 39, 43, 55, 61, 91, 95\}$$

are mutually equivalent, and the codes

$$C(\mathcal{N}(i)), \quad i \in S_2 := \{5, 11, 15, 29, 47, 53, 63, 87\}$$

are mutually equivalent.

Proof. Let α be a primitive element of $GF(2^8)$ satisfying $\alpha^8 + \alpha^4 + \alpha^3 + \alpha^2 + 1 = 0$. Then the bases $\mathcal{N}(i)$ and $\mathcal{N}(j)$ are complementary for $(i,j) = (5,63), (9,95), (11,15), (21,55), (29,47), (39,61), (43,91), (53,87)$. Let the sets $\mathcal{A}_1, \ldots, \mathcal{A}_4$ be defined by

$$\mathcal{A}_1 := \{5, 63, 53, 87\}, \quad \mathcal{A}_2 := \{11, 15, 29, 47\},$$

$$\mathcal{A}_3 := \{43, 91, 9, 95\}, \quad \mathcal{A}_4 := \{39, 61, 21, 55\}.$$

By combining the above remark and Corollaries 7 and 8, we see that the codes $C(\mathcal{N}(i))$, $i \in \mathcal{A}_h$, are equivalent, for fixed $h = 1, 2, 3, 4$. But we can say more. For $\eta := \alpha^{85}$, $\langle \eta, \eta^2 \rangle$ is the normal basis of $GF(2^2)$ over $GF(2)$. In Table III we give the binary images of the elements $\eta^{2^j}\alpha^i$ with respect to the normal bases $\mathcal{N}(i)$, $i = 5, 29, 9, 61$ and $j = 0, 1$. From this table and Theorem 5

i	$j = 0$	$j = 1$
5	1100 1111	0100 1111
29	0100 1111	1100 1111
9	0000 1011	1000 1011
61	1000 1011	0000 1011

Table III: *Binary images*

we see that the codes $C(\mathcal{N}(5))$ and $C(\mathcal{N}(29))$ are equivalent and that the codes $C(\mathcal{N}(9))$ and $C(\mathcal{N}(61))$ are also equivalent. This completes the proof.

□

For example, Corollary 9 explains the fact that the [3,1] codes over $GF(2^8)$ constructed in [6] have an identical minimum distance profile with respect to the bases $\mathcal{N}(i)$, $i \in S_1$, namely (8,4,1) and an identical minimum distance profile with respect to the bases $\mathcal{N}(i)$, $i \in S_2$, namely (8,5,1).

B. Construction of a class of SDEC codes

In this and following subsections we will give a number of constructions of SDEC codes. The parameters of these codes are summarized in the Tables IV-IX.

We start by considering linear codes C over $GF(q^m)$ that have a parity check matrix of the form

$$H = \begin{bmatrix} \gamma_0 & \gamma_1 & \cdots & \cdots & \gamma_{N-1} \\ \gamma_0^q & \gamma_1^q & \cdots & \cdots & \gamma_{N-1}^q \\ \gamma_0^{q^2} & \gamma_1^{q^2} & \cdots & \cdots & \gamma_{N-1}^{q^2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \gamma_0^{q^{t-1}} & \gamma_1^{q^{t-1}} & \cdots & \cdots & \gamma_{N-1}^{q^{t-1}} \end{bmatrix}, \tag{34}$$

or

$$\tilde{H} = \begin{bmatrix} 1 & 1 & \cdots & \cdots & 1 \\ & & H & & \end{bmatrix}, \tag{35}$$

where $\gamma_0, \gamma_1, \ldots, \gamma_{N-1}$ are elements of $GF(q^m)$ and $1 \leq t \leq m$.

Lemma 10. For integers v and t, $v \leq t$, and elements $\alpha_0, \alpha_1, \ldots, \alpha_{v-1} \in GF(q^m)$, the vectors

$$\begin{pmatrix} \alpha_0 \\ \alpha_0^q \\ \vdots \\ \alpha_0^{q^{t-1}} \end{pmatrix}, \ldots\ldots, \begin{pmatrix} \alpha_{v-1} \\ \alpha_{v-1}^q \\ \vdots \\ \alpha_{v-1}^{q^{t-1}} \end{pmatrix} \tag{36}$$

are linearly dependent over $GF(q^m)$ if and only if $\alpha_0, \ldots, \alpha_{v-1}$ are linearly dependent over $GF(q)$.

Proof. The "if" part of the lemma is obvious. The "only if" part will be shown by induction on t. The case $t = 1$ is trivial. Now let $t \geq 2$ and suppose that $\alpha_0, \ldots, \alpha_{v-1}$, $v \leq t$ are such that

$$\begin{pmatrix} \alpha_0 \\ \alpha_0^q \\ \vdots \\ \alpha_0^{q^{t-1}} \end{pmatrix}, \ldots\ldots, \begin{pmatrix} \alpha_{v-1} \\ \alpha_{v-1}^q \\ \vdots \\ \alpha_{v-1}^{q^{t-1}} \end{pmatrix}$$

are linearly dependent over $GF(q^m)$. So, there exist $x_0, \ldots, x_{v-1} \in GF(q^m)$ such that

$$\sum_{i=0}^{v-1} x_i \alpha_i^{q^r} = 0, \quad r = 0, 1, \ldots, t-1. \tag{37}$$

If $v < t$ or if $v = t$ and one of the x_i equals zero, then we may apply the induction hypothesis and we are through. So, suppose that $v = t$ and that all $x_0, \ldots, x_{t-1}$ are nonzero. Substitute $y_i := x_i/x_{t-1}$ and $\beta_i := \alpha_i/\alpha_{t-1}$, $i = 0, 1, \ldots, t-2$ in formula (37) (we may assume that all α_i are nonzero, otherwise the lemma is trivial). Then we obtain

$$\sum_{i=0}^{t-2} y_i \beta_i^{q^r} = -1, \quad r = 0, 1, \ldots, t-1. \tag{38}$$

By subtracting the $(r+1)$-th equation from the r-th equation we get

$$\sum_{i=0}^{t-2} y_i \gamma_i^{q^r} = 0, \quad r = 0, 1, \ldots, t-2, \tag{39}$$

where $\gamma_i := \beta_i - \beta_i^q$, $i = 0, 1, \ldots, t-2$. This means that the vectors

$$\begin{pmatrix} \gamma_0 \\ \gamma_0^q \\ \vdots \\ \gamma_0^{q^{t-2}} \end{pmatrix}, \ldots\ldots, \begin{pmatrix} \gamma_{t-2} \\ \gamma_{t-2}^q \\ \vdots \\ \gamma_{t-2}^{q^{t-2}} \end{pmatrix}$$

are linearly dependent over $GF(q^m)$. By the induction hypothesis we may conclude that $\gamma_0, \gamma_1, \ldots, \gamma_{t-2}$ are linearly dependent over

$GF(q)$. So, there exist elements $z_0, z_1, \ldots, z_{t-2} \in GF(q)$, not all zero, such that

$$0 = \sum_{i=0}^{t-2} z_i \gamma_i = \sum_{i=0}^{t-2} z_i \beta_i - (\sum_{i=0}^{t-2} z_i \beta_i)^q. \tag{40}$$

Hence, $z_{t-1} := -\sum_{i=0}^{t-2} z_i \beta_i$ is an element of $GF(q)$ and thus

$$\sum_{i=0}^{t-1} z_i \alpha_i = 0 \text{ with } z_0, z_1, \ldots, z_{t-1} \in GF(q).$$

This proves the assertion.

□

For a linear code C over $GF(q^m)$, we denote by $C(q)$ the *q-ary subfield subcode* of C, i.e.

$$C(q) := \{\underline{c} = (c_0, \ldots, c_{N-1}) \in C \ : \ c_0, \ldots, c_{N-1} \in GF(q)\}. \tag{41}$$

By $d(q)$ we denote the minimum distance of $C(q)$, which is defined to be infinite if $C(q)$ consists of the zero codeword only.

Theorem 11. The symbol distance of a linear code C over $GF(q^m)$ with a parity check matrix as in (34) equals $min\{t+1, d(q)\}$.

Proof. If $d(q) < t+1$, then by Lemma 10 it follows that any $d(q)-1$ columns of H are linearly independent over $GF(q^m)$, and hence the minimum symbol distance of C equals $d(q)$.

If $d(q) \geq t+1$, then by Lemma 10 any t columns of H are linearly independent over $GF(q^m)$, and hence by the Singleton bound the minimum symbol distance of C equals $t+1$.

□

Example 1. Let α be an N-th root of unity and let C be the cyclic code of length N over $GF(q^m)$ with roots $\alpha, \alpha^q, \alpha^{q^2}, \ldots, \alpha^{q^{m-1}}$. Clearly, the q-ary subfield subcode $C(q)$ of C has minimum distance at most $m+1$. With Theorem 11 it follows that the minimum symbol distance of C equals the minimum distance of $C(q)$ (see also [15]).

We now give a theorem that says something about the minimum digit distance of an SDEC code $C(B)$ depending on the linear independence relations of the basis elements of B with respect to a subfield $GF(q^a)$ of $GF(q^m)$.

Theorem 12. Let C be a linear $[N, K]$ code over $GF(q^m)$ with a parity check matrix solely containing elements of $GF(q^a)$ and let $B = \langle \beta_0, \beta_1, \ldots, \beta_{m-1} \rangle$ be a basis of $GF(q^m)$ over $GF(q)$. If the minimum digit distance d of $C(B)$ is less than $s + r - 1$, then $d = d(q)$, where s is the symbol distance of C and where r is such that any r distinct elements of B are linearly independent over $GF(q^a)$.

Proof. Suppose that $d < s + r - 1$. Let $\underline{c} \in C(B)$ be a codeword of digit weight d. So, there are d not necessarily distinct indices $p_0, \ldots, p_{d-1} \in \{0, 1, \ldots, m-1\}$ and indices $j_0, \ldots, j_{d-1} \in \{0, 1, \ldots, N-1\}$ such that

$$\sum_{h=0}^{d-1} c_{j_h p_h} \beta_{p_h} \gamma_{i j_h} = 0, \quad i = 0, 1, \ldots, N-K-1, \tag{42}$$

where $\gamma_{ij} \in GF(q^a)$, $i = 0, 1, \ldots, N-K-1$, $j = 0, 1, \ldots, N-1$ are the elements of a parity check matrix for C. Because the symbol distance of C is s, there are at least s distinct elements in $\{j_0, \ldots, j_{d-1}\}$, say $j_0, \ldots, j_{s-1}$. The equations (42) yield that

$$\sum_{h=0}^{s-2} c_{j_h p_h} \beta_{p_h} \gamma_{i j_h} = - \sum_{h=s-1}^{d-1} c_{j_h p_h} \beta_{p_h} \gamma_{i j_h}, \quad i = 0, 1, \ldots, N-K-1. \tag{43}$$

The matrix consisting of the elements $\gamma_{i j_h}$, $i = 0, 1, \ldots, N-K-1$, $h = 0, 1, \ldots, s-2$ has rank $s-1$, and so (43) implies that

$$\beta_{p_i} = \sum_{j=s-1}^{d-1} \xi_{ij} \beta_{p_j}, \quad i = 0, 1, \ldots, s-2, \tag{44}$$

with $\xi_{ij} \in GF(q^a)$, $i = 0, 1, \ldots, s-2$, $j = s-1, \ldots, d-1$. Because $d < s + r - 1$ and because any r elements of B are linearly independent over $GF(q^a)$, equations (44) yield that for all $i \in \{0, 1, \ldots, s-2\}$ there exists a $u_i \in \{s-1, \ldots, d-1\}$ such that $\beta_{p_i} = \beta_{p_{u_i}}$. Consequently, at most $d - s + 1$ elements of

$\{\beta_{p_0}, \beta_{p_1}, \ldots, \beta_{p_{d-1}}\}$ are mutually different. Now, define

$$I := \{i \ : \ c_{ij} \neq 0 \text{ for some } j \in \{0, 1, \ldots, m-1\}\}$$

and

$$J := \{j \ : \ c_{ij} \neq 0 \text{ for some } i \in \{0, 1, \ldots, N-1\}\}.$$

Above we have shown that $|J| \leq d - s + 1$. Because $\underline{c} \in C(B)$, we have

$$\sum_{j \in J} (\sum_{i \in I} c_{ij} \gamma_{hi}) \beta_j = 0, \quad h = 0, 1, \ldots, N - K - 1. \tag{45}$$

Since $|J| \leq d - s + 1 \leq r$, it follows from (45) that

$$(c_{0j}, c_{1j}, \ldots, c_{(N-1)j}) \in C(q) \text{ for all } j \in J.$$

The weight of such a word is at most d and at least $d(q)$, hence $d(q) \leq d$. Trivially we have that $d(q) \geq d$, so we may conclude that $d = d(q)$.

□

Lemma 13. For a normal or polynomial basis B of $GF(2^8)$ over $GF(2)$, any two distinct elements of B are linearly independent over $GF(2^4)$.

Proof. Let β_1, β_2 be two elements of B such that

$$x_1\beta_1 + x_2\beta_2 = 0 \text{ for some nonzero } x_1, x_2 \in GF(2^4).$$

Then, $\beta_1(\beta_2)^{-1} \in GF(2^4)$ and there are i, j, $0 \leq j \leq i \leq 7$ such that $\beta_1 = \alpha^{2^i}$, $\beta_2 = \alpha^{2^j}$ and $2^i - 2^j = 0 \bmod 17$ if B is a normal basis, or $\beta_1 = \alpha^i$, $\beta_2 = \alpha^j$ and $i - j = 0 \bmod 17$ if B is a polynomial basis. This is only possible if $i = j$, hence $\beta_1 = \beta_2$, and the lemma follows.

□

Example 2. Let γ be a primitive element of $GF(2^4)$ and let C be the [16,14] code over $GF(2^8)$ with parity check matrix

$$H = \begin{bmatrix} 1 & 1 & 1 & \ldots & 1 \\ 0 & 1 & \gamma & \ldots & \gamma^{14} \end{bmatrix}.$$

This code has symbol distance 3 and the binary subfield subcode has minimum distance 4. By Theorem 12 and Lemma 13 the binary image of C with respect to a normal or polynomial basis of $GF(2^8)$ over $GF(2)$ has minimum distance profile (4,3,1). These codes are $\{(1,0)\}$-correcting and $\{(0,2)\}$-detecting.

Theorem 14. For an $\langle\langle Nm, k, m\rangle\rangle$ SDEC code C over $GF(q)$ such that for any codeword in C the sum of its symbols equals zero, the minimum distance profile $\underline{mdp}(C)$ of C satisfies the inequality

$$d_1(C) \geq \frac{d_0(C)}{2}.$$

Proof. Puncture C in the first symbol and denote the resulting code by $\hat{C}$. If

$$\underline{\hat{c}} = (c_{10}, \ldots, c_{1(m-1)}, \ldots, c_{(N-1)0}, \ldots, c_{(N-1)(m-1)})$$

is a codeword of $\hat{C}$, then

$$\underline{c} := (\sum_{i=1}^{N-1} c_{i0}, \ldots, \sum_{i=1}^{N-1} c_{i(m-1)}; \underline{\hat{c}})$$

is a codeword of C. Clearly the digit weight of $\underline{c}$ is at most twice the digit weight of $\underline{\hat{c}}$. We get analogous results if we puncture C in any other symbol. It follows that $d_0(C) \leq 2d_1(C)$.

□

Consequently, an SDEC code C satisfying the condition of Theorem 14 and having symbol distance 3 and minimum digit distance d_0 is $\{(1,0), (0, \lfloor \frac{d_0-1}{2} \rfloor)\}$ -correcting, i.e. it can correct single symbol errors and up to $\lfloor \frac{d_0-1}{2} \rfloor$ digit errors. Note that the q-ary images of the linear codes over $GF(q^m)$ that have a parity check matrix containing a row of ones satisfy the condition of the theorem.

Similarly to Theorem 11 we have the following result.

Theorem 15. The symbol distance of a linear code C over $GF(q^m)$ with a parity check matrix $\tilde{H}$ as in (35) equals $min\{t+2, d(q)\}$.

Proof.

