The design of the IEEE 802.12 coding scheme

Simon E.C. Crouch James A. Davis Jonathan Jedwab

30 December 2004 (revised 1 August 2006)

Abstract

In 1995 the IEEE approved the 802.12 standard for data transmission at 100-Mbit/s using the Demand Priority network access protocol. 100VG-AnyLAN products conforming to this standard offered an upgrade path for Ethernet and Token Ring networks, without requiring new building wiring. A key factor in the approval of the 802.12 standard was the demonstrated error detection properties of its coding scheme. In particular, the coding scheme allows the detection of error bursts affecting encoded data carried on four parallel conductors, using nothing more than the standard IEEE 32-bit cyclic redundancy check applied to the unencoded data. Although these error detection properties were presented for verification as part of the standards process, for many years commercial considerations prevented public disclosure of how the code was actually found. These considerations no longer apply, and in this paper we explain in detail the design principles of the code, combining geometrical insight, linear algebra, combinatorial reasoning, and computer search.

Keywords

cyclic redundancy check, error detection, polynomial, IEEE 802.12, 100VG-AnyLAN

1 The code design problem for IEEE 802.12

In 1992 the Higher Speed Study Group of IEEE 802.3 began considering technical proposals to increase the speed of 10-Mbit/s Ethernet networks by a factor of ten, to 100-Mbit/s. A proposal that almost, but not entirely, retained the original Medium Access Control (MAC) protocol was standardised in IEEE 802.3u in 1995 and marketed as Fast Ethernet. Also in 1995, an alternative proposal using the new Demand Priority MAC protocol [1] was standardised in IEEE 802.12 [2] and marketed as 100VGAny-LAN [3]. Although Fast Ethernet ultimately prevailed over 100VG-AnyLAN in the marketplace, both technologies were responsible for the sale of millions of network devices: in 1995 alone, Fast

S.E.C. Crouch is with the Epidemiology and Genetics Unit, Department of Health Sciences, University of York, York YO10 5DD, United Kingdom.

J.A. Davis is with the Department of Mathematics and Computer Science, University of Richmond, VA 23173. He is grateful for support from NSA grant MDA 904-03-1-0032.

J. Jedwab is with the Department of Mathematics, Simon Fraser University, 8888 University Drive, Burnaby BC, Canada V5A 1S6. He is grateful for support from NSERC of Canada.

Ethernet accounted for 665,000 adapter sales while 100VG-AnyLAN accounted for 218,000 (reported in [4], quoting market research from IDC).

The basic operating environment of IEEE 802.3 at 10-Mbit/s involves two conductors: a first for data transmission, and a second for collision detection using the Carrier Sense Multiple Access with Collision Detection MAC [5]. Using "Manchester" encoding, a data '1' is transmitted as the code symbol pair 01 and a data '0' is transmitted as the code symbol pair 10. A data rate of 10-Mbit/s is achieved by transmitting code symbols at a rate of 20-Mbit/s over a single conductor. The rapid proliferation of IEEE 802.3 networks in the late 1980s was spurred by the realisation that it was often unnecessary to incur the significant expense of installing new data cabling: each of the required conductors could be a twisted pair of copper telephone wiring originally intended for voice use.

Although IEEE 802.3 requires only two conductors, surveys in the early 1990s revealed a large installed base of Category 3 telephone cabling comprising four twisted pairs of wires, with two of the twisted pairs remaining unused for 10-Mbit/s transmission purposes. A key objective of IEEE 802.12 was to allow operation at 100-Mbit/s by making use of all four of these twisted pairs (also known as voice-grade cabling, hence the trade name "VG"), and this is the physical environment studied in this paper. We note however that IEEE 802.12 was also designed to operate in two less challenging environments, namely: two twisted pairs of shielded cabling (as implemented in Token Ring networks), and a single optical fibre (allowing connection of widely separated hubs and end nodes) [6]. Furthermore the IEEE 802.12 codewords for a four-conductor environment can easily be multiplexed to obtain the codewords required for a two-pair or single-fibre environment [7].