Case 1: $d(q) \geq t+2$. Assume that $t+1$ columns of $\tilde{H}$ are linearly dependent over $GF(q^m)$, say

$$\begin{pmatrix} 1 \\ \gamma_0 \\ \gamma_0^q \\ \vdots \\ \gamma_0^{q^{t-1}} \end{pmatrix}, \ldots\ldots, \begin{pmatrix} 1 \\ \gamma_t \\ \gamma_t^q \\ \vdots \\ \gamma_t^{q^{t-1}} \end{pmatrix}.$$

So, there exist $x_0, \ldots, x_t \in GF(q^m)$ such that

$$\sum_{i=0}^{t} x_i \gamma_i^{q^{r-1}} = 0, \quad r = 1, \ldots, t, \tag{46}$$

$$\sum_{i=0}^{t} x_i = 0. \tag{47}$$

Without loss of generality we may assume that $\gamma_t \neq 0$. Then from (46) and (47) it follows that

$$\sum_{i=0}^{t-1} x_i \left(\frac{\gamma_i}{\gamma_t} - 1\right)^{q^{r-1}} = 0, \quad r = 1, \ldots, t. \tag{48}$$

Now, by Lemma 10 there exist $z_0, \ldots, z_{t-1} \in GF(q)$, not all zero, such that

$$\sum_{i=0}^{t-1} z_i \left(\frac{\gamma_i}{\gamma_t} - 1\right)^{q^{r-1}} = 0, \quad r = 1, \ldots, t, \tag{49}$$

and so

$$\sum_{i=0}^{t-1} z_i \gamma_i^{q^{r-1}} - \left(\sum_{i=0}^{t-1} z_i\right) \gamma_t^{q^{r-1}} = 0, \quad r = 1, \ldots, t. \tag{50}$$

Hence $(z_0, \ldots, z_{t-1}, -\sum_{i=0}^{t-1} z_i, 0, \ldots, 0)$ is a nonzero codeword of the q-ary subfield subcode of weight at most $t+1 < d(q)$, a contradiction. It follows that any $t+1$ columns of $\tilde{H}$ are linearly independent over $GF(q^m)$ and thus $sd(C) \geq t+2$. By the Singleton bound $sd(C)$ cannot be more than $t+2$, so $sd(C) = t+2$.
Case 2: $d(q) \leq t+1$. In the same way as in case 1 it can be shown that the minimum symbol distance of the code is at least $d(q)$, and thus $sd(C) = d(q)$.

□

We wish to say something about the minimum digit distance of the q-ary images of the codes over $GF(q^m)$ with a parity check matrix as in (35). To do this we need the following lemma.

Lemma 16. Let $B = \{\beta_0, \ldots, \beta_{u-1}\}$ be a set of u distinct elements of $GF(q^m)$ that are linearly independent over $GF(q)$ and such that any r distinct elements of B are linearly independent over $GF(q^a)$. If $\xi_0, \xi_1, \ldots, \xi_{u-1}$ are elements of $GF(q^a)$ that are not all zero, such that

$$\sum_{i=0}^{u-1} \beta_i \xi_i^{q^{v-1}} = 0, \quad 1 \leq v \leq t,$$

then $u \geq t + r$.

Proof. We will prove this lemma by induction on t. For $t = 1$ the assertion is trivial. So, let $t \geq 2$ and suppose

$$\sum_{i=0}^{u-1} \beta_i \xi_i^{q^{v-1}} = 0, \quad 1 \leq v \leq t. \tag{51}$$

Without loss of generality we may assume that $\xi_0 \neq 0$. Then it follows that

$$\beta_0 = \sum_{i=1}^{u-1} \beta_i \eta_i^{q^{v-1}}, \quad 1 \leq v \leq t, \tag{52}$$

where $\eta_i := -\xi_i/\xi_0$, $i = 1, \ldots, u-1$. By subtracting the $(v+1)$th equation from the vth equation in (52) and by substituting $\theta_i := \eta_i - \eta_i^q$, $i = 1, \ldots, u-1$, we obtain

$$\sum_{i=1}^{u-1} \beta_i \theta_i^{q^{v-1}} = 0, \quad 1 \leq v \leq t-1. \tag{53}$$

Now, because the elements of B are linearly independent over $GF(q)$, at least one of the θ_i is not zero. Therefore, by the induction hypothesis we have that $u - 1 \geq t - 1 + r$, and the assertion follows.

□

Theorem 17. Let C be a code over $GF(q^m)$ with parity check matrix

$$H = \begin{bmatrix} 1 & 1 & \dots & \dots & 1 \\ \eta_0 & \eta_1 & \dots & \dots & \eta_{N-1} \\ \eta_0^q & \eta_1^q & \dots & \dots & \eta_{N-1}^q \\ \eta_0^{q^2} & \eta_1^{q^2} & \dots & \dots & \eta_{N-1}^{q^2} \\ \dots & \dots & \dots & \dots & \dots \\ \eta_0^{q^{t-1}} & \eta_1^{q^{t-1}} & \dots & \dots & \eta_{N-1}^{q^{t-1}} \end{bmatrix}, \tag{54}$$

where $0 \leq t \leq m$ and $\eta_0, \dots, \eta_{N-1} \in GF(q^a)$, and let $B = \langle \beta_0, \dots, \beta_{m-1} \rangle$ be a basis of $GF(q^m)$ over $GF(q)$ such that any r distinct elements of B are linearly independent over $GF(q^a)$. If the minimum digit distance d of the code $C(B)$ is less than $2(r+t)$, then $d = d(q)$.

Proof. Suppose that $d(q) \neq d$, hence $d(q) > d$.
Let $\underline{c} = (c_{00}, \dots, c_{0(m-1)} : \dots ; c_{(N-1)0}, \dots, c_{(N-1)(m-1)})$ be a codeword of digit weight d in $C(B)$. Then there are distinct elements $j_0, \dots, j_{u-1} \in \{0, 1, \dots, m-1\}$ such that

$$|\{(i, j_b) \ : \ c_{ij_b} \neq 0, \ i = 0, 1, \dots, N-1\}| > 0, \quad b = 0, 1, \dots, u-1,$$

and

$$|\{(i, j_b) \ : \ c_{ij_b} \neq 0, \ i = 0, 1, \dots, N-1; \ b = 0, 1, \dots, u-1\}| = d. \tag{55}$$

Because $\underline{c}$ is a codeword of $C(B)$ we have

$$\sum_{i=0}^{N-1} \sum_{b=0}^{u-1} c_{ij_b} \beta_{j_b} \eta_i^{q^{v-1}} = 0, \quad v = 1, \dots, t, \tag{56}$$

$$\sum_{i=0}^{N-1} c_{ij_b} = 0, \quad b = 0, 1, \dots, u-1. \tag{57}$$

Equation (57) yields that

$$|\{i \ : \ c_{ij_b} \neq 0, \ i = 0, 1, \dots, N-1\}| \geq 2, \quad b = 0, 1, \dots, u-1, \tag{58}$$

and thus $d \geq 2u$. We define

$$\xi_b := \sum_{i=0}^{N-1} c_{ij_b} \eta_i, \quad b = 0, 1, \dots, u-1.$$

Because $d(q) > d$ all ξ_b, $b = 0, 1, \ldots, u-1$ are nonzero. Also by (56) we have that

$$\sum_{b=0}^{u-1} \beta_{j_b} \xi_b^{q^{v-1}} = 0, \quad v = 1, \ldots, t. \tag{59}$$

By Lemma 16 $u \geq t+r$ and consequently $d \geq 2(t+r)$. This proves the theorem.

□

Example 3. Let $B = \{\beta_0, \ldots, \beta_7\}$ be a normal or a polynomial basis of $GF(2^8)$ over $GF(2)$ and let $\{\xi_1, \xi_2, \xi_3, \xi_4\}$ be a basis of $GF(2^4)$ over $GF(2)$. Define $\eta := \sum_{i=0}^{4} \xi_i$ and let C be the $[6, 4]$ code over $GF(2^8)$ with parity check matrix

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & \eta & \xi_1 & \xi_2 & \xi_3 & \xi_4 \end{bmatrix}.$$

The code C has minimum symbol distance 3 and the binary subfield subcode has minimum distance 6. From Lemma 13 it follows that any two distinct elements of B are linearly independent over $GF(2^4)$. With Theorems 14 and 17 we see that the minimum distance profile (d_0, d_1, d_2) of $C(B)$ satisfies

$$d_0 = 6, \quad d_1 \geq 3, \quad d_2 = 1.$$

Lemma 18. For a normal basis $B = \{\beta, \beta^2, \ldots, \beta^{2^7}\}$ of $GF(2^8)$ over $GF(2)$, any three elements of B are linearly independent over $GF(2^2)$.

Proof. Let α be a primitive element of $GF(2^8)$ that satisfies $\alpha^8 = \alpha^4 + \alpha^3 + \alpha^2 + 1$, and let $\gamma := \alpha^{85}$. Then, γ is a primitive element of $GF(4)$. Assume there are i, j, $0 < i, j < 8$, $i \neq j$ and η_1, η_2, η_3 in $GF(4)$ such that

$$\eta_1 \beta + \eta_2 \beta^{2^i} + \eta_3 \beta^{2^j} = 0.$$

If one of η_1, η_2, η_3 is zero, then the quotient of two distinct elements of B would be in $GF(2^4)$ which is impossible (Lemma 13). Also, it is immediatly clear that not all η_k are the same. There are two

possibilities left:

1. All η_k are distinct. Then without loss of generality we have

$$\beta + \gamma\beta^{2^i} + \gamma^2\beta^{2^j} = 0. \tag{60}$$

From (60) it follows that

$$T_{8,2}(\beta) + \gamma T_{8,2}(\beta^{2^i}) + \gamma^2 T_{8,2}(\beta^{2^j}) = 0. \tag{61}$$

Because B is a normal basis, we have

$$T_{8,1}(\beta) = T_{8,1}(\beta^{2^i}) = T_{8,1}(\beta^{2^j}) = 1. \tag{62}$$

Because $T_{8,1}(\xi) = T_{2,1}(T_{8,2}(\xi))$ for all $\xi \in GF(2^8)$, it follows from (28) that

$$T_{8,2}(\beta^{2^k}) = \gamma \text{ or } \gamma^2, \quad k = 0, i, j. \tag{63}$$

With (61) it is easy to see that either all $T_{8,2}(\beta^{2^k})$ equal γ, or all $T_{8,2}(\beta^{2^k})$ equal γ^2, $k = 0, i, j$. But then at least two of $T_{8,4}(\beta)$, $T_{8,4}(\beta^{2^i})$, $T_{8,4}(\beta^{2^j})$ are the same, since the trace function $T_{m,n}$ maps a normal basis of $GF(2^m)$ over $GF(2)$ onto a normal basis of $GF(2^n)$ over $GF(2)$, and three mutually different elements of a normal basis of $GF(2^4)$ over $GF(2)$ are mapped by $T_{4,2}$ onto two different elements of a normal basis of $GF(4)$ over $GF(2)$. With (60) we see that $T_{8,4}(\beta) = T_{8,4}(\beta^{2^i}) = T_{8,4}(\beta^{2^j})$. Clearly, any three mutually distinct elements of a normal basis of $GF(2^8)$ over $GF(2)$ cannot be mapped onto the same element of $GF(2^4)$ by $T_{8,4}$. Therefore $\beta, \beta^{2^i}, \beta^{2^j}$ are not all distinct, a contradiction.

2. Two η_k are the same. Then, this implies that the code C over $GF(2^8)$ with parity check matrix

$$H = \begin{bmatrix} 1 & \gamma \end{bmatrix},$$

has a binary image $C(B)$ with minimum distance 3. It can be checked that the binary images of $\gamma\alpha^{11}$ and $\gamma^2\alpha^{11}$ with respect to the normal basis $\mathcal{N}(11)$ have weight larger than 3, and thus it follows that $(\alpha^{11\cdot 2^i}\gamma)(\mathcal{N}(11))$ has weight larger than 3 for all integers i. Analogously we can see that the binary images of $\gamma\alpha^{95\cdot 2^i}$ with respect to the normal basis $\mathcal{N}(95)$ have weight larger than 2. From this it follows that the codes $C(\mathcal{N}(11))$ and $C(\mathcal{N}(95))$ have minimum digit distance at least 4. With Corollary 9 we see that this is the case for all normal bases, and we have a contradiction. This proves the lemma.

□

Example 4. (See [6]). Let α and γ be as in the proof of the lemma above. Consider the $[3,1]$ code C over $GF(2^8)$ with parity check matrix

$$H = \begin{bmatrix} 1 & 1 & 1 \\ 1 & \gamma & \gamma^2 \end{bmatrix}.$$

The code C has symbol distance 3. Let $B = \{\beta, \ldots, \beta^{2^7}\}$ be an arbitrary normal basis of $GF(2^8)$ over $GF(2)$ and assume $C(B)$ has minimum distance profile (d_0, d_1, d_2). Clearly $d_2 = 1$ and from Lemma 18, Theorems 14 and 17 it follows that $d_0 \geq 8$, $d_1 \geq 4$. According to [10] $d_0 \leq 9$ and $d_1 \leq 5$. Because d_0 has to be even we get $d_0 = 8$. It can be seen that $(\alpha^{95}, \alpha^{180}, \alpha^{10})$ is a codeword of C and its binary image with respect to $\mathcal{N}(95)$ has weight profile $(8,4,1)$. With Corollary 9 it follows that the codes

$$C(\mathcal{N}(k)), \quad k = 9, 21, 39, 43, 55, 61, 91, 95$$

all have minimum distance profile $(8,4,1)$ and they are equivalent. For $B := \mathcal{N}(11)$ we will show that $d_1 = 5$. Clearly d_1 is the minimum distance of the binary image with respect to B of the code $\tilde{C}$ with parity check matrix $[1 \ \ \gamma]$. We have seen in the proof of Lemma 18 that there are no words in $\tilde{C}(B)$ with weight profile (4,1). Hence if there is a word of digit weight 4 in this code , then there exist i, j, k, l such that

$$(\beta^{2^i} + \beta^{2^j}) = \gamma(\beta^{2^k} + \beta^{2^l}).$$

Using a table of $GF(2^8)$ with respect to the basis $\mathcal{N}(11)$ (see [8]) we see that the set of elements of $GF(2^8)$ with binary image of weight 2 with respect to $\mathcal{N}(11)$ is the union of the conjugate classes $\mathcal{C}(1)$, $\mathcal{C}(13)$, $\mathcal{C}(127)$ and $\mathcal{C}(119)$, where

$$\mathcal{C}(i) := \{\alpha^{i \cdot 2^j} \ : \ j = 0, 1, \ldots, 7\}. \tag{64}$$

Because $2^i - 2^j \neq 0 \bmod 85$, $0 \leq i, j < 8$, $i \neq j$ it follows that no two distinct elements in $\mathcal{C}(k)$, $k = 1, 3, 127$, have their quotient in $GF(4)$. Analogously we can see that the quotient of two distinct elements of $\mathcal{C}(119)$ is never in $GF(4)$ and in a similar way we can check that the quotient of two elements in distinct conjugate

classes mentioned above is never in $GF(4)$. Hence $d_1 > 4$. Therefore, by Corollary 9 the codes

$$C(\mathcal{N}(k)), \quad k = 5, 11, 15, 29, 47, 53, 63, 87$$

have minimum distance profile $(8, 5, 1)$. These codes can correct one symbol error, correct up to three bit errors and detect four bit errors.

C. Self-orthogonal SDEC codes

A linear code C is called *self-orthogonal* whenever it is contained in its dual code $C^{\perp}$. It is called *self-dual* if it is equal to its dual. In this subsection we will construct self-orthogonal (self-dual) SDEC codes over $GF(q)$ from self-orthogonal (self-dual) codes over $GF(q^m)$. The following theorem can easily be shown.

Theorem 19. For a linear code C over $GF(q^m)$ and complementary bases A and B of $GF(q^m)$ over $GF(q)$ we have

$$(C(A))^{\perp} = C^{\perp}(B).$$

Corollary 20. For a self-orthogonal (self-dual) linear code C over $GF(q^m)$ and a self-complementary basis B of $GF(q^m)$ over $GF(q)$, the q-ary code $C(B)$ is also self-orthogonal (self-dual).

In [19] the binary images of some self-dual codes over $GF(2^m)$ were studied. Here we will consider two special classes of codes over $GF(2^m)$.

Let γ be a primitive Nth root of unity in $GF(2^m)$ and let $s \geq \frac{(N+1)}{2} + 1$ (note that N is odd). We define the code $C_{1,N,s}$ as the BCH code of length N over $GF(2^m)$ with generator polynomial

$$g_1(x) = \prod_{i=0}^{s-2}(x - \gamma^i), \tag{65}$$

and the code $C_{2,N,s}$ as the BCH code of length N over $GF(2^m)$ with generator polynomial

$$g_2(x) = \frac{g_1(x)}{(x-1)}. \tag{66}$$

Clearly, $C_{1,N,s}$ is an $[N, N-s+1]$ code over $GF(2^m)$ with symbol distance s (it is MDS). The generator polynomial of $C_{1,N,s}^{\perp}$ is given by (see [16])

$$g_1^{\perp}(x) = \prod_{i=s-1}^{N-1} (x - \gamma^{-i}) = \prod_{i=1}^{N-s+1} (x - \gamma^{i}). \tag{67}$$

From $s \geq \frac{(N+1)}{2} + 1$ it follows that $s - 2 \geq N - s + 1$, hence $C_{1,N,s}$ is self-orthogonal.

Lemma 21. The extended code $\bar{C}_{2,N,s}$ of $C_{2,N,s}$ has symbol distance s.

Proof. Clearly, $C_{2,N,s}$ has symbol distance $s-1$. Let $\underline{c}$ be a codeword of $C_{2,N,s}$ of minimum weight $s-1$ and let $\bar{\underline{c}}$ be the word of $\bar{C}_{2,N,s}$ obtained by extending $\underline{c}$. If $\bar{\underline{c}}$ has weight $s-1$, then

$$\sum_{i=0}^{N-1} c_i = 0.$$

Hence, $\bar{\underline{c}}$ is a codeword of $C_{1,N,s}$ which is a contradiction because $C_{1,N,s}$ has minimum distance s.

□

It is easy to see that if G is a generator matrix for $C_{1,N,s}$, then

$$G^* = \begin{bmatrix} 1 & 1 & \dots & 1 \\ & & & 0 \\ & G & & \vdots \\ & & & 0 \end{bmatrix} \tag{68}$$

is a generator matrix for $\bar{C}_{2,N,s}$. The code $C_{1,N,s}$ is self-orthogonal. Hence it is easy to see that $G^*(G^*)^T = 0$ and thus $\bar{C}_{2,N,s}$ is also self-orthogonal. If $s = \frac{N+1}{2} + 1$, then $\bar{C}_{2,N,s}$ is self-dual.

By Corollary 20 we have the following result.

Corollary 22. The binary images of the codes $C_{1,N,s}$ and $\bar{C}_{2,N,s}$ with respect to self-complementary bases of $GF(2^m)$ over $GF(2)$ are self-orthogonal if $s \geq \frac{N+1}{2} + 1$.

To prove Lemma 24 we will need the following simple result.

Lemma 23. If in a binary self-orthogonal code $\underline{c}_1$ and $\underline{c}_2$ are codewords of weight divisible by 4, then their sum $\underline{c}_1 + \underline{c}_2$ also has weight divisible by 4.

Proof. Clearly,

$$wt(\underline{c}_1 + \underline{c}_2) = wt(\underline{c}_1) + wt(\underline{c}_2) - 2(\underline{c}_1, \underline{c}_2),$$

where (,) denotes the standard inner product in real vector spaces. In this case $(\underline{c}_1, \underline{c}_2)$ is even (C is self-orthogonal) and the lemma follows.

□

If all the codewords of a binary code C have weight divisible by 4, then C is called *doubly even.* We have the following lemma.

Lemma 24. Let N be a divisor of $2^m - 1$ and let $\Omega = \langle \omega_0, \ldots, \omega_{m-1} \rangle$ be a self-complementary basis of $GF(2^m)$ over $GF(2)$ such that for all divisors ν of N, $\nu > 1$, the binary vectors

$$\underline{g_i} := (\underline{(\omega_i)(\Omega)}, \underline{(\omega_i\gamma)(\Omega)}, \ldots, \underline{(\omega_i\gamma^{\nu-1})(\Omega)}), \quad i = 0, 1, \ldots, m-1,$$

where γ is a primitive νth root of unity in $GF(2^m)$, all have digit weight divisible by 4. Then the codes $C_{1,N,s}(\Omega)$, where N is a divisor of $2^m - 1$ and $s \geq \frac{N+1}{2} + 1$, are doubly even. If $N + 1$ is divisible by 4 and $s \geq \frac{N+1}{2} + 1$, then the code $\bar{C}_{2,N,s}(\Omega)$ is also doubly even.

Proof. 1. A generator matrix for the code $C_{1,N,s}$ is given by

$$G = \begin{bmatrix} 1 & \gamma & \ldots & \gamma^{N-1} \\ 1 & \gamma^2 & \ldots & \gamma^{2(N-1)} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \gamma^{N-s+1} & \ldots & \gamma^{(N-s+1)(N-1)} \end{bmatrix},$$

where γ is a primitive Nth root of unity in $GF(2^m)$. By the assumption of the lemma all the rows of the matrix $G(\Omega)$ have weight divisible by 4. From Corollary 22 and Lemma 23 it follows

that $C_{1,N,s}(\Omega)$ is doubly even.
2. A generator matrix of $\bar{C}_{2,N,s}$ is given by

$$G^* = \begin{bmatrix} 1 & 1 & \dots & 1 \\ & & & 0 \\ & G & & \vdots \\ & & & 0 \end{bmatrix}.$$

The first m rows of $G^*(\Omega)$ have weight $N+1$. Hence if $N+1$ is divisible by 4 all the rows of $G^*(\Omega)$ have weight divisible by 4, and the assertion follows with Corollary 22 and Lemma 23.

□

Lemma 25. Let $\Omega = \langle \omega_0, \dots, \omega_{m-1} \rangle$ be a self-complementary basis of $GF(2^m)$ over $GF(2)$, let γ be a primitive element of a subfield $GF(2^k)$ of $GF(2^m)$, $k > 1$ and let $N = 2^k - 1$. Then, the vectors

$$(\underline{(\omega_i)(\Omega)}, \underline{(\omega_i\gamma)(\Omega)}, \dots, \underline{(\omega_i\gamma^{N-1})(\Omega)}), \quad i = 0, 1, \dots, m-1$$

all have digit weight divisible by 4.

Proof. Let $0 \le t \le m-1$ and define

$$\underline{c} = (c_{00}, \dots, c_{0(m-1)}, \dots, c_{(N-1)0}, \dots c_{(N-1)(m-1)}) :=$$

$$(\underline{(\omega_t)(\Omega)}, \underline{(\omega_t\gamma)(\Omega)}, \dots, \underline{(\omega_t\gamma^{N-1})(\Omega)}).$$

Then, by equation (33) we have

$$c_{ij} = T_{m,1}(\omega_t\omega_j\gamma^i) = T_{k,1}(\gamma^i T_{m,k}(\omega_t\omega_j)),$$

$$i = 0, 1, \dots, N-1; j = 0, 1, \dots, m-1. \tag{69}$$

We define the binary N-vector X_j by

$$X_j := (c_{0j}, \dots, c_{(N-1)j}), \quad j = 0, 1, \dots, m-1. \tag{70}$$

If $T_{m,k}(\omega_t\omega_j) = 0$, then $wt(X_j) = 0$. If $T_{m,k}(\omega_t\omega_j) \neq 0$, it is easy to see that

$$\{\gamma^i T_{m,k}(\omega_t\omega_j) \ : \ i = 0, 1, \dots, N-1\} = GF(2^k) \setminus \{0\}$$

and thus $wt(X_j) = 2^{k-1}$. Hence, if $k > 2$, the vector $\underline{c}$ has weight divisible by 4. We now consider the case $k = 2$, so let γ be a primitive element of $GF(2^2)$. Because $T_{m,1}(\omega_i\omega_j) = \delta_{ij}$, $i, j = 0, 1, \ldots, m-1$, it is easy to see that

$$T_{m,2}(\omega_t\omega_j) = 0 \text{ or } 1, \quad j = 0, 1, \ldots, m-1, \; j \neq t, \tag{71}$$

Without loss of generality we may assume that $T_{m,2}((\omega_t)^2) = \gamma^2$. Clearly $\sum_{j=0}^{m-1} \omega_j = 1$, hence

$$\sum_{j=0}^{m-1} T_{m,2}(\omega_t\omega_j) = T_{m,2}(\omega_t) = \gamma \text{ and } \sum_{j\neq t} T_{m,2}(\omega_t\omega_j) = 1. \tag{72}$$

From (71) and (72) it follows that the number of j, $0 \leq j \leq m-1$, such that $T_{m,2}(\omega_t\omega_j) \neq 0$ is even . All X_j have weight 0 or 2, hence the assertion follows.

□

Lemmas 24 and 25 imply the following result.

Corollary 26. If k is a divisor of m such that $k > 1$ and $N := 2^k - 1$ is a prime and if $s \geq 2^{k-1} + 1$, then the codes $C_{1,N,s}(\Omega)$ and $\bar{C}_{2,N,s}(\Omega)$, where Ω is a self-complementary basis of $GF(2^m)$ over $GF(2)$, are self-orthogonal and doubly even.

Example 5. ([20]). Let C be the [8,4] extended Reed-Solomon code over $GF(2^3)$ with generator matrix

$$G = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 & 0 \\ 1 & \alpha^2 & \alpha^4 & \alpha^6 & \alpha & \alpha^3 & \alpha^5 & 0 \\ 1 & \alpha^3 & \alpha^6 & \alpha^2 & \alpha^5 & \alpha & \alpha^4 & 0 \end{bmatrix},$$

where α is a primitive element of $GF(2^3)$ that satisfies $\alpha^3 + \alpha + 1 = 0$. Clearly C has symbol distance 5. Let $\Omega = \langle \alpha^3, \alpha^6, \alpha^5 \rangle$ be the (unique) self-complementary basis of $GF(2^3)$ over $GF(2)$ (note that it is also normal). Then, by Corollary 26 and [10] the minimum distance of $C(\Omega)$ is 8. Hence $C(\Omega)$ is equivalent to the extended binary Golay code ([16,Chapter 20]). This code has minimum distance profile (8,5,3,2,1).

Example 6. Let α be a primitive element of $GF(2^6)$ that is a zero of the polynomial x^6+x+1. Then, the normal basis $\mathcal{N}(23)$ is self-complementary. Let γ be a primitive element of $GF(2^2)$ and let C be the [3,1] code over $GF(2^6)$ with parity check matrix

$$H = \begin{bmatrix} 1 & 1 & 1 \\ 1 & \gamma & \gamma^2 \end{bmatrix}.$$

This code has symbol distance 3 and by Theorem 17 the minimum distance d_0 of the binary code $C(\mathcal{N}(23))$ is at least 6 (It is easy to see that any two distinct elements of $\mathcal{N}(23)$ are linearly independent over $GF(2^2)$). From Corollary 26 it follows that $C(\mathcal{N}(23))$ is self-orthogonal and doubly even, hence $d_0 \geq 8$. Theorem 14 and [10] yield that the code has minimum distance profile (8,4,1) and it is optimal.

Example 7. Let α be as in Example 6, let η be a primitive element of $GF(2^3)$ and let C be the [7,3] BCH code over $GF(2^6)$ with zeros $1, \eta, \eta^2, \eta^3$. This code has symbol distance 5 and it is self-orthogonal. Hence, the code $C(\mathcal{N}(23))$ is self-orthogonal and doubly even, and its digit distance d_0 is at least 8. A rather tedious calculation shows that there is no word of digit weight 8 in $C(\mathcal{N}(23))$. Hence, the minimum distance profile of $C(\mathcal{N}(23))$ is at least (12,6,3,2,1). The true minimum distance profile is (12,7,5,2,1) and the code is 0-optimal.

Lemma 27. Let m be even and let $\Omega = \{\omega_0, \ldots, \omega_{m-1}\}$ be a self-complementary basis of $GF(2^m)$ over $GF(2)$ such that for all $i = 0, 1, \ldots, \frac{m}{2} - 1$ there is a $1 \leq j_i \leq m-1$ such that

$$\omega_{2i+1} = \omega_{2i}^{2^{j_i}}. \tag{73}$$

If γ is a primitive Nth root of unity in $GF(2^m)$ that does not occur in any proper subfield of $GF(2^m)$, then the vectors

$$\underline{g_i} = (\underline{(\omega_i)(\Omega)}, \underline{(\omega_i\gamma)(\Omega)}, \ldots, \underline{(\omega_i\gamma^{N-1})(\Omega)}), \quad i = 0, 1, \ldots, m-1, \tag{74}$$

all have digit weight divisible by 4.

Proof. Let

$$\underline{c} = (c_{00}, \ldots, c_{0(m-1)}, \ldots, c_{(N-1)0}, \ldots, c_{(N-1)(m-1)}) =$$

$$(\underline{(1)(\Omega)}, \underline{(\gamma)(\Omega)}, \ldots, \underline{(\gamma^{N-1})(\Omega)}),$$

and define

$$X_j := (c_{0j}, c_{1j}, \ldots, c_{(N-1)j}) =$$

$$(T_{m,1}(\omega_j), T_{m,1}(\omega_j\gamma), \ldots, T_{m,1}(\omega_j\gamma^{N-1})), \quad j = 0, 1, \ldots, m-1.$$

Clearly, $wt(X_j) = 0 \bmod 2$ for all $j = 0, 1, \ldots, m-1$ and by (73) we have that $wt(X_{2j}) = wt(X_{2j+1})$, $j = 0, 1, \ldots, \frac{m}{2} - 1$. It follows that $wt(\underline{c}) = 0 \bmod 4$. Because γ does not occur in a proper subfield of $GF(2^m)$ we see that $\Gamma := \langle 1, \gamma, \ldots, \gamma^{m-1} \rangle$ is a basis of $GF(2^m)$ over $GF(2)$ and thus

$$\omega_i = \sum_{j=0}^{m-1} x_{ij}\gamma^j, \quad x_{ij} \in GF(2), \quad i, j = 0, 1, \ldots, m-1.$$

But then

$$\underline{g_i} = \textstyle\sum_{j=0}^{m-1} x_{ij}(\underline{(\gamma^j)(\Omega)}, \underline{(\gamma^{j+1})(\Omega)}, \ldots, \underline{(\gamma^{j+N-1})(\Omega)}),$$

$$i = 0, 1, \ldots, m-1. \tag{75}$$

By the above, all the vectors on the right hand side of (75) have weight divisible by 4. Furthermore, these vectors are orthogonal. Hence $\underline{g_i}$ also has weight divisible by 4 ($i = 0, 1, \ldots, m-1$).

□

Example 8. Let α be a primitive element of $GF(2^4)$ such that $\alpha^4 = \alpha + 1$. Then it can easily be seen that $\Omega = \langle \alpha^3, \alpha^{12}, \alpha^7, \alpha^{13} \rangle$ is a self-complementary basis of $GF(2^4)$ over $GF(2)$ and that it satisfies the condition of Lemma 27. Let γ be a primitive 5th root of unity in $GF(2^4)$ and let C be the [5,2] BCH code over $GF(2^4)$ with roots $1, \gamma, \gamma^2$. The code C has symbol distance 4 and by Theorem 17 the minimum digit distance of $C(\Omega)$ is at least 6. The code $C(\Omega)$ is self-orthogonal and by Lemmas 24 and 27 it is doubly even. Hence it has a minimum distance profile of at least (8,4,2,1) and this is in fact the true minimum distance profile. The code is 0-optimal.

Example 9. (See [21]) Let α and Ω be as in Example 8 and let C be a self-dual [16,8] extended Reed-Solomon code over $GF(2^4)$.

The code C has symbol distance 9. The code $C(\Omega)$ is self-dual and by Lemmas 24, 25 and 27 it is doubly even . It follows that it has a minimum digit distance d_0 of at least 12. By [17] we have $d_0 \leq 4[\frac{16 \cdot 4}{24}] + 4 = 12$, hence d_0 equals 12. The code has minimum distance profile $(12, 8, 7, 6, 5, 4, 3, 2, 1)$.

Example 10. Let α be as in Examples 6 and 7, let η be a primitive 9th root of unity in $GF(2^6)$ and let C be the [9,4] BCH code over $GF(2^6)$ with roots $1, \eta, \eta^2, \eta^3, \eta^4$. This code has symbol distance 6. The normal basis $\mathcal{N}(15)$ of $GF(2^6)$ over $GF(2)$ is self-complementary and thus $C(\mathcal{N}(15))$ is self-orthogonal. By Lemmas 24, 25 and 27 it follows that $C(\mathcal{N}(15))$ is doubly even. A rather tedious calculation shows that there is no codeword of digit weight 8 in the code. Hence the minimum distance profile of the code is at least (12,6,4,3,2,1). This is the true minimum distance profile.