The Demand Priority MAC allows all four conductors to be used for data transmission simultaneously. Various binary and multilevel transmission schemes were considered as more efficient alternatives to 1-bit/2-bit Manchester encoding. The scheme selected for IEEE 802.12, after careful consideration of radiated emission and noise susceptibility properties [6], was 5-bit/6-bit encoding. With this choice, code symbols are transmitted at a rate of 30-Mbit/s over each conductor to give a data transmission rate of 25-Mbit/s over each of four conductors, for a combined data transmission rate of 100-Mbit/s. For details on how the code symbol transmission rate was increased, from 20-Mbit/s over a single conductor to 30-Mbit/s over each of four conductors, while dealing with issues of cross talk, see [6].

The *code design problem* for IEEE 802.12 is to select a particular mapping of 5-bit data values to 6-bit code values in order to satisfy four constraints simultaneously:

1. the number of transmitted 0's should closely match the number of transmitted 1's for each conductor, in order to achieve near-perfect DC balance and so control baseline wander at the receiver

2. the run length of transmitted symbols (namely the maximum number of identical consecutive symbols) should be small for each conductor, in order to ensure a high density of signal transitions and so assist accurate clock recovery at the receiver

3. the code should guarantee the detection of errors arising from the corruption of up to three code bits located anywhere within a single encoded data packet, in order to

satisfy IEEE 802 Functional Requirement 5.6.3 on Hamming distance (see Section 4)

4. the code should guarantee the detection of a significant duration of burst error arising from the arbitrary corruption of code bits occurring across all four conductors in parallel.

Table 1 shows the actual choice of 5-bit/6-bit code standardised in IEEE 802.12. We shall describe how this code was designed to satisfy each of the above constraints in turn.

2 DC balance

Consider the stream of data bits shown in Figure 1, with the order of bit transmission from left (first) to right (last). The data stream is split into five-bit data words $D_7, D_6, \ldots, D_0$ prior to encoding using the 5-bit/6-bit code. The resulting stream of codewords $C_7, C_6, \ldots, C_0$ is assigned to the four parallel conductors in a cyclic manner, with the order of transmission of code bits on each conductor from bottom (first) to top (last).

We wish to satisfy the constraint of DC balance for each conductor. There are $\binom{6}{3} = 20$ 6-bit codewords that are balanced, namely those having weight 3, and clearly all of these should be included in the 5-bit/6-bit code. We then assign one weight 2 codeword and one weight 4 codeword to each of the remaining 12 5-bit data words. For each conductor independently, an *alternation rule* is implemented: the first time that any one of these 12 data words is to be encoded we choose the codeword of weight 2; subsequently, whenever an unbalanced codeword is required (corresponding to any of the 12 data words), we choose the opposite weight to that previously selected for that conductor. In this way, the sequence of unbalanced codewords on each conductor (ignoring balanced codewords) alternates between weight 2 and weight 4, giving near-perfect DC balance.

At this point we have a free choice as to which 12 of the $\binom{6}{2} = 15$ codewords of weight 2 and which 12 of the 15 codewords of weight 4 should be included in the code, as well as to the assignment of codewords to data words.

3 Run length

Given that all balanced codewords are included in the code, the run length is at least six because the codewords 111000 and 000111 can be transmitted on the same conductor in succession. We now force the run length to be exactly six by excluding the unbalanced codewords 111100, 000011, 110000, and 001111 from the code.

It remains to exclude one further weight 2 and weight 4 codeword.

4 Single-code-bit error detection

Before describing how single-code-bit errors are detected in IEEE 802.3 and IEEE 802.12 we shall review the algorithm for calculating and checking cyclic redundancy check (CRC) bits.

The 32-bit CRC algorithm uses a fixed polynomial $g(x)$ of degree 32 with 0–1 coefficients, which in the case of an IEEE 802 Local Area Network is the primitive polynomial

$$g(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1. \quad (1)$$

Let an unencoded data packet comprise the bits $f_{k-1}, f_{k-2}, \ldots, f_0$, where f_{k-1} is the first transmitted bit. We associate the polynomial $F(x) = \sum_{i=0}^{k-1} f_i x^i$ with these bits and define the CRC check polynomial for these bits to be