Lemma 28. Let N be a divisor of $2^m - 1$ such that $N \neq 2^u + 1$ for all $u = 1, \ldots, m-1$. If γ is a primitive Nth root of unity in $GF(2^m)$ and $\Omega = \langle \omega_0, \omega_1, \ldots, \omega_{m-1} \rangle$ is a self-complementary basis of $GF(2^m)$ over $GF(2)$, then the vectors $\underline{g_i}$ given in (74) $(i = 0, 1, \ldots, m-1)$ all have digit weight divisible by 4.

Proof. Let

$$X_j := (T_{m,1}(\omega_i \omega_j), T_{m,1}(\omega_i \omega_j \gamma), \ldots, T_{m,1}(\omega_i \omega_j \gamma^{N-1})),$$

$$j = 0, 1, \ldots, m-1,$$

and let $h(x) := \sum_{u=0}^{k} h_u x^u$ be the minimal polynomial of γ. Then,

$$< X_j, (h_0, \ldots, h_k, 0, \ldots, 0) >= \sum_{u=0}^{k} h_u T_{m,1}(\omega_i \omega_j \gamma^u)$$

$$= T_{m,1}(\omega_i \omega_j h(\gamma)) = 0.$$

It follows that X_j $(j = 0, 1, \ldots, m-1)$ are codewords of the binary cyclic code C of length n with zeros $\{\gamma^i \mid i = 0, 1, \ldots, N-1\} \setminus R$, where $R := \{\gamma^{-1}, \gamma^{-2}, \ldots, \gamma^{-2^{k-1}}\}$. Because of the condition on N we see that no two elements of R have product 1. Hence by McEliece's theorem [18] it follows that all the codewords of C have weight divisible by 4. Therefore all X_j have weight divisible by 4 and thus all $\underline{g_i}$ have weight divisible by 4.

□

Corollary 29. If m is odd, N a divisor of $2^m - 1$, $s \geq \frac{N+1}{2} + 1$ and if Ω is a self-complementary basis of $GF(2^m)$ over $GF(2)$, then the code $C_{1,N,s}(\Omega)$ is self-orthogonal and doubly even. If $N+1$ is divisible by 4, then the code $\bar{C}_{2,N,s}(\Omega)$ is also self-orthogonal and doubly even.

Proof. If m is odd there exists no u such that $2^u + 1$ is a divisor of $2^m - 1$. With Lemmas 24 and 28 the assertion follows.

□

We conjecture that Corollary 29 also holds if m is even.

We conclude this section with the following interesting example (see also [9]).

Example 11. Let α be a primitive element of $GF(3^2)$ that satisfies $\alpha^2 = -\alpha + 1$ and let $\gamma = 1 - \alpha$. Then, γ is a primitive 4th root of unity in $GF(3^2)$. Let C be the [4,3] BCH code over $GF(3^2)$ with generator matrix

$$G = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \gamma & -1 & -\gamma \\ 1 & -1 & 1 & -1 \end{bmatrix},$$

and let $\hat{C}$ be the [6,3] code over $GF(3^2)$ with generator matrix

$$\hat{G} = \begin{bmatrix} & \gamma & 0 \\ G & 0 & 0 \\ & 0 & \gamma \end{bmatrix}.$$

It can easily be seen that $\hat{C}$ has symbol distance 4 and that it is not self-dual. For the basis $A = \{1, \alpha\}$ of $GF(3^2)$ over $GF(3)$ it can easily be seen that

$$\hat{G}(A)(\hat{G}(A))^T = 0.$$

Hence $\hat{C}(A)$ is self-dual and it has minimum digit distance at least 6 (the codewords of $\hat{C}(A)$ have digit weight divisible by 3). From [16,Chapter 20], it follows that $\hat{C}(A)$ is equivalent to the

extended ternary Golay code and it can easily be seen that its minimum distance profile is (6,4,2,1). Note that $GF(3^2)$ has no self-complementary basis [27].

D. SDEC codes from codes with smaller symbols

Construction. Let C be an $\langle Nr, Kr, m\rangle$ SDEC code with symbol distance s and minimum distance profile $(d_0, d_1, \ldots, d_{s-1})$. Then, by taking r consecutive symbols together we obtain from C an $\langle N, K, rm\rangle$ SDEC code with symbol distance at least $\lceil \frac{s}{r} \rceil$ and minimum distance profile at least $(d_0, d_r, d_{2r}, \ldots)$.

For example, let α be a primitive element of $GF(q^m)$ and let C be the $[q^m, q^m - s + 1]$ extended Reed-Solomon code over $GF(q^m)$ with parity check matrix

$$H = \begin{bmatrix} 1 & 1 & \ldots & 1 & 1 \\ 1 & \alpha & \ldots & \alpha^{q^m-2} & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \alpha^{s-2} & \ldots & \alpha^{(q^m-2)(s-2)} & 0 \end{bmatrix}. \tag{76}$$

Note that C is cyclic and that it has symbol distance s (see [16,Ch.10,Sec.3]). Let B be a basis of $GF(q^m)$ over $GF(q)$ and let r be a divisor of $s-1$. Then, by shortening C and by applying the construction mentioned above, we obtain from $C(B)$ $\langle N, N - \frac{s-1}{r}, rm\rangle$ SDEC codes where $q^m - s + 1 \leq Nr \leq q^m$. These codes have symbol distance $\lceil \frac{s}{r} \rceil$ and minimum distance profile at least $(s, s-r, s-2r, \ldots)$. If $q = 2$, then the first component of this minimum distance profile is even. These codes can be decoded by decoding methods for (extended) Reed-Solomon codes over $GF(q^m)$.

Example 12. For $q = 2$, $m = 4$, $s = 5$, $r = 2$ and $3 \leq N \leq 8$, we obtain binary $\langle N, (N-2), 8\rangle$ SDEC codes with minimum distance profile at least (6,3,1). For $N = 6, 7, 8$ the first component is the best known [30]. The codes are {(1,0),(0,2)}-correcting and thus

can correct single symbol errors and double bit errors.

Example 13. For $q = 2$, $m = 4$, $s = 7$, $r = 2$ and $4 \leq N \leq 8$ we obtain binary $\langle N, (N-3), 8\rangle$ SDEC codes with minimum distance profile at least (8,5,3,1). They are {(1,0),(0,3)}-correcting and {(0,4)} detecting or {(1,1),(0,3)}-correcting.

Example 14. Take $q = 2$, $m = 4$, $s = 9$, $r = 2$ and $5 \leq N \leq 8$. If we take for B a self-complementary basis we obtain $\langle N, N-4, 8\rangle$ SDEC codes that are self-orthogonal, doubly even and have minimum distance profile at least (12,7,5,3,1). The codes are {(1,0),(0,5)}-correcting and {(1,1),(0,6)}-detecting or {(1,1),(0,5)}-correcting or {(2,0),(1,2),(0,4)}-correcting. By taking $r = 4$ we also obtain a $\langle 4, 2, 16\rangle$ self-dual doubly even binary SDEC code C with minimum distance profile at least (12,5,1). The code can correct any symbol error and up to four bit errors and if a symbol is erased, it can correct two bit errors. To correct these error patterns, decoding can be done with an extended RS code decoder.

Example 15. For $q = 2$, $m = 5$, $s = 5$, $r = 2$, $3 \leq N \leq 16$ we obtain $\langle N, N-2, 10\rangle$ binary SDEC codes with minimum distance profile at least (6,3,1). For $N \geq 9$ the first component is the best known [30].

Of course we can apply the construction mentioned above to other codes.

Example 16. Let C be the [11,6] cyclic code over $GF(2^{10})$ with zeros η, η^3, η^4, η^5, η^9, where η is a primitive 11-th root of unity in $GF(2^{10})$. Note that $H = \{\eta, \ldots, \eta^{10}\}$ is a normal basis of $GF(2^{10})$ over $GF(2)$. The extended code $\bar{C}$ of C is a [12,6] self-dual code over $GF(2^{10})$ and it has symbol distance 6. The finite field $GF(2^{10})$ has four self-complementary normal bases over $GF(2)$, namely $\mathcal{N}(17)$, $\mathcal{N}(55)$, $\mathcal{N}(19)$ and $\mathcal{N}(221)$, all indexed with respect to a primitive element of $GF(2^{10})$ that satisfies $\alpha^{10} = \alpha^3 + 1$. Let $\mathcal{N}$ be one of these bases. By Corollary 20 and Lemmas 23 and 27 the binary code $\bar{C}(\mathcal{N})$ is self-dual and doubly even. Let ω be a primitive element of $GF(2^2)$ and let $\Omega = \{\omega, \omega^2\}$ be the self-complementary normal basis of $GF(2^2)$. Without loss of generality we may assume that $\underline{\omega}(\mathcal{N}) = (1010101010)$ and it is easy

to see that for all $\beta \in GF(2^2)$ we have $wt(\underline{\beta(\mathcal{N})}) = 5 \cdot wt(\underline{\beta(\Omega)})$. This implies that the binary image of the $GF(2^2)$ subfield subcode $\tilde{C}$ of $\bar{C}$ with respect to the basis Ω is also self-dual and doubly even. The code $\tilde{C}$ has symbol distance 6, minimum distance profile (8,6,4,3,2,1), and it is equivalent to the extended binary Golay code. With the construction method mentioned above we obtain a $\langle 6, 3, 4\rangle$ binary SDEC code with minimum distance profile (8,4,2). This code is optimal. It can correct any symbol error and up to three bit errors. If one symbol is erased, it can correct one bit error and detect one symbol error and two bit errors.

E. SDEC codes from shortened cyclic codes

In this subsection we will give some bounds on the minimum distance profile of SDEC codes constructed from shortened cyclic codes.

We start by considering SDEC codes with two parity symbols. Let C be the $[N, N-2]$ code over $GF(q^m)$ with parity check matrix

$$H = \begin{bmatrix} 1 & 1 & \dots & 1 \\ \alpha_0 & \alpha_1 & \dots & \alpha_{N-1} \end{bmatrix},$$

where $\alpha_0, \dots, \alpha_{N-1}$ are distinct eth roots of unity in $GF(q^m)$. Let ω be a primitive νth root of unity in $GF(q^m)$ such that e is a divisor of ν and such that Ω defined by

$$\Omega = \langle \omega^{k_0}, \dots, \omega^{k_{m-1}} \rangle$$

is a basis of $GF(q^m)$ over $GF(q)$, for some $0 \leq k_0 < \dots < k_{m-1} < \nu$. Suppose also that all $\alpha_i \omega^{k_j}$, $i = 0, 1, \dots, N-1$, $j = 0, 1, \dots, m-1$ are distinct. We denote the ith position of a codeword of a cyclic code of length ν by ω^i.

Theorem 30. Let C_ν be the cyclic code of length ν over $GF(q)$ with zeros $1, \omega, \omega^e, \omega^{2e}, \omega^{3e}, \dots, \omega^{\nu-e}$ and let $\tilde{C}_\nu$ be the code over $GF(q)$ of length Nm obtained by shortening C_ν in the positions

$$\{\omega^i \mid 0 \leq i < \nu\} \setminus \{\alpha_i \omega^{k_j} \mid i = 0, 1, \dots N-1;\ j = 0, 1, \dots, m-1\}.$$

Then, $C(\Omega)$ is equivalent to a subcode of $\tilde{C}_\nu$. If there are integers

$w_1, \ldots, w_t$ such that

$$\tilde{\Omega} := \{\sum_{i=1}^{t} \omega^{w_i k_0 e}, \ldots, \sum_{i=1}^{t} \omega^{w_i k_{m-1} e}\}$$

is a basis of $GF(q^m)$ over $GF(q)$, then $C(\Omega)$ is equivalent to $\tilde{C}_\nu$.

Proof. Let $\underline{c} = (c_{00}, \ldots, c_{0(m-1)}, \ldots, c_{(N-1)0}, \ldots, c_{(N-1)(m-1)})$ be a codeword of $C(\Omega)$. The first row of H implies that

$$\sum_{j=0}^{m-1} \sum_{i=0}^{N-1} c_{ij} \omega^{k_j} = 0.$$

Because Ω is a basis it follows that $\sum_{i=0}^{N-1} c_{ij} = 0$ for all $j = 0, 1, \ldots, m-1$. Hence,

$$\sum_{i=0}^{N-1} \sum_{j=0}^{m-1} c_{ij} \alpha_i \omega^{k_j} = 0 \tag{77}$$

and

$$\sum_{i=0}^{N-1} c_{ij} = 0, \quad j = 0, 1, \ldots, m-1. \tag{78}$$

There are distinct $0 \leq u_0, \ldots, u_{N-1} < e$ such that

$$\alpha_i = \omega^{\frac{\nu}{e} u_i}, \quad i = 0, 1, \ldots, N-1. \tag{79}$$

For all vectors $\underline{y} = (y_{00}, \ldots, y_{0(m-1)}, \ldots, y_{(N-1)0}, \ldots, y_{(N-1)(m-1)})$ in $(GF(q))^{mN}$ we define the polynomial $Y(x)$ by

$$Y(x) := \sum_{i=0}^{N-1} \sum_{j=0}^{m-1} y_{ij} x^{\frac{\nu}{e} u_i + k_j}.$$

1. If $\underline{c}$ is a codeword of $C(\Omega)$, it immediatly follows from (77) that $C(\omega) = 0$. Now let $0 \leq z < \frac{\nu}{e}$. Then (78) yields

$$C(\omega^{ze}) = \sum_{i=0}^{N-1} \sum_{j=0}^{m-1} c_{ij} \omega^{\nu u_i z + k_j z e} = \sum_{j=0}^{m-1} (\sum_{i=0}^{N-1} c_{ij}) \omega^{k_j z e} = 0.$$

All $\alpha_i \omega^{k_j}$ are distinct. Hence $C(x)$ is a codeword of $\tilde{C}_\nu$ and the first part of the theorem follows.

2. Let $\tilde{\underline{c}}$ be a codeword of $\tilde{C}_\nu$. We define the word $\underline{c} \in (GF(q))^{Nm}$ as $\underline{c} := (c_{00}, \ldots, c_{0(m-1)}, \ldots, c_{(N-1)0}, \ldots, c_{(N-1)(m-1)})$ where c_{ij} is the element of $\tilde{\underline{c}}$ on position $\alpha_i \omega^{k_j}$. From the definition of $\tilde{C}_\nu$ it follows that $C(\omega) = 0$. Also, for all $0 \leq z < \frac{\nu}{e}$ we have $C(\omega^{ze}) = 0$. Hence,

$$0 = \sum_{p=1}^{t} C(\omega^{w_p e}) = \sum_{p=1}^{t} \sum_{i=0}^{N-1} \sum_{j=0}^{m-1} c_{ij} \omega^{w_p k_j e} =$$

$$= \sum_{j=0}^{m-1} \Big(\sum_{i=0}^{N-1} c_{ij} \Big) \Big(\sum_{p=1}^{t} \omega^{w_p k_j e} \Big).$$

If $\tilde{\Omega}$ is a basis, then $\sum_{i=0}^{N-1} c_{ij} = 0$ for all $j = 0, 1, \ldots, m-1$ and thus $\underline{c}$ is a codeword of $C(\Omega)$. This yields the second part of the theorem.

□

This theorem implies that the minimum digit distance of the code $C(\Omega)$ is at least the minimum distance of the cyclic code C_ν.

Example 17. Let C be the code of Example 4. Clearly the elements of the normal bases $\mathcal{N}(5)$ and $\mathcal{N}(95)$ are 51st roots of unity in $GF(2^8)$ and γ is a 3rd root of unity. Also, all $\alpha^{k \cdot 2^j} \gamma^i$, $j = 0, 1, \ldots, 7$, $i = 0, 1, 2$ are distinct 51st roots of unity in $GF(2^8)$ ($k = 5, 95$). Therefore, Theorem 30 yields that the minimum distances of $C(\mathcal{N}(5))$ and $C(\mathcal{N}(95))$ are at least the minimum distance of the binary cyclic code of length 51 with zeros $1, \omega, \omega^3, \omega^9$ (ω being a 51st root of unity in $GF(2^8)$), which is 8 (see [22,Appendix D]). Then, with Corollary 9 and [10] we see that all the normal bases yield binary images of C with minimum distance 8.