$$C(x) = \Big(x^{32} F(x) + (x^k + 1) L(x) \Big) \bmod g(x),$$

where $L(x) = \sum_{i=0}^{31} x^i$ and calculations are carried out in $\mathbb{Z}_2[x]$. The 32 bits associated with $C(x)$ are appended to the k data bits (f_i), giving a transmitted message polynomial $M(x) = x^{32} F(x) + C(x)$. Suppose the transmitted message suffers a corruption in transit (possibly affecting the CRC bits themselves), so that the received message polynomial is $M'(x) = M(x) + E(x)$. The CRC algorithm, in effect, calculates the value of $M'(x) \bmod g(x)$ and compares it with the value of $M(x) \bmod g(x)$ that would be received in the absence of transmission errors. Note that the value $M(x) \bmod g(x)$ depends only on k, not on the data bit values (f_i). The corruption is detectable (although not correctable) provided that $M'(x) \bmod g(x) \neq M(x) \bmod g(x)$:

$$\text{the error } E(x) \text{ is detectable by the CRC algorithm if and only if } E(x) \bmod g(x) \neq 0. \quad (2)$$

It follows from (2) that any error burst affecting 32 or fewer consecutive data bits is detectable by the CRC algorithm. Furthermore the 32 check bits of the CRC allow certain patterns of widely separated single-code-bit errors to be detected. In the case of IEEE 802.3, each data packet contains at most 1518 octets (8-bit data groups). There are no solutions to the polynomial equation

$$(1 + x^r) \bmod g(x) = 0 \quad \text{where integer } r \text{ satisfies } 0 < r < 8 \cdot 1518$$

because $g(x)$ is primitive, and by computer search [9] neither are there any solutions to the polynomial equation

$$(1 + x^r + x^s) \bmod g(x) = 0 \quad \text{where integer } r, s \text{ satisfy } 0 < r < s < 8 \cdot 1518.$$

Therefore any two or three single-code-bit errors in an IEEE 802.3 data packet are detectable by the CRC, as required by IEEE Functional Requirement 5.6.3 [8]: "A minimum of four bit cells in error shall be necessary for an undetected error to occur (Hamming distance 4)". Note that IEEE 802.3 does not contain any provision for declaring a data packet to be in error on the basis of receiving an invalid Manchester-encoded codeword 00 or 11, and so single-code-bit errors must be assumed to lead to single-data-bit errors in IEEE 802.3.

In contrast, in IEEE 802.12 the physical layer is able to notify the Demand Priority MAC of invalid codewords, namely codewords that do not appear in the 5-bit/6-bit code table. An error that changes one or more bits of a codeword to produce another valid

codeword *induces* an error in the original data word. For example, if the data word 01011 is encoded to 111001 according to Table 1, and one code bit is corrupted during transmission so that the received codeword is 011001, then the decoded data word is 10110. The effect of the single-code-bit error is to induce the data error $01011 + 10110 = 11101$. In general the consequence of one or more single-code-bit errors affecting a given 6-bit codeword, if the resulting codeword is valid, is to induce an error $a(x)$ of degree less than 5 in the decoded data. (IEEE 802.12 also specifies a stream cipher to be added to the data bits prior to encoding in order to avoid repetitive data patterns, and the same stream cipher bits are added to the decoded data. This affects the induced data error for a particular data word, but does not alter the set of induced data errors across all data words under a particular corruption such as change of the first transmitted bit of a codeword. Since we are only concerned with the latter we can ignore the stream cipher in our error analysis.)

IEEE 802.12 specifies a packet size of either 1518 data octets for use with IEEE 802.3 packets, or 4096 data octets for use with Token Ring packets. CRC check bits are calculated using the polynomial (1), but are applied to the data *prior* to 5-bit/6-bit encoding. Although a further CRC applied to the encoded bits would have greatly facilitated error detection in IEEE 802.12, this was considered in the standardisation process to be an unacceptable complication to the protocol.

There are three methods by which transmission errors can be detected in an IEEE 802.12 packet:

1. one or more invalid 6-bit codewords
2. violation of the alternation rule on one or more conductors (for example, two weight 4 codewords on a particular conductor separated only by weight 3 codewords)
3. an invalid CRC for the decoded data.

An error that is not detected by any of these three methods is *undetectable*.