Example 18. Let C be the code over $GF(2^{10})$ with parity check matrix

$$H = \begin{bmatrix} 1 & 1 & 1 \\ 1 & \gamma & \gamma^2 \end{bmatrix},$$

where $\gamma = \alpha^{341}$, α being a primitive element of $GF(2^{10})$ that satisfies $\alpha^{10} = \alpha^3 + 1$. Then γ is a primitive element of $GF(4)$. Let B be one of the normal bases $\mathcal{N}(31)$, $\mathcal{N}(93)$ or one of the polynomial bases $\mathcal{P}(31) := \{1, \alpha^{31}, \ldots, \alpha^{9 \cdot 31}\}$, $\mathcal{P}(93)$ or $\mathcal{P}(155)$. The elements of B are 33rd roots of unity and furthermore the set

$\{\beta\gamma^i \mid \beta \in B,\ i = 0,1,2\}$ has cardinality 30. By Theorem 30 the binary images of C with respect to these bases have a minimum distance that is at least the minimum distance of the binary cyclic code of length 33 with zeros $1, \omega, \omega^3, \omega^9$ (ω a primitive 33rd root of unity), which is 10 ([22,Appendix D]). In fact the minimum distance profile of the codes with respect to the bases $\mathcal{N}(93)$ and $\mathcal{P}(93)$ is $(10,6,1)$ and with respect to the bases $\mathcal{N}(31)$, $\mathcal{P}(31)$ and $\mathcal{P}(155)$ it is $(10,5,1)$.

Example 19. Let γ be a primitive 5th root of unity and let C be the [5,2] code over $GF(2^{12})$ with parity check matrix

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & \gamma & \gamma^2 & \gamma^3 & \gamma^4 \end{bmatrix}.$$

If Ω is a basis of $GF(2^{12})$ over $GF(2)$ solely consisting of 13th roots of unity (for example a normal or polynomial basis), then C and Ω satisfy the conditions of Theorem 30 with $e = 5$ and $\nu = 65$. It follows that the minimum distance of $C(\Omega)$ is at least the minimum distance of the binary cyclic code of length 65 and zeros $1, \eta, \eta^5$ (η being a 65th root of unity), which is 8 [22]. With Theorem 14 it follows that $C(\Omega)$ has minimum distance profile at least (8,4,1).

Example 20. Let α be a primitive element of $GF(2^{18})$ that satisfies $\alpha^{18} = \alpha^7 + 1$ and let $\gamma := \alpha^{87381}$. Then γ is a primitive element of $GF(4)$. Let C be the code over $GF(2^{18})$ with parity check matrix

$$H = \begin{bmatrix} 1 & 1 & 1 \\ 1 & \gamma & \gamma^2 \end{bmatrix}.$$

If Ω is a basis of $GF(2^{18})$ over $GF(2)$ solely consisting of 19th roots of unity (for example a normal or polynomial basis), then C and Ω satisfy the conditions of Theorem 30 with $e = 3$ and $\nu = 57$. It follows that the minimum digit distance of the binary code $C(\Omega)$ is at least the minimum distance of the binary cyclic code of length 57 with zeros $1, \eta, \eta^3$, which is 14 [22] (η is a primitive 57th root of unity). In fact $C(\Omega)$ has minimum distance profile (14,7,1).

Example 21. Let C be the code over $GF(2^{28})$ with parity check

matrix

$$H = \begin{bmatrix} 1 & 1 & 1 \\ 1 & \gamma & \gamma^2 \end{bmatrix},$$

where γ is a primitive element of $GF(4)$. If Ω is a basis of $GF(2^{28})$ over $GF(2)$ consisting solely of 29th roots of unity (normal or polynomial basis), then C and Ω satisfy the conditions of Theorem 30 with $e = 3$ and $\nu = 87$. It follows that the minimum digit distance of $C(\Omega)$ is at least the minimum distance of the binary cyclic code of length 87 with roots 1, η, η^3 which is 22 [25] (η is a primitive 87th root of unity). It follows that $C(\Omega)$ has minimum distance profile at least (22,11,1).

These $[N, N-2]$ extended codes are not very good if N becomes large.

Theorem 31. Let C be an $[N, N-2]$ code over $GF(q^m)$ with parity check matrix

$$H = \begin{bmatrix} 1 & \dots & 1 \\ \alpha_0 & \dots & \alpha_{N-1} \end{bmatrix},$$

where all α_i are distinct in $GF(q^m)$. Let $B = \langle \beta_0, \dots, \beta_{m-1} \rangle$ be a basis of $GF(q^m)$ over $GF(q)$. If

$$\binom{N}{2} > \frac{q^m - 1}{m(q-1)},$$

then the minimum digit distance of $C(B)$ is 4.

Proof. Suppose that $C(B)$ has minimum distance digit larger than 4. Then $C(B)$ contains no codeword of digit weight 4, and hence

$$x\beta(\alpha_i + \alpha_j), \quad x \in GF(q),\ x \neq 0, \quad \beta \in B, \quad 0 \leq i < j \leq N-1,$$

must be distinct nonzero elements of $GF(q^m)$. There are $(q-1)m\binom{N}{2}$ of these elements and the theorem follows.

□

For example, if $q = 2$ and $m = 8$ it follows that if $C(B)$ has minimum digit distance larger than 4, then N is at most 8.

Now, we generalize the construction given above.

Construction. Let $r \geq m$, and let ν and e be integers such that ν and e are divisors of $q^r - 1$ and e is a divisor of ν. Let $\Gamma = \{\gamma_0, \ldots \gamma_{N-1}\}$ be a set of distinct eth roots of unity and let $\Omega = \langle \omega_0, \ldots, \omega_{m-1} \rangle$ be a set of distinct νth roots of unity that are linearly independent over $GF(q)$ and such that the set

$$A = \Gamma\Omega = \{\gamma_i \omega_j \mid i = 0, 1, \ldots, N-1; j = 0, 1, \ldots, m-1\} \quad (80)$$

has cardinality Nm. Let H be the following (partitioned) matrix

$$\left[\begin{array}{ccccccccccc} & I_m & & & \cdots & & & & I_m & & \\ \omega_0\gamma_0 & \cdots & \omega_{m-1}\gamma_0 & & & & \omega_0\gamma_{N-1} & \cdots & \omega_{m-1}\gamma_{N-1} \\ (\omega_0\gamma_0)^{a_1} & \cdots & (\omega_{m-1}\gamma_0)^{a_1} & \cdots & \cdots & & (\omega_0\gamma_{N-1})^{a_1} & \cdots & (\omega_{m-1}\gamma_{N-1})^{a_1} \\ \cdots & \cdots & \cdots & \cdots & \cdots & & \cdots & \cdots & \cdots \\ (\omega_0\gamma_0)^{a_t} & \cdots & (\omega_{m-1}\gamma_0)^{a_t} & & & & (\omega_0\gamma_{N-1})^{a_t} & \cdots & (\omega_{m-1}\gamma_{N-1})^{a_t} \end{array}\right], \quad (81)$$

where the elements of $GF(q^r)$ in this matrix are interpreted as column vectors of length r over $GF(q)$. We denote the $\langle\langle Nm, k, m \rangle\rangle$ code over $GF(q)$ defined by this parity check matrix by $C(\Omega, \Gamma, a_1, \ldots, a_t)$.

Let C_ν be the cyclic code of length ν over $GF(q)$ with zeros

$$1, \eta, \eta^{a_1}, \ldots, \eta^{a_t}, \eta^e, \eta^{2e}, \ldots, \eta^{\nu-e}, \quad (82)$$

where η is a primitive νth root of unity and let $\tilde{C}_\nu$ be the code over $GF(q)$ obtained by shortening C_ν in the positions

$$\{\eta^i \mid i = 0, 1, \ldots, \nu - 1\} \setminus A. \quad (83)$$

We have the following theorem.

Theorem 32. The code $C = C(\Omega, \Gamma, a_1, \ldots, a_t)$ is an $\langle\langle Nm, k, m \rangle\rangle$ SDEC code over $GF(q)$ with symbol distance at least 3 and dimension k satisfying

$$k \geq mN - m - degree(P(1, a_1, \ldots, a_t; x)),$$

where $P(1, a_1, \ldots, a_t; x)$ denotes the least common multiple of the minimal polynomials of $\eta, \eta^{a_1}, \ldots, \eta^{a_t}$ over $GF(q)$. The code C is

equivalent to a subcode of $\tilde{C}_\nu$ and if there are integers $w_1, \ldots, w_s$ such that

$$\tilde{\Omega} := \{\sum_{i=1}^{s} \omega_0^{w_i e}, \ldots, \sum_{i=1}^{s} \omega_{m-1}^{w_i e}\}$$

is a set of m linearly independent elements of $GF(q^r)$ over $GF(q)$, then C is equivalent to $\tilde{C}_\nu$.

Proof. Assume that the code C has symbol distance 2. Then, there are $0 < i_1 < i_2 \leq n$ and $c_{i_1 0}, \ldots, c_{i_1(m-1)}, c_{i_2 0}, \ldots c_{i_2(m-1)}$ in $GF(q)$ such that

$$c_{i_1 j} + c_{i_2 j} = 0, \quad j = 0, 1, \ldots, m-1, \tag{84}$$

$$\sum_{j=0}^{m-1} c_{i_1 j} \omega_j \gamma_{i_1} + \sum_{j=0}^{m-1} c_{i_2 j} \omega_j \gamma_{i_2} = 0. \tag{85}$$

Equations (84) and (85) imply that $\gamma_{i_1} = \gamma_{i_2}$, a contradiction. Consequently, C has symbol distance at least 3. The statement on the dimension of C is obvious. Also, it should be clear to the reader that the remaining assertions of Theorem 32 can be proved analogously to Theorem 30.

□

Example 22. Let α be a primitive element of $GF(2^8)$ that satisfies $\alpha^8 = \alpha^4 + \alpha^3 + \alpha^2 + 1$ and let $\Gamma = \{1, \alpha^8, \alpha^{16}, \ldots, \alpha^{8(N-1)}\}$, where $4 \leq N \leq 31$. The set $\Omega = \langle 1, \alpha, \alpha^2, \ldots, \alpha^7 \rangle$ is a polynomial basis of $GF(2^8)$ over $GF(2)$ and the set $\Omega\Gamma$ has cardinality $8N$. By Theorem 32 ($r = m = 8$, $e = \nu = 255$, $t = 1$) and Theorem 14 the code $C(\Omega, \Gamma, 3)$ is a binary $\langle\langle 8N, k, 8\rangle\rangle$ SDEC code with $k \geq 8(N-3)$, symbol distance at least 3 and minimum distance profile at least (6,3,1) (the binary cyclic code of length 255 and zeros $1, \alpha, \alpha^3$ has minimum distance 6). If N is 5 we can do better, namely let Ω be any polynomial or normal basis of $GF(2^8)$ over $GF(2)$ that consists only of 85th roots of unity of $GF(2^8)$ and let Γ be the set of the 5th roots of unity in $GF(2^8)$. It follows from Lemma 13 that the set $\Gamma\Omega$ has cardinality 40. By Theorem 32 ($m = r = 8$, $e = 5$, $\nu = 85$, $t = 1$) $C(\Omega, \Gamma, 3)$ is a binary $\langle\langle 40, k, 8\rangle\rangle$ SDEC code C of dimension at least 16 and the minimum distance of the code is at least the minimum distance δ

of the binary cyclic code of length 85 and zeros $1, \eta, \eta^3, \eta^5, \eta^{15}$, where η is a primitive 85th root of unity. With [25] we see that $\delta = 10$, hence C has minimum distance profile at least (10,5,1). If we take $\Omega = \mathcal{N}(9)$, we obtain a code with minimum distance profile (10,7,3,1). If we take $\Omega = \mathcal{N}(21)$, we obtain a code with minimum distance profile (10,6,4) and symbol distance 3. The symbol distance clearly depends on the choice of the set Ω.

The codes in this subsection can be decoded in the following way. Let $C = C(\Omega, \Gamma, a_1, \ldots, a_t)$ be defined as before and let it have minimum distance profile $(d_0, d_1, \ldots)$. Suppose we have the following two decoders.

1. A decoder $\Phi : (GF(q^r))^N \to C^N \cup \{\infty\}$ for the $[N, N-2]$ code C^N over $GF(q^r)$ with parity check matrix

$$H_N = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ \gamma_0 & \gamma_1 & \cdots & \gamma_{N-1} \end{bmatrix}$$

that corrects any single symbol error. If $\underline{x} \in (GF(q^r))^N$ has symbol distance at most one to a codeword of C^N, then there is a $\underline{d} \in (GF(q^r))^N$ of symbol weight at most 1 such that $\Phi(\underline{x}) = \underline{x} + \underline{d} \in C^N$. Otherwise we have $\Phi(\underline{x}) = \infty$ (error detection).

2. A decoder $\Psi : ((GF(q))^m)^N \to \tilde{C}_\nu \cup \{\infty\}$ for the q-ary code $\tilde{C}_\nu$ defined by (82) and (83) that corrects up to t digit errors, where $2t + 1 \leq d_0$.

It is a well known fact that we can extend Ω to a basis

$$\bar{\Omega} = \{\omega_0, \ldots, \omega_{m-1}, \omega_m, \ldots, \omega_{r-1}\}$$

of $GF(q^r)$ over $GF(q)$. We define the mapping $\underline{f} : ((GF(q))^r)^N \to ((GF(q))^m)^N$ by

$$\underline{f}((x_{00}, \ldots, x_{0(r-1)}, \ldots, x_{(N-1)0}, \ldots, x_{(N-1)(r-1)})) :=$$

$$(x_{00}, \ldots, x_{0(m-1)}, \ldots, x_{(N-1)0}, \ldots, x_{(N-1)(m-1)}) \qquad (86)$$

for all $\underline{x} \in ((GF(q))^r)^N$. The decoding algorithm can be described as follows.

Let $\underline{x}$ be a corrupted codeword, i.e. $\underline{x} = \underline{c} + \underline{e}$ for some $\underline{c} \in C$ and $\underline{e} \in ((GF(q))^m)^N$.

Step 1. Form the vector

$$\underline{\xi} = (\sum_{j=0}^{m-1} x_{0j}\omega_j, \ldots, \sum_{j=0}^{m-1} x_{(N-1)j}\omega_j) \in (GF(q^r))^N$$

and let $\hat{\underline{\xi}} = \Phi(\underline{\xi})$.

Step 2. If $\hat{\underline{\xi}} = \infty$, then define $\hat{\underline{c}} = \Psi(\underline{x})$ and stop. Otherwise, let $\hat{\underline{x}}$ be defined by

$$\hat{\underline{x}} = (\hat{x}_{00}, \ldots, \hat{x}_{0(r-1)}, \ldots, \hat{x}_{(N-1)0}, \ldots, \hat{x}_{(N-1)(r-1)}) = \hat{\underline{\xi}}(\bar{\Omega}).$$

If $\hat{x}_{ij} = 0$ for all $i = 0, 1, \ldots, N-1$, $j = m, \ldots, r-1$ and $\underline{f}(\hat{\underline{x}})H^T = 0$, then define $\hat{\underline{c}} := \underline{f}(\hat{\underline{x}})$. Otherwise, define $\hat{\underline{c}} := \Psi(\underline{x})$.

We claim that if $\underline{e}$ is a single symbol error pattern or a digit error pattern of weight at most t, then the algorithm above yields $\underline{c} = \hat{\underline{c}}$. This can be seen as follows.

1. Let $\underline{e} \in ((GF(q)^m)^N$ have symbol weight 1. Then, it is obvious that $\underline{c} = \underline{f}((\Phi(\underline{\xi}))(\bar{\Omega}))$ and thus $\underline{c} = \hat{\underline{c}}$.

2. Let $\underline{e} \in ((GF(q)^m)^N$ have digit weight larger than 1 and at most t and symbol weight at least 2. Clearly, if $\Phi(\underline{\xi})$ is equal to ∞, then $\Psi(\underline{x}) = \underline{c}$. So let $\Phi(\underline{\xi}) \neq \infty$ and suppose $\underline{f}(\hat{\underline{x}}) = \underline{c} + \underline{e} + \underline{d}$, where $\underline{d}$ is a word of symbol weight at most 1. Then, $\underline{e} + \underline{d}$ is a nonzero word with weight profile $(w_0, w_1, \ldots, w_{N-1})$ where $w_1 \leq t$. We have assumed that $2t + 1 \leq d_0$ and we have by Theorem 14 that $d_0 \leq 2d_1$. Consequently, $d_1 > t$ and $\underline{e} + \underline{d}$ is not a codeword of C. It follows that $\underline{f}(\hat{\underline{x}})H^T \neq 0$, and step 2 yields $\underline{c} = \hat{\underline{c}}$.

It is easy to see that a Reed-Solomon decoder over $GF(q^r)$ can be modified to obtain the decoder Φ. Analogously, a decoder for the cyclic code C_ν over $GF(q)$ can be modified to obtain the decoder Ψ.

F. Extending SDEC codes

There are two natural ways of extending SDEC codes.

Definition. Let C be an $\langle\langle Nm, k, m\rangle\rangle$ SDEC code over $GF(q)$. The *symbol extended code* of C is the $\langle\langle (N+1)m, k, m\rangle\rangle$ SDEC code $\bar{C}^S$ defined by

$$\bar{C}^S := \{(\underline{c}_0, \ldots, \underline{c}_{N-1}, \underline{c}_\infty) \ : \ (\underline{c}_0, \ldots, \underline{c}_{N-1}) \in C, \ \underline{c}_\infty = -\sum_{i=0}^{N-1} \underline{c}_i\}.$$

The *digit extended code* of C is the $\langle\langle N(m+1), k, m+1\rangle\rangle$ SDEC code $\bar{C}^D$ defined by

$$\bar{C}^D := \{(c_{00}, \ldots, c_{0(m-1)}, c_{0\infty}; \ldots; c_{(N-1)0}, \ldots, c_{(N-1)(m-1)}, c_{(N-1)\infty}) \ :$$

$$\underline{c} \in C, \ c_{i\infty} = -\sum_{j=0}^{m-1} c_{ij}, \ i = 0, 1, \ldots, N-1\}.$$

A special type of symbol extended codes was considered in Subsections B and E. In this section we will consider digit extended codes. The following lemma is obvious.

Lemma 34. Let C be an SDEC code with symbol distance s. Then, the digit extended code $\bar{C}^D$ also has symbol distance s and its minimum distance profile $(d_0, d_1, \ldots, d_{s-1})$ satisfies

$$d_i \geq 2(s-i), \ \ i = 0, 1, \ldots, s-1.$$

For example, if we use shortened and extended Reed-Solomon codes we will often get good SDEC codes by digit extension.

Example 23. Let $N = 2^m$, let C be the $[N+1, N-1]$ doubly extended Reed-Solomon code over $GF(2^m)$ and let B be an arbitrary basis of $GF(2^m)$ over $GF(2)$. This code has symbol distance 3, and by digit extension and symbol shortening $C(B)$ we obtain $\langle\langle P(m+1), (P-2)m, m+1\rangle\rangle$ binary SDEC codes $(P = 3, \ldots, N+1)$ with minimum distance profile at least (6,4,2). If $m = 3$, then for $P = 3, 4$ the codes are optimal and for

$P = 7,8,9$ the first component is the best known [30]. If C is the $[N+2, N-1]$ triply extended Reed-Solomon code over $GF(2^m)$ with symbol distance 4 (see [16,Ch.11,Sec.5]), then by digit extension and symbol shortening we obtain $\langle\langle P(m+1), (P-3)m, m+1\rangle\rangle$ codes ($P = 4, \ldots, N+2$) with minimum distance profile at least (8,6,4,2). If $m = 3$, then for $P = 4,5,6,7$ the codes are optimal, for $P = 8$ all components are the best known and for $P = 9,10$ the first component is the best known. If $m = 4$, then for $P = 7,10$ the first component is the best known. Many more good codes can be constructed in this way.

In some cases we obtain more than the bound of the lemma.

Example 24. [7] Let α be a primitive element of $GF(2^8)$ that satisfies $\alpha^8 = \alpha^4 + \alpha^3 + \alpha^2 + 1$ and let $\gamma = \alpha^{85}$. Then, γ is a primitive element of $GF(2^2)$. Let C be the [3,2] code over $GF(2^8)$ with symbol distance 2 and with generator matrix

$$G = \begin{bmatrix} 1 & 1 & 1 \\ 1 & \gamma & \gamma^2 \end{bmatrix}.$$

Let B be an arbitrary normal basis of $GF(2^8)$ over $GF(2)$ and consider the digit extended code $\bar{C}^D(B)$ of $C(B)$. We will show that the minimum distance profile of this code is (6,2).

1. Clearly a codeword of symbol weight 3 in $\bar{C}^D(B)$ has digit weight at least 6.

2. Let $\underline{c}$ be a codeword of $\bar{C}^D(B)$ of symbol weight two and suppose it has digit weight four. Then there are elements $\beta_1, \beta_2, \beta_3, \beta_4$ of B such that

$$(\beta_1 + \beta_2)\gamma = (\beta_3 + \beta_4)\gamma^2.$$

In Example 4 it was shown that if $B = \mathcal{N}(11)$, then this is impossible and similarly we can show that it is impossible if $B = \mathcal{N}(43)$. Consequently, with respect to these two bases the code has minimum distance profile at least (6,2), and with the help of [10] we see that it is (6,2) and the code is optimal. By Corollary 9 this is the case for all normal bases of $GF(2^8)$ over $GF(2)$. The codes are {(0,1)}-correcting and {(1,0),(0,4)}-detecting.

G. Tables of binary SDEC codes

In this subsection we give tables of binary $\langle N, K, m\rangle$ SDEC codes obtained by taking binary images of codes over $GF(2^m)$ that have a maximal minimum symbol distance (with two exceptions). The codes denoted by an asterix (*) are optimal. The underlined components of the minimum distance profile are optimal. We give tables for $m = 2, 3, 4, 6, 8$ and a table of sporadic examples.

G.1. Binary $\langle N, K, 2\rangle$ SDEC codes for $1 \leq K < N \leq 12$

In Table IV a number of $\langle N, K, 2\rangle$ SDEC codes for $1 \leq K < N \leq 12$ are given. For a table of optimal linear codes over $GF(4)$ we refer to [28]. Below we provide the constructions of the codes whose parameters are in Table IV.

Let ω be a primitive element of $GF(4)$ and let Ω be the basis $\langle \omega, \omega^2\rangle$ of $GF(4)$ over $GF(2)$. Note that the binary image of a linear code over $GF(4)$ is independent of the choice of the basis of $GF(4)$ over $GF(2)$ (Theorem 3).

A. The $\langle N, N-1, 2\rangle$ codes are the binary images with respect to an arbitrary basis of an $[N, N-1]$ code over $GF(4)$ with symbol distance two.

B. The $\langle N, N-2, 2\rangle$ codes for $N \geq 8$ are obtained by expurgating the codes from construction A in one symbol.

C. The $\langle N, N-2, 2\rangle$ codes for $N = 6, 7$ are obtained by applying the construction of Subsection D to shortened binary [15,11] Hamming codes.

D. In Example 16 we have constructed a $\langle 12, 6, 2\rangle$ code with minimum distance profile (8,6,4,3,2,1). By shortening this code in a number of symbols we obtain $\langle N, N-6, 2\rangle$ codes for $N = 8, 9, 10, 11$ with the same minimum distance profile. By puncturing the code in one symbol we obtain a $\langle 11, 6, 2\rangle$ code with minimum distance profile (6,4,3,2,1). By shortening this code we obtain $\langle N, N-5, 2\rangle$ codes for $N = 7, 8, 9, 10$ with the same minimum distance profile.

	N	K	min.dist.profile		N	K	min.dist.profile
A	2	1	2 1*	G	10	1	13 11 9 7 6 5 4 3 2 1*
G	3	1	4 2 1*	H	10	2	9 7 5 4 3 2 1
A	3	2	2 1*	N	10	3	$\underline{8}$ 6 4 3 2 1
G	4	1	5 3 2 1*	D	10	4	$\underline{8}$ 6 4 3 2 1
H	4	2	4 2 1*	D	10	5	6 4 3 2 1*
A	4	3	2 1*	J	10	6	4 3 2 1*
G	5	1	6 4 3 2 1*	E	10	7	4 2 1*
H	5	2	4 3 2 1*	B	10	8	2 1*
I	5	3	3 2 1*	A	10	9	2 1*
A	5	4	2 1*	G	11	1	14 12 10 8 7 6 5 4 3 2 1*
G	6	1	8 6 4 3 2 1*	H	11	2	10 8 6 5 4 3 2 1
F	6	2	5 3 2 1	M	11	3	8 6 5 4 3 2 1
I	6	3	4 3 2 1*	N	11	4	$\underline{8}$ 6 4 3 2 1
C	6	4	$\underline{3}$ 1	D	11	5	8 6 4 3 2 1*
A	6	5	2 1*	D	11	6	6 4 3 2 1*
G	7	1	9 7 5 4 3 2 1*	J	11	7	4 3 2 1*
D	7	2	6 4 3 2 1	E	11	8	4 2 1*
F	7	3	$\underline{5}$ 3 2 1	B	11	9	2 1*
E	7	4	$\underline{4}$ 2 1	A	11	10	2 1*
C	7	5	$\underline{3}$ 1	G	12	1	16 14 12 10 8 7 6 5 4 3 2 1*
A	7	6	2 1*	H	12	2	$\underline{12}$ 10 8 6 5 4 3 2 1
G	8	1	10 8 6 5 4 3 2 1*	L	12	3	8 7 6 5 4 3 2 1
D	8	2	$\underline{8}$ 6 4 3 2 1	M	12	4	8 6 5 4 3 2 1
D	8	3	$\underline{6}$ 4 3 2 1	N	12	5	$\underline{8}$ 6 4 3 2 1
F	8	4	$\underline{5}$ 3 2 1	D	12	6	8 6 4 3 2 1*
E	8	5	$\underline{4}$ 2 1	K	12	7	4 3 2 1
B	8	6	2 1*	J	12	8	4 3 2 1*
A	8	7	2 1*	E	12	9	4 2 1*
G	9	1	12 10 8 6 5 4 3 2 1*	B	12	10	2 1 *
H	9	2	$\underline{8}$ 6 5 4 3 2 1	A	12	11	2 1*
D	9	3	$\underline{8}$ $\underline{6}$ 4 3 2 1				
D	9	4	$\underline{6}$ 4 3 2 1				
J	9	5	4 3 2 1*				
E	9	6	4 2 1*				
B	9	7	2 1*				
A	9	8	2 1*				

Table IV: *Binary* $\langle N, K, 2 \rangle$ *SDEC codes for* $1 \leq K < N \leq 12$.

E. For the [15,12] BCH code C_1 over $GF(4)$ with zeros 1 and α, where α is a primitive element of $GF(16)$, the binary $\langle 15, 12, 2\rangle$ SDEC code $C_1(\Omega)$ has minimum distance profile (4,2,1). By shortening we obtain $\langle N, N-3, 2\rangle$ codes for $N = 7, \ldots, 12$ with $\underline{mdp} = (4, 2, 1)$.

F. Let C_2 be the linear code over $GF(4)$ with generator matrix

$$G_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & \omega & \omega^2 & \omega^2 & 0 \\ 0 & 1 & 0 & 0 & \omega^2 & 0 & \omega^2 & \omega \\ 0 & 0 & 1 & 0 & \omega^2 & \omega & 0 & \omega^2 \\ 0 & 0 & 0 & 1 & 0 & \omega^2 & \omega & \omega^2 \end{bmatrix}.$$

The binary image of this code with respect to the basis Ω is an $\langle 8, 4, 2\rangle$ SDEC code with minimum distance profile (5,3,2,1). By shortening we obtain $\langle 7, 3, 2\rangle$ and $\langle 6, 2, 2\rangle$ codes with the same minimum distance profile.

G. Define G_3 to be the matrix $[1\ \omega\ \omega^2]$. The $\langle N, 1, 2\rangle$ codes for $N = 3, \ldots, 12$ in Table IV are the binary images with respect to Ω of the codes over $GF(4)$ with generator matrices G_3, $[G_3\ 1]$, $[G_3\ 1\ 1]$, $[G_3\ G_3]$, ... , $[G_3\ G_3\ G_3\ G_3]$ respectively.

H. Define G_4 to be the matrix $G_4 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & \omega & \omega^2 \end{bmatrix}$. The $\langle N, 2, 2\rangle$ codes for $N = 4, \ldots, 12$ in Table IV are the binary images with respect to Ω of the codes over $GF(4)$ with generator matrices

$$\begin{bmatrix} G_4 & \begin{matrix} 1 \\ 0 \end{matrix} \end{bmatrix}, \begin{bmatrix} G_4 & \begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix} \end{bmatrix}, \begin{bmatrix} G_4 & G_4 & \begin{matrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{matrix} \end{bmatrix},$$

$$\begin{bmatrix} G_4 & G_4 & \begin{matrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{matrix} \end{bmatrix}, \begin{bmatrix} G_4 & G_4 & \begin{matrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & \omega \end{matrix} \end{bmatrix},$$

$$\begin{bmatrix} G_4 & G_4 & G_4 & \begin{matrix} 1 & 1 & 1 \\ 0 & 0 & 0 \end{matrix} \end{bmatrix}$$

respectively.

I. The $\langle 6, 3, 2\rangle$ code which is the binary image with respect to Ω of the code over $GF(4)$ with generator matrix

$$G_5 = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & \omega & \omega^2 & 0 & 1 & 0 \\ 1 & \omega^2 & \omega & 0 & 0 & 1 \end{bmatrix}$$

has minimum distance profile (4,3,2,1). By puncturing this code we obtain a $\langle 5,3,2\rangle$ code with minimum distance profile (3,2,1).

J. The binary $\langle 15,11,2\rangle$ code which is the binary image with respect to Ω of the [15,11] cyclic code over $GF(4)$ with zeros $1,\alpha,\alpha^4,\alpha^5$ has minimum distance profile (4,3,2,1). By shortening this code we obtain binary $\langle N,N-4,2\rangle$ SDEC codes for $N=9,\ldots,12$ with minimum distance profile (4,3,2,1).

K. The $\langle 12,7,2\rangle$ code is the binary image of a [12,7,4] code over $GF(4)$ [28].

L. The $\langle 12,3,2\rangle$ code is the binary image of a [12,3,8] code over $GF(4)$ [28].

M. Let C_6 be the [15,8] BCH code over $GF(4)$ with zeros $\alpha,\alpha^2,\alpha^3,\alpha^5$ where α is a primitive element of $GF(16)$. Let $\bar{C}_6$ be its symbol extension. The code $\bar{C}_6$ has symbol distance 7 and its binary image with respect to Ω has minimum distance profile (8,6,5,4,3,2,1). By shortening we obtain $\langle 12,4,2\rangle$ and $\langle 11,3,2\rangle$ codes with the same minimum distance profile.

N. By expurgating SDEC codes we obtain SDEC codes with at least the same minimum distance profile.

G.2. Binary $\langle N,K,3\rangle$ SDEC codes for $1\leq K<N\leq 8$

In Table V a number of binary $\langle N,K,3\rangle$ SDEC codes are given. Let α be a primitive element of $GF(8)$ that satisfies $\alpha^3+\alpha+1=0$.

A. The $\langle N,N-1,3\rangle$ codes for $N\geq 3$ are trivial, as before in the case of 2-bit symbols. The $\langle 2,1,3\rangle$ code is the binary image with respect to the basis $\langle 1,\alpha,\alpha^2\rangle$ of the code over $GF(8)$ with generator matrix $[1\ \alpha^3]$.

B. The $\langle 7,4,3\rangle$ code with minimum distance profile (5,3,2,1) is the binary image with respect to the normal basis $\mathcal{N}(3):=\langle\alpha^3,\alpha^6,\alpha^5\rangle$ of the [7,4] Reed-Solomon code over $GF(8)$ with zeros α,α^2 and α^3. By shortening we obtain $\langle N,N-3,3\rangle$ codes for $N=5,6$ with the same minimum distance profile.

C. The $\langle 8,4,3\rangle$ code is from Example 5. By shortening this code we obtain $\langle N,N-4,3\rangle$ codes for $N=5,6,7$ with minimum dis-

	N	K	min.dist.profile		N	K	min.dist.profile
A	2	1	3 1*	F	7	1	12 9 7 5 3 2 1
H	3	1	4 2 1*	G	7	2	8 5 4 3 2 1
A	3	2	2 1*	C	7	3	8 5 3 2 1
D	4	1	6 4 2 1*	B	7	4	5 3 2 1
H	4	2	4 2 1*	H	7	5	4 2 1*
A	4	3	2 1*	A	7	6	2 1*
C	5	1	8 5 3 2 1	F	8	1	13 10 8 6 4 3 2 1
B	5	2	5 3 2 1	I	8	2	9 6 5 4 3 2 1
H	5	3	4 2 1*	G	8	3	8 5 4 3 2 1
A	5	4	2 1*	C	8	4	8 5 3 2 1
F	6	1	9 7 5 3 2 1	E	8	5	4 3 2 1
C	6	2	8 5 3 2 1	H	8	6	4 2 1*
B	6	3	5 3 2 1	A	8	7	2 1*
H	6	4	4 2 1*				
A	6	5	2 1*				

Table V: *Binary* $\langle N, K, 3\rangle$ *SDEC codes for* $1 \leq K < N \leq 8$.

tance profile (8,5,3,2,1).