IEEE 802.12 strengthens the alternation rule by employing, on each conductor, two possible end delimiters ED2 and ED4 to mark the end of the sequence of codewords of a given packet. The chosen delimiter indicates the expected weight of the next unbalanced codeword. By a parity argument, any odd number of single-code-bit errors affecting distinct codewords on a particular conductor will cause a violation of the alternation rule, either within the sequence of codewords or at the end delimiter. It follows that the alternation rule will detect any odd number of single-code-bit errors occurring anywhere within an encoded data packet and affecting distinct codewords.

In order to check that IEEE Functional Requirement 5.6.3 on Hamming distance is satisfied we must consider the effect of two further cases: two single-code-bit errors affecting distinct codewords within an encoded data packet; and three single-code-bit errors, two of which affect the same codeword. Both of these cases are dealt with by the result, from computer search, that there are no solutions to the polynomial equation

$$\left[a(x) + x^{5r}b(x)\right] \bmod g(x) = 0 \quad \text{where } \deg a(x), b(x) < 5 \text{ and integer } r \text{ satisfies } 0 < 5r < 8 \cdot 4096.$$

(This result does not depend on the allocation of codewords to the parallel conductors, nor on the choice of 5-bit/6-bit code table. It can be shown that the case of three single-code-bit errors, two of which affect the same codeword, will always be detected by the alternation rule and so in fact will be rejected prior to CRC checking.) Hence IEEE 802.12 achieves Hamming distance 4, as required by IEEE Functional Requirement 5.6.3.

5 Burst error detection

We have shown how IEEE 802.12 detects single-code-bit errors. We now consider the detection of burst errors due, for example, to electrical interference affecting all four parallel conductors simultaneously. Figure 1 shows a burst error of duration two code bit periods that can arbitrarily corrupt eight code bits (to any one of $2^8 - 1 = 255$ corrupted states). When the error burst straddles the boundary between codewords, as shown in Figure 1, the error induced on the unencoded data can involve eight consecutive data words $D_7, D_6, \ldots, D_0$ and as many as 40 consecutive data bits. Since the CRC is guaranteed to detect burst errors of only 32 or fewer consecutive data bits, we see that without a careful choice of 5-bit/6-bit code table even an error burst as short as two code bits in duration could lead to an undetectable error! Strengthening the burst error detection capability of the four-conductor proposals was a high priority in the standardisation process of 100Mbit/s IEEE 802.3 and 802.12.

A major contribution to improving burst error detection arises from the geometrical insight that the arrangement of codewords on conductors shown in Figure 1 is unnecessarily constrained. Consider instead the arrangement shown in Figure 2, in which the codewords are still allocated cyclically to the conductors but the transmission of codewords on two of the conductors is offset by three code bit periods relative to the other two conductors. In the offset arrangement, an error burst of four or fewer code bit periods can affect no more than six consecutive data words and therefore no more than 30 data bits. Hence the CRC will detect the data error induced by such a burst regardless of the choice of 5-bit/6-bit code table.

We have just seen that the burst error detection capability of the four-conductor system can be increased from one to four code bit periods simply by the offset arrangement of Figure 2. In the remainder of this paper we show how to increase further the burst error detection from four to seven code bit periods by means of the choice of 5-bit/6-bit code table.

We must consider three possible configurations of an error burst of duration seven code bit periods relative to eight successive codewords $C_7, C_6, \ldots, C_0$. These configurations are labelled as A, B and C in Table 2, which shows the number of most significant bits of codewords C_1 and C_0, and the number of least significant bits of codewords C_7 and C_6, that are affected by the burst in each of the configurations. The seven-bit burst shown in Figure 2 has configuration C. We use "most significant" to mean the leftmost (lowermost, first transmitted) bits of a codeword and "least significant" to mean the rightmost (uppermost, last transmitted) bits. Codewords C_5, C_4, C_3 and C_2 can be so heavily corrupted by the burst, in each of the configurations, that we do not attempt to

control the errors affecting them.