D. The $\langle 4, 1, 3\rangle$ code is the repetition of the $\langle 2, 1, 3\rangle$ code.

E. The $\langle 8, 5, 3\rangle$ code is obtained by taking the binary image with respect to an arbitrary basis of the extended [8,5] Reed-Solomon code.

F. The $\langle 7, 1, 3\rangle$ code is obtained by taking the binary image with respect to an arbitrary basis of the [7,1] Reed-Solomon code over $GF(8)$. By puncturing we obtain the $\langle 6, 1, 3\rangle$ code. By adding the 3 by 3 identity matrix to the generator matrix of the $\langle 7, 1, 3\rangle$ code we obtain the $\langle 8, 1, 3\rangle$ code.

G. The $\langle 8, 3, 3\rangle$ code is the binary image with respect to the normal basis $N(3)$ of the [8,3] extended Reed-Solomon code over $GF(8)$. By shortening this code we obtain the $\langle 7, 2, 3\rangle$ code.

H. The $\langle 8, 6, 3\rangle$ code is the binary image with respect to an arbitrary basis of the [8,6] extended Reed-Solomon code. By shortening we obtain the $\langle N, N-2, 3\rangle$ codes for $N = 3, \ldots, 7$.

I. The $\langle 8, 2, 3\rangle$ code is the binary image with respect to the basis

$\mathcal{N}(3)$ of the [8,2] code over $GF(8)$ with generator matrix

$$\begin{bmatrix} 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 & 0 \\ 1 & \alpha^2 & \alpha^4 & \alpha^6 & \alpha & \alpha^3 & \alpha^5 & 1 \end{bmatrix}.$$

G.3. Binary $\langle N, K, 4\rangle$ SDEC codes for $1 \leq K < N \leq 8$

In Table VI a number of binary $\langle N, K, 4\rangle$ codes are given. Let α be a primitive element of $GF(16)$ that satisfies $\alpha^4 + \alpha + 1 = 0$.

	N	K	min.dist.profile		N	K	min.dist.profile
B	2	1	4 1*	D	7	1	14 11 8 5 3 2 1
C	3	1	6 3 1*	G	7	2	9 6 5 3 2 1
A	3	2	3 1*	G	7	3	7 4 3 2 1
B	4	1	8 5 2 1	I	7	4	5 3 2 1
C	4	2	5 3 1*	E	7	5	4 2 1*
A	4	3	2 1*	A	7	6	2 1*
D	5	1	10 7 4 2 1	B	8	1	16 13 10 7 4 3 2 1
F	5	2	8 4 2 1	G	8	2	12 9 6 5 3 2 1
E	5	3	4 2 1*	L	8	3	8 6 4 3 2 1
A	5	4	2 1*	K	8	4	6 4 3 2 1
D	6	1	12 9 6 4 2 1	J			8 5 3 1
G	6	2	8 6 3 2 1	I	8	5	5 3 2 1
H	6	3	6 4 2 1	E	8	6	4 2 1*
H			8 4 2*	A	8	7	2 1*
E	6	4	4 2 1*				
A	6	5	2 1*				

Table VI: *Binary* $\langle N, K, 4\rangle$ *SDEC codes for* $1 \leq K < N \leq 8$.

A. The $\langle N, N-1, 4\rangle$ codes for $N \geq 4$ are trivial, as before in the 2-and 3-bit cases. The $\langle 3, 2, 4\rangle$ code is the dual of the $\langle 3, 1, 4\rangle$ code constructed in [6].

B. If G_1 is a systematic generator matrix of the binary [8,4] Hamming code, then the matrices G_1, $[G_1\ G_1]$ and $[G_1\ G_1\ G_1\ G_1]$ are generator matrices for the $\langle N, 1, 4\rangle$ codes for $N = 2, 4$ and 8, respectively.

C. The $\langle 3, 1, 4\rangle$ code is from [6]. The $\langle 4, 2, 4\rangle$ code is from [12,13,14].

D. If G_2 and G_3 are the generator matrices of the $\langle 2,1,4\rangle$ and $\langle 3,1,4\rangle$ codes respectively, then $[G_2\ G_3]$, $[G_3\ G_3]$ and $[G_2\ G_3\ G_3]$ are generator matrices for the $\langle 5,1,4\rangle$, $\langle 6,1,4\rangle$ and $\langle 7,1,4\rangle$ codes, respectively.

E. The $\langle N, N-2, 4\rangle$ codes for $N = 5,6,7,8$ are obtained by shortening the binary image with respect to an arbitrary basis of $GF(16)$ of an extended [16,14] Reed-Solomon code over $GF(16)$.

F. The $\langle 5,2,4\rangle$ code is from Example 8.

G. The $\langle 7,3,4\rangle$ code is the binary image with respect to the self-complementary basis $S := \langle \alpha^3, \alpha^7, \alpha^{12}, \alpha^{13}\rangle$ of the [7,3] code over $GF(16)$ with generator matrix

$$G_4 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \alpha^3 & \alpha^6 & \alpha^9 & \alpha^{12} & 1 & 0 \\ 1 & \alpha^6 & \alpha^{12} & \alpha^3 & \alpha^9 & 0 & 0 \end{bmatrix}.$$

Shortening this code in the last symbol gives the $\langle 6,2,4\rangle$ code with minimum distance profile (8,6,3,2,1).

The $\langle 8,2,4\rangle$ code is the binary image with respect to the basis S of the [8,2] code over $GF(16)$ with generator matrix

$$G_5 = \begin{bmatrix} \alpha^3 & \alpha^6 & \alpha^{12} & \alpha^9 & \alpha^7 & \alpha^{14} & \alpha^{13} & \alpha^{11} \\ \alpha^6 & \alpha^{12} & \alpha^9 & \alpha^3 & \alpha^{14} & \alpha^{13} & \alpha^{11} & \alpha^7 \end{bmatrix}.$$

By puncturing this code we obtain the $\langle 7,2,4\rangle$ code with minimum distance profile (9,6,5,3,2,1).

H. The first-mentioned $\langle 6,3,4\rangle$ code is the binary image with respect to the normal basis $\mathcal{N}(7) := \langle \alpha^7, \alpha^{14}, \alpha^{13}, \alpha^{11}\rangle$ of the [6,3] code over $GF(16)$ with parity check matrix

$$H_6 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ \alpha^3 & \alpha^6 & \alpha^{12} & \alpha^9 & 1 & 0 \\ \alpha^6 & \alpha^{12} & \alpha^9 & \alpha^3 & 1 & 0 \end{bmatrix}.$$

The second-mentioned $\langle 6,3,4\rangle$ code is from Example 16.

I. The $\langle 8,5,4\rangle$ code is the binary image with respect to the polynomial basis $\langle 1, \alpha, \alpha^2, \alpha^3\rangle$ of the [8,5] code over $GF(16)$ with parity

check matrix

$$H_7 = \begin{bmatrix} \alpha^3 & \alpha^6 & \alpha^9 & \alpha^{12} & \alpha^7 & \alpha^{14} & \alpha^{13} & 0 \\ \alpha^6 & \alpha^{12} & \alpha^3 & \alpha^9 & \alpha^{14} & \alpha^{13} & \alpha^{11} & 0 \\ \alpha^9 & \alpha^3 & \alpha^6 & \alpha^{12} & \alpha^6 & \alpha^{12} & \alpha^9 & 1 \end{bmatrix}.$$

By shortening this code we obtain the $\langle 7,4,4\rangle$ code.

J. The second-mentioned $\langle 8,4,4\rangle$ code is obtained by applying the construction of Subsection D to the $\langle 16,8,2\rangle$ code with minimum distance profile (8,6,5,4,3,2,1) which is the binary image with respect to an arbitrary basis of $GF(4)$ over $GF(2)$ of the extended [16,8] BCH code over $GF(4)$ with zeros $\alpha, \alpha^2, \alpha^3$ and α^5.

K. The $\langle 8,4,4\rangle$ code with minimum distance profile (6,4,3,2,1) is obtained by shortening the binary image with respect to an arbitrary basis of $GF(16)$ over $GF(2)$ of an extended [16,12] Reed-Solomon code over $GF(4)$.

L. The $\langle 8,3,4\rangle$ code is the binary image with respect to an arbitrary polynomial, normal or self-complementary basis of $GF(16)$ over $GF(2)$ of the code over $GF(16)$ with parity check matrix

$$H_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \alpha^3 & \alpha^6 & \alpha^9 & \alpha^{12} & \alpha^7 & \alpha^{14} & \alpha^{13} & \alpha^{11} \\ \alpha^6 & \alpha^{12} & \alpha^3 & \alpha^9 & \alpha^{14} & \alpha^{13} & \alpha^{11} & \alpha^7 \\ \alpha^9 & \alpha^3 & \alpha^{12} & \alpha^6 & \alpha^6 & \alpha^{12} & \alpha^9 & \alpha^3 \\ \alpha^{12} & \alpha^9 & \alpha^6 & \alpha^3 & \alpha^{13} & \alpha^{11} & \alpha^7 & \alpha^{14} \end{bmatrix}.$$

G.4. Binary $\langle N,K,6\rangle$ SDEC codes for $1 \le K < N \le 7$

Table VII gives binary $\langle N,K,6\rangle$ codes for $1 \le K < N \le 7$. Define α to be a primitive element of $GF(2^6)$ that satisfies $\alpha^6+\alpha+1=0$.

A. The $\langle 4,3,6\rangle$ code is obtained by applying the construction of Subsection D to the $\langle 8,6,3\rangle$ code of Table V. By shortening we obtain $\langle 3,2,6\rangle$ and $\langle 2,1,6\rangle$ codes with the same minimum distance profile. The $\langle 7,6,6\rangle$ code is the binary image with respect to the polynomial basis $\langle 1,\alpha,\ldots,\alpha^5\rangle$ or one of the normal bases $\mathcal{N}(5)$, $\mathcal{N}(23)$ or $\mathcal{N}(31)$ of $GF(2^6)$ over $GF(2)$ of the [7,6] code over $GF(2^6)$ with parity check matrix

$$\begin{bmatrix} 1 & \gamma & \gamma^2 & \gamma^3 & \gamma^4 & \gamma^5 & \gamma^6 \end{bmatrix},$$

	N	K	min.dist.profile		N	K	min.dist.profile
A	2	1	4 1*	F	6	1	16 12 9 6 3 1
B	3	1	8 4 1*	C	6	2	12 7 5 2 1
A	3	2	4 1*	E	6	3	7 5 2 1
F	4	1	10 6 3 1	G	6	4	4 3 1
D	4	2	8 3 1	A	6	5	3 1*
A	4	3	4 1*	F	7	1	20 16 12 9 6 3 1
F	5	1	12 9 6 3 1	F	7	2	12 9 6 4 2 1
E	5	2	8 5 2 1	C	7	3	12 7 5 2 1
G	5	3	4 3 1	E	7	4	7 5 2 1
A	5	4	3 1	G	7	5	4 3 1
				A	7	6	3 1*

Table VII: *Binary* $\langle N, K, 6\rangle$ *SDEC codes for* $1 \leq K < N \leq 7$.

where $\gamma = \alpha^9$. By shortening we obtain the $\langle 6,5,6\rangle$ and $\langle 5,4,6\rangle$ codes.

B. The $\langle 3,1,6\rangle$ code is from Example 6.

C. The $\langle 7,3,6\rangle$ code is from Example 7. By shortening we obtain the $\langle 6,2,6\rangle$ code.

D. The $\langle 4,2,6\rangle$ code is obtained by applying the construction of Subsection D to the $\langle 8,4,3\rangle$ code from Table V.

E. The $\langle 7,4,6\rangle$ code C_1 is the binary image with respect to the normal basis $\mathcal{N}(23)$ of $GF(2^6)$ over $GF(2)$ of the [7,4] BCH code over $GF(2^6)$ with zeros α^9, α^{18} and α^{27}. By shortening this code in a symbol we obtain the $\langle 6,3,6\rangle$ code. By shortening C_1 in the last two symbols we obtain a $\langle 5,2,6\rangle$ code with minimum distance profile (8,5,2,1).

F. The $\langle 7,2,6\rangle$ code is the binary image with respect to the basis $\mathcal{N}(23)$ of the code over $GF(2^6)$ with generator matrix

$$G_2 = \begin{bmatrix} 1 & \gamma & \gamma^2 & \gamma^3 & \gamma^4 & \gamma^5 & \gamma^6 \\ 1 & \gamma^2 & \gamma^4 & \gamma^6 & \gamma & \gamma^3 & \gamma^5 \end{bmatrix},$$

where $\gamma = \alpha^9$. The code generated by the first row of G_2 with respect to $\mathcal{N}(23)$ gives the $\langle 7,1,6\rangle$ code C_3. By puncturing this code C_3 we obtain the $\langle 6,1,6\rangle$ and the $\langle 5,1,6\rangle$ codes. The $\langle 4,1,6\rangle$ code is obtained by puncturing C_3 in the third, fifth and sixth positions.

G. The $\langle 7,5,6\rangle$ code is the binary image with respect to the polynomial basis of α or one of the normal bases $\mathcal{N}(5)$, $\mathcal{N}(23)$ or $\mathcal{N}(31)$ of the [7,5] BCH code over $GF(2^6)$ with zeros 1 and α^9. By shortening we obtain the $\langle 6,4,6\rangle$ and $\langle 5,3,6\rangle$ codes.

G.5. Binary $\langle N,K,8\rangle$ SDEC codes for $1 \le K < N \le 8$

Table VIII gives binary $\langle N,K,8\rangle$ codes for $1 \le K < N \le 8$. Define α to be a primitive element of $GF(2^8)$ that satisfies $\alpha^8+\alpha^4+\alpha^3+\alpha^2+1=0$.

	N	K	min.dist.profile		N	K	min.dist.profile
A	2	1	5 1*	G	7	1	23 18 14 8 6 3 1
B	3	1	8 5 1	H	7	2	16 11 7 5 3 1
A	3	2	4 1*	F	7	3	12 7 5 3 1
E	4	1	12 7 4 1	C	7	4	8 5 3 1
B	4	2	7 4 1	B	7	5	6 3 1
A	4	3	4 1*	A	7	6	4 1*
F	5	1	16 11 7 4 1	G	8	1	28 23 18 14 8 6 3 1
D	5	2	10 7 4 1	I	8	2	20 15 11 8 5 3 1
B	5	3	6 3 1	H	8	3	16 11 7 5 3 1
A	5	4	4 1*	F	8	4	12 7 5 3 1
G	6	1	19 13 9 6 3 1	C	8	5	8 5 3 1
F	6	2	12 7 5 3 1	B	8	6	6 3 1
C	6	3	8 5 3 1	A	8	7	4 1*
B	6	4	6 3 1				
A	6	5	4 1*				

Table VIII: *Binary* $\langle N,K,8\rangle$ *SDEC codes for* $1 \le K < N \le 8$.

A. The $\langle N,N-1,8\rangle$ SDEC codes, $3 \le N \le 8$, are obtained from the $[2N,2N-2]$ shortened Reed-Solomon codes over $GF(16)$ by applying the construction of Subsection D with $r=2$. The $\langle 2,1,8\rangle$ code is obtained by puncturing the $\langle 3,1,8\rangle$ code of the table.

B. The $\langle N,N-2,8\rangle$ SDEC codes, $5 \le N \le 8$, are from Example 12. The $\langle 4,2,8\rangle$ code is from [8]. The $\langle 3,1,8\rangle$ code is from [6].

C. The $\langle N,N-3,8\rangle$ SDEC codes for $6 \le N \le 8$ are from Example 13.

D. The $\langle 5,2,8\rangle$ SDEC code is the binary image with respect to

the normal basis $\mathcal{N}(11)$ or $\mathcal{N}(47)$ of $GF(2^8)$ over $GF(2)$ of the code over $GF(2^8)$ with generator matrix

$$G_1 = \begin{bmatrix} 1 & \gamma & \gamma^2 & \gamma^3 & \gamma^4 \\ 1 & \gamma^2 & \gamma^4 & \gamma & \gamma^3 \end{bmatrix},$$

where $\gamma = \alpha^{51}$ (see also [1]).