To emphasise the difficulty of the code table selection problem, we count the number of possible sets of codewords $(C_7, C_6 \ldots, C_0)$ consistent with the alternation rule as $(44^2 - 2 \cdot 12^2)^4$. (For each of the four wires containing codeword pairs such as C_7 and C_3, the codewords of a pair can each take $32 + 12 = 44$ possible values but we exclude pairs for which both codewords have the same weight 2 or the same weight 4 since they would violate the alternation rule.) Each of these sets can be corrupted to any one of $2^{4 \cdot 7} - 1$ states by an error burst of duration 7 code bit periods, in each of configurations A, B and C. Therefore the total number of error cases handled simultaneously is $(44^2 - 2 \cdot 12^2)^4 (2^{28} - 1) \sim 10^{22}$. The number of possible code tables to select from, that are consistent with the choices already made to satisfy the DC balance and run length constraints (see Sections 2 and 3), is

$$\left(\frac{32!}{12!}\right) (13 \cdot 12!)^2 \sim 10^{46}.$$

(The first factor arises by assigning the balanced codewords to 20 data words. The second factor arises by choosing which weight 2 codeword out of 13 and which weight 4 codeword out of 13 to eliminate, and then assigning the retained weight 2 and weight 4 codewords in pairs to the remaining 12 data words.)

Jain [9] gives a very careful analysis of the error detection properties of the (previously determined) FDDI 4-bit/5-bit transmission code, according to a model of physical errors corrupting code bits. However we are not aware of any previous analysis which shows how to choose a block encoding in advance so that all possible induced data errors arising from a large given set of physical transmission errors will be detectable by a CRC calculated on the unencoded data and using a predetermined CRC polynomial. A summary of the burst error detection properties of IEEE 802.12 was given in [7] but no indication was given as to how the code table was selected to achieve these properties.

5.1 Configurations A and C

We will use linear algebra (Sections 5.1.1 and 5.1.2) and combinatorial reasoning (Section 5.1.3) to deal with configurations A and C, as defined in Table 2 with respect to eight codewords $C_7, C_6, \ldots, C_0$ corresponding to at most 40 data bits. A key observation is that, for both configurations, there are two codewords that can be corrupted in only one bit position. We will exploit this to avoid all 255 undetectable induced data errors occupying at most 40 bits. From (2), these errors are represented by the 255 polynomials

$$S = \{j(x)g(x) : j(x) \neq 0, j(x) \in P_7\},$$

where P_7 denotes the set of polynomials of degree at most 7.

We wish to represent each of the polynomials of S as comprising eight 5-bit data groups. To do so, let P_4 denote the set of polynomials of degree at most 4 and define the mappings $T_i : P_7 \mapsto P_4$ for $i = 0, 1, \ldots, 7$ by

$$x^{35}T_7(j(x)) + x^{30}T_6(j(x)) + \ldots + x^5 T_1(j(x)) + T_0(j(x)) := j(x)g(x) \quad \text{for } j(x) \in P_7. \tag{3}$$

It is easy to check that each T_i is a linear map. Although some of the linear algebra arguments we present could be viewed more succintly in the framework of vector spaces, we will often use an explicit treatment in terms of matrices. We write

$$T_i(j(x)) \longleftrightarrow E_i j \quad \text{for } i = 0, 1, \ldots 7 \text{ and } j(x) \in P_7,$$

where E_i is a 5×8 matrix, and we use the natural correspondence between the polynomial $j(x) = j_7 x^7 + j_6 x^6 + \ldots + j_0$ and the vector $j = (j_7, j_6, \ldots, j_0)^T$. This allows us to write

$$j(x)g(x) \longleftrightarrow ((E_7 j)^T, (E_6 j)^T, \ldots, (E_0 j)^T)^T \quad \text{for } j(x) \in P_7.$$

Now from (1) and (3) we can write the E_i explicitly, in particular:

$$E_0 = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \quad E_1 = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \end{pmatrix},$$

$$E_6 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}, \quad E_7 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

For example, taking $j(x) = x^6 + x^3 + x^2 + 1$, we have $T_0(x^6 + x^3 + x^2 + 1) \longleftrightarrow E_0[01001101]^T = [10011]^T \longleftrightarrow x^4 + x + 1$.

Denote by M_i (respectively L_i) the set of possible induced 5-bit data errors arising from an arbitrary corruption of the i most significant (respectively least significant) bit positions of any 6-bit codeword. For each codeword, each of the $2^i - 1$ possible corruptions of each codeword that is itself a valid codeword results in an induced data error. We will deal with configurations A and C simultaneously by means of the following plan:

(P1) Determine the set $\mathcal{P}$ of triples of distinct 5-bit non-zero values $\{a, b, c\}$ for which the 16 ordered pairs in $\{0, a, b, c\} \times \{0, a, b, c\}$ intersect with the 256 ordered pairs of 5-bit values $\{(E_1 j, E_0 j) : j(x) \in P_7\}$ only in the element $(0, 0)$.

(P2) Determine the set $\mathcal{P}'$ of triples of distinct 5-bit non-zero values $\{\alpha, \beta, \gamma\}$ for which the 16 ordered pairs in $\{0, \alpha, \beta, \gamma\} \times \{0, \alpha, \beta, \gamma\}$ intersect with the 256 ordered pairs of 5-bit values $\{(E_7 j, E_6 j) : j(x) \in P_7\}$ only in the element $(0, 0)$.

(P3) Select a 5-bit/6-bit code mapping so that, for some $\{a, b, c\} \in \mathcal{P}$ and some $\{\alpha, \beta, \gamma\} \in \mathcal{P}'$, we have $M_1 \subseteq \{a, b, c\}$ and $L_1 \subseteq \{\alpha, \beta, \gamma\}$.

This will ensure that, regardless of the values of the eight codewords $(C_7, C_6, \ldots, C_0)$ carried on the four conductors, any error burst having configuration A or C will be detectable by means of invalid codewords and the CRC. In the case of configuration A, we

avoid all 255 undetectable errors in S by controlling the induced 5-bit data errors from change of the least significant bit of any values of codewords C_7 and C_6. In the case of configuration C, we do likewise by controlling the induced 5-bit data errors from change of the most significant bit of any values of codewords C_1 and C_0.

5.1.1 Determination of $\mathcal{P}$

In this subsection we determine the possible triples $\{a, b, c\}$ of $\mathcal{P}$, as specified in (P1). We write $\mathrm{col}(A)$ for the column space of a matrix A.

The nullspace of the matrix E_i (namely the set of vectors j for which $E_i j = 0$) can be represented as the column space of a matrix N_i. For $i = 0, 1, 6$ and 7 we calculate N_i explicitly as:

$$N_0 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad N_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}, \quad N_6 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad N_7 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

By inspection, each of the matrices E_i $(i = 0, 1, 6, 7)$ has rank 5 and each of the matrices N_i $(i = 0, 1, 6, 7)$ has rank 3. Likewise, the matrices $E_0 + E_1$ and $E_6 + E_7$ also have rank 5, and their corresponding nullspaces are given by the column space of respective matrices N_{01} and N_{67}, also of rank 3:

$$N_{01} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}, \quad N_{67} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}.$$

We now constrain the possible triples of $\mathcal{P}$.

Lemma 1 *No vector contained in* $\mathrm{col}(E_1 N_0)$, $\mathrm{col}(E_0 N_1)$ *or* $\mathrm{col}(E_0 N_{01})$ *can appear in a triple of* $\mathcal{P}$.

Proof: Consider a non-zero polynomial $j(x) \in P_7$ whose associated vector j satisfies $E_0 j = 0$, so that j is contained in $\mathrm{col}(N_0)$. By the definition of $\mathcal{P}$, the ordered pair $(E_1 j, E_0 j) = (E_1 j, 0)$ cannot occur as an element of $\mathcal{P} \times \{0\}$; therefore no vector contained in $\mathrm{col}(E_1 N_0)$ can appear in a triple of $\mathcal{P}$. Similarly, no vector contained in $\mathrm{col}(E_0 N_1)$ can appear in a triple of $\mathcal{P}$. Furthermore, consider a vector j contained in $\mathrm{col}(N_{01})$. The

ordered pair $(E_1 j, E_0 j) = ((E_1 + (E_0 + E_1))j, E_0 j) = (E_0 j, E_0 j)$ cannot occur as an element of $\mathcal{P} \times \mathcal{P}$ and so no element of $\mathrm{col}(E_0 N_{01})$ can appear in a triple of $\mathcal{P}$. $\square$