E. The $\langle 4, 1, 8 \rangle$ SDEC code is the binary image with respect to the normal basis $\mathcal{N}(11)$ or $\mathcal{N}(47)$ of the code over $GF(2^8)$ with parity check matrix

$$H_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ \gamma & \gamma^2 & \gamma^4 & \gamma^8 \\ \gamma^2 & \gamma^4 & \gamma^8 & \gamma \end{bmatrix},$$

where $\gamma = \alpha^{51}$ or α^{119}.

F. The $\langle N, N-4, 8 \rangle$ SDEC codes for $6 \leq N \leq 8$ are from Example 14. The $\langle 5, 1, 8 \rangle$ code is the binary image with respect to one of the normal bases $\mathcal{N}(11)$, $\mathcal{N}(43)$, $\mathcal{N}(47)$ or $\mathcal{N}(95)$ of the code over $GF(2^8)$ with generator matrix

$$G_3 = \begin{bmatrix} 1 & \gamma & \gamma^2 & \gamma^3 & \gamma^4 \end{bmatrix},$$

where $\gamma = \alpha^{51}$.

G. The $\langle 8, 1, 8 \rangle$ code is the binary image with respect to one of the normal bases $\mathcal{N}(9)$, $\mathcal{N}(15)$, $\mathcal{N}(29)$ or $\mathcal{N}(91)$ of the code over $GF(2^8)$ with generator matrix

$$G_4 = \begin{bmatrix} \eta^3 & \eta^6 & \eta^{12} & \eta^9 & \eta^7 & \eta^{14} & \eta^{13} & \eta^{11} \end{bmatrix},$$

where $\eta = \alpha^{17}$. By puncturing this code in one symbol we obtain the $\langle 7, 1, 8 \rangle$ code of the table. The $\langle 6, 1, 8 \rangle$ code is the binary image with respect to the basis $\mathcal{N}(39)$ or $\mathcal{N}(55)$ of the code with generator matrix

$$G_5 = \begin{bmatrix} \eta^3 & \eta^6 & \eta^{12} & \eta^9 & \eta^7 & \eta^{14} \end{bmatrix},$$

where $\eta = \alpha^{17}$.

H. The $\langle 8, 3, 8 \rangle$ code is obtained by applying the construction of

Subsection D to the $\langle 16, 6, 4\rangle$ code of Table IX. By shortening we obtain the $\langle 7, 2, 8\rangle$ code.

I. The $\langle 8, 2, 8\rangle$ code is obtained by taking the binary image with respect to one of the normal bases $\mathcal{N}(9)$, $\mathcal{N}(43)$, $\mathcal{N}(91)$ or $\mathcal{N}(95)$ of the code over $GF(2^8)$ with generator matrix

$$G_6 = \begin{bmatrix} \eta^3 & \eta^6 & \eta^{12} & \eta^9 & \eta^7 & \eta^{14} & \eta^{13} & \eta^{11} \\ \eta^6 & \eta^{12} & \eta^9 & \eta^3 & \eta^{14} & \eta^{13} & \eta^{11} & \eta^7 \end{bmatrix},$$

where $\eta = \alpha^{17}$.

G.6. Sporadic examples of SDEC codes

Table IX gives the parameters of some sporadic examples of SDEC codes.

	m	N	K	min.dist.profile
A	4	16	6	16 14 11 9 7 6 5 4 3 2 1
B	8	$9 \leq N \leq 16$	$N-2$	4 3 1
C	8	$9 \leq N \leq 31$	$N-3$	6 3 1
D	9	4	2	8 4 1
E	10	5	2	12 8 4 1
F	10	$5 \leq N \leq 16$	$N-2$	6 3 1
G	16	3	1	13 7 1
H	16	4	2	12 5 1

Table IX: *Sporadic examples of SDEC codes.*

A. The $\langle 16, 6, 4\rangle$ code is the binary image with respect to the normal basis $\mathcal{N}(3) = \langle \alpha^3, \alpha^6, \alpha^{12}, \alpha^9\rangle$ of $GF(16)$ over $GF(2)$ (α being a primitive element of $GF(16)$ that satisfies $\alpha^4 + \alpha + 1 = 0$) of the [16,6] extended Reed-Solomon code over $GF(16)$.

B. These codes are obtained by shortening the code of Example 2.

C. See Example 22.

D. Let $\langle \gamma_1, \gamma_2, \gamma_3\rangle$ be a basis of $GF(8)$ over $GF(2)$. Let C_1 be the code over $GF(2^9)$ with parity check matrix

$$H_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ \gamma_1 & \gamma_2 & \gamma_3 & 0 \end{bmatrix}.$$

Let α be a primitive element of $GF(2^9)$ satisfying $\alpha^9 + \alpha^4 + 1 = 0$ and let $\mathcal{N}(5)$ denote the normal basis of $GF(2^9)$ over $GF(2)$ containing α^5. The code $C(\mathcal{N}(5))$ has minimum distance profile (8,4,1) (see [1]).

E. The $\langle 5, 2, 10\rangle$ code is the binary image with respect to the normal basis $\mathcal{N}(63) := \langle \alpha^{63}, \ldots\rangle$ of $GF(2^{10})$ over $GF(2)$ of the code with generator matrix

$$G_2 = \begin{bmatrix} \eta & \eta^2 & \eta^4 & \eta^8 & \eta^{16} \\ \eta^2 & \eta^4 & \eta^8 & \eta^{16} & \eta \end{bmatrix},$$

where α is a primitive element of $GF(2^{10})$ satsfying $\alpha^{10}+\alpha^3+1 = 0$ and $\eta = \alpha^{165}$.

F. See Example 15.

G. Let α be a primitive element of $GF(2^{16})$ satisfying $\alpha^{16} = \alpha^{12} + \alpha^3 + \alpha + 1$, let $\gamma = \alpha^{13107}$ (so γ is a primitive 5th root of unity) and let $\mathcal{N}(299)$ denote the normal basis of $GF(2^{16})$ over $GF(2)$ containing α^{299}. If C_3 is the code over $GF(2^{16})$ with generator matrix

$$G_3 = \begin{bmatrix} 1 & \gamma & \gamma^2 \end{bmatrix},$$

then the code $C_3(\mathcal{N}(299))$ is a $\langle 3, 1, 16\rangle$ binary SDEC code with minimum distance profile (13,7,1) (see [1]).

H. See Example 14.

References

[1] J. P. Boly, *On Combined Symbol and Digit Error Control Codes*, Master's thesis, Eindhoven University of Technology, Department of Mathematics and Computing Science, November 1986.

[2] C. L. Chen, "Error correcting codes with byte error detection capability", *IEEE Trans. Comput.*, vol. C-32, no. 7, pp. 615-621, July 1983.

[3] H. Davenport, "Bases for finite fields", *J. London Math. Soc.*, vol. 43, pp. 21-39, 1968.

[4] L. A. Dunning, "SEC-BED-DED codes for error control in byte organized memory systems", *IEEE Trans. Comput.*, vol. C-34, no 6, pp. 557-562, June 1985.

[5] L. A. Dunning and M. R. Varanasi, "Code constructions for error control in byte organized memory systems", *IEEE Trans. Comput.*, vol. C-32, no 6, pp. 535-542, June 1983.

[6] W. J. van Gils, "A triple modular redundancy technique providing multiple-bit error protection without using extra redundancy", *IEEE Trans. Comput.*, vol. C-35, no 7, pp. 623-631, July 1986.

[7] W. J. van Gils, "An error-control coding system for storage of 16-bit words in memory arrays composed of three 9-bit wide units", *Philips Journal of Research*, vol. 41, pp. 391-399, August 1986.

[8] W. J. van Gils and J. P. Boly, "On combined symbol-and-bit error-control $[4,2]$ codes over $\{0,1\}^8$ to be used in the $(4,2)$-concept fault-tolerant computer", *IEEE Trans. Inform. Theory*, vol. IT-33, no.6, November 1987.

[9] D. Y. Goldberg, "Reconstructing the ternary Golay code", *J. Combin. Theory Ser. A*, vol. 42, pp.296-299, 1986.

[10] H. J. Helgert and R. D. Stinaff, "Minimum-distance bounds for binary linear codes", *IEEE Trans. Inform. Theory*, vol. IT-19, no 3, pp. 344-356, May 1973.

[11] S. Kaneda, "A class of odd weight column SEC-DED-SbED codes for memory systems applications", *IEEE Trans. Comput.*, vol. C-33, no 8, pp. 737-739, August 1984.

[12] T. Krol, "The '(4,2)-concept' fault-tolerant computer", *Dig. 12th Int. Symp. Fault Tolerant Computing*, Santa Monica, CA, June 1982.

[13] T. Krol, "(N,K) Concept fault-tolerance", *IEEE Trans. Comput.*, vol. C-35, no 4, pp. 339-349, April 1986.

[14] T. Krol and B. J. Vonk, U.S. patent pending.

[15] J. H. van Lint and R. M. Wilson, "On the minimum distance of cyclic codes", *IEEE Trans. Inform. Theory*, vol. IT-32, no 1, pp. 23-40, January 1986.

[16] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland Publishing Company, 1977.

[17] C. L. Mallows and N. J. A. Sloane, "An upper bound for self-dual codes", *Inform. and Contr.*, vol. 22, pp. 188-200, 1973.

[18] R. J. McEliece, "Weight congruences for p-ary cyclic codes", *Discrete Math.*, vol. 3, pp. 177-192, 1972.

[19] G. Pasquier, *Etude de codes sur une extension de* $GF(2)$ *et de leurs images binaires*, These de 3eme cycle, Univ. de Provence, Toulon, France, 1980.

[20] G.Pasquier, "The binary Golay code obtained from an extended cyclic code over $GF(8)$", *European J. Combin.*, vol. 1, pp.369-370, 1980.

[21] G. Pasquier, "A binary extremal doubly even self-dual code (64,32,12) obtained from an extended Reed-Solomon code over $GF(16)$", *IEEE Trans. Inform. Theory*, vol. IT-27, no 6, pp. 807-808, November 1981.

[22] W. W. Peterson and E. J. Weldon jr., *Error-Correcting Codes*, 2nd. ed., MIT Press, 1972.

[23] P. Piret, "Binary codes for compound channels", *IEEE Trans. Inform. Theory*, vol. IT-31, no 3, pp. 49-56, May 1985.

[24] P. Piret, "Convolutional codes for unequal error protection in time-varying channels", submitted to *IEEE Trans. Inform. Theory.*

[25] G. Promhouse and S. E. Tavares, "The minimum distance of all binary cyclic codes of odd lengths from 69 to 99", *IEEE Trans. Inform. Theory*, vol. IT-24, no 4, pp. 438-442, July 1978.

[26] S. M. Reddy, "A class of linear codes for error control in byte per card organized digital systems", *IEEE Trans. Comput.*, vol. C-27, no 5, pp. 455-459, May 1978.

[27] G. Seroussi and A. Lempel, "Factorization of symmetric matrices and trace-orthogonal bases in finite fields", *SIAM J. Comput.*, vol. 9, pp. 758-767, 1980.

[28] L. M. G. M. Tolhuizen, *On the Optimal Use and the Construction of Linear Block Codes*, Master's Thesis, Eindhoven University of Technology, Department of Mathematics and Computing Science, November 1986.

[29] M. R. Varanasi, T. R. N. Rao and Son Pham, "Memory package error detection and correction", *IEEE Trans. Comput.*, vol. C-32, no 9, pp.872-874, September 1983.

[30] T. Verhoeff, "An updated table of minimum distance bounds for binary linear codes", submitted to *IEEE Trans. Inform. Theory.*

Ontwerp van foutencorrigerende codeerschemas voor drie problemen van, door ruis gestoorde, informatie transmissie, opslag en verwerking

Samenvatting

Dit proefschrift handelt over het ontwerp van foutencorrigerende codeerschemas voor drie verschillende problemen van, door ruis gestoorde, informatie transmissie, opslag en verwerking. Het gemeenschappelijke van deze problemen is hun praktisch, industrieel belang en het feit dat ze niet elegant opgelost kunnen worden met behulp van traditionele foutencorrigerende codeerschemas.

Het eerste probleem behelst de transmissie en opslag van berichten bestaande uit delen van onderling verschillende importantie. Het is dan voor de hand liggend deze, in belangrijkheid verschillende, delen van een verschillende bescherming tegen fouten te voorzien. Dit kan door middel van het gebruik van verschillende codeerschemas voor de verschillende delen, maar ook op een meer elegante manier door middel van een zogenaamd codeerschema voor ongelijke foutenprotectie.

Het tweede codeerschema is ontworpen voor het gebruik als automatisch leesbare identificatie code in een geautomatiseerde fabricage omgeving. Het identificatie nummer (en zo mogelijk andere bruikbare informatie) van een produkt wordt gecodeerd in een vierkante matrix van ronde puntjes op een contrasterende achtergrond. Problemen die in de praktijk optreden zijn de rotatie van punt matrices en de verminking van puntjes als gevolg van schrijffouten, stofdeeltjes en leesfouten. Ter oplossing zijn broncodes en zogenaamde vierkant cyclische kanaalcodes ontworpen.

Het derde deel van dit proefschrift beschrijft codeerschemas voor systemen waarbinnen zowel digit fouten als symbool fouten

optreden, waar een symbool een positiegebonden groep van digits is. Voorbeelden van zulke systemen zijn computers en samengestelde kanalen. We geven de gedetailleerde ontwerpen van codes en decodeurs voor drie speciale toepassingen. Deze zijn een gegeneraliseerd verdrievoudigd computer systeem, een geheugen systeem bestaande uit drie 9-bits brede eenheden voor opslag van 16-bits woorden, en een '(4,2) concept' foutentolererende computer. Tenslotte wordt enige algemene theorie over deze zogenaamde gecombineerde symbool en digit foutencorrigerende codes ontwikkeld.

Curriculum vitae

The author was born in Tilburg, The Netherlands, on November 20, 1956. He received the M.Sc. degree in mathematics from the Eindhoven University of Technology, Eindhoven, The Netherlands, in 1982. Since August 1982, he has been with the Philips Research Laboratories, Eindhoven.

STELLINGEN

behorende bij het proefschrift van W.J. van Gils
5 januari 1988, Technische Universiteit Eindhoven

I

Kennis over de automorfisme groep van een lineaire code kan de hoeveelheid werk benodigd voor het berekenen van een optimaal encodeur/decodeur paar aanzienlijk verminderen.

L.M.G.M. Tolhuizen and W.J. van Gils, *A large automorphism group decreases the number of computations in the construction of an optimal encoder/decoder pair for a linear block code*, to appear in IEEE Trans. on Information Theory.

II

Het voortdurende gesteggel over de begrippen 'fault', 'error' en 'failure' binnen het vakgebied 'fault-tolerant computing' geeft aan dat dit vak nog in een infantiel stadium verkeert.

III

In een volledig verbonden netwerk van n processoren, waarvan er ten hoogste t onbetrouwbaar zijn, is het mogelijk interactieve consistentie te bereiken dan en slechts dan als n groter of gelijk aan $3t + 1$ is.

W.J. van Gils, *How to cope with faulty processors in a completely connected network of communicating processors*, Information Processing Letters, vol. 20, pp. 207-213, May 1985.

IV

Het niet bezorgen van landelijke dagbladen in dunbevolkte agrarische gedeelten van Nederland vergroot de intellectuele achterstand van de plattelandsbevolking aldaar.

V

De door Mao-Chao Lin afgeleide Hamming grens voor lineaire codes met ongelijke foutenprotectie is incorrect.

Mao-Chao Lin, *Coding for Unequal Error Protection*, Ph.D. dissertation University of Hawaii, december 1986.

VI

De gecombineerde bit- en symboolfoutencorrigerende capaciteit van de binaire [27,16] code uit de octrooiaanvrage van Bannon, Bhansali, Minnich, Finney, Suarez en Chisholm kan ook verkregen worden met een veel eenvoudiger codeerschema, dat aan elk bericht bestaande uit twee bytes een derde redundant byte, zijnde hun som, toevoegt en verder drie pariteitsbits, één voor elk byte, die het aantal enen modulo twee in een byte tellen.

US Patent Application 391,062

VII

Het vereenvoudigen van het sociale verzekeringsstelsel of van het belastingstelsel in Nederland blijft een illusie zolang men het belang van ieder individu wil blijven behartigen.

VIII

Door handhaving van de verplichting tot het schrijven van een aantal stellingen bij een proefschrift, behoudt een lid van de promotiecommissie de mogelijkheid zich op een eenvoudige wijze van zijn taak te kwijten door tegen een stelling te opponeren.