The matrices $E_1 N_0$, $E_0 N_1$, and $E_0 N_{01}$ involved in Lemma 1 each have rank 3:

$$E_1 N_0 = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}, \quad E_0 N_1 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}, \quad E_0 N_{01} = E_1 N_{01} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}.$$

Lemma 2 *If $\{a, b, c\}$ is a triple in $\mathcal{P}$, and $a + [00101]^T \notin \{0, b, c\}$, then $\{a + [00101]^T, b, c\}$ is a triple in $\mathcal{P}$.*

Proof: If, for some $j(x) \in P_7$, we have $(E_1 j, E_0 j) = (a_1, a_0)$ then

$$(E_1(j + [11100000]^T), E_0(j + [11100000]^T)) = (a_1 + [00101]^T, a_0)$$

and

$$(E_1(j + [00000111]^T), E_0(j + [00000111]^T)) = (a_1, a_0 + [00101]^T).$$

The condition that $a + [00101]^T \notin \{0, b, c\}$ ensures that the elements of $\{a + [00101]^T, b, c\}$ are distinct and non-zero. $\square$

Lemmas 1 and 2 point to the use of Table 3, in which the 32 possible 5-bit induced data error values are arranged in a 4×4 table, with values that differ by $[00101]^T$ grouped in pairs. The table entries are arranged so that offsets of $\mathrm{col}(E_1 N_0)$ appear horizontally, offsets of $\mathrm{col}(E_0 N_1)$ appear vertically, and the main diagonal contains $\mathrm{col}(E_0 N_{01})$. Lemma 1 excludes any non-zero 5-bit value contained in the uppermost row ($\mathrm{col}(E_1 N_0)$), the leftmost column ($\mathrm{col}(E_0 N_1)$), or the main diagonal ($\mathrm{col}(E_0 N_{01})$) of the table from appearing in a triple of $\mathcal{P}$. The elements of all triples $\{a, b, c\}$ in $\mathcal{P}$ must therefore be drawn from the 12 remaining non-zero 5-bit values, which are highlighted in Table 3.

We next introduce a lemma on the pairwise intersection of offsets of column spaces of the matrices $E_0 N_1$, $E_1 N_0$ and $E_0 N_{01}$, which further illustrates the underlying structure of Table 3.

Lemma 3 *For any 5-bit vectors h and h', there is a 5-bit vector h'' for which*

$$(h + \mathrm{col}(E_0 N_1)) \cap (h' + \mathrm{col}(E_1 N_0)) = \{h'', h'' + [00101]^T\}.$$

Similar statements hold for the intersection of offsets of $\mathrm{col}(E_0 N_1)$ and $\mathrm{col}(E_0 N_{01})$, and for the intersection of offsets of $\mathrm{col}(E_1 N_0)$ and $\mathrm{col}(E_0 N_{01})$.

Proof: Let the columns of $E_0 N_1$ be (v_1, v_2, v_3) and the columns of $E_1 N_0$ be (w_1, w_2, w_3). The intersection of $h + \mathrm{col}(E_0 N_1)$ and $h' + \mathrm{col}(E_1 N_0)$ is given by the solutions of

$$h + a_1 v_1 + a_2 v_2 + a_3 v_3 = h' + b_1 w_1 + b_2 w_2 + b_3 w_3 \tag{4}$$

for 0–1 coefficients $a_1, a_2, a_3, b_1, b_2, b_3$, or equivalently

$$h + h' = [E_0N_1; E_1N_0]\,[a_1; a_2; a_3; b_1; b_2; b_3]^T. \tag{5}$$

Now, by inspection, the 5×6 matrix $[E_0N_1; E_1N_0]$ has full rank 5 and so (5) has a solution $[a_1; a_2; a_3; b_1; b_2; b_3]^T$ for all values of h and h'. Furthermore, by the fundamental theorem of linear algebra, $[E_0N_1; E_1N_0]$ has nullspace of dimension 1, and we can verify directly that this nullspace is $\{[000000]^T, [100111]^T\}$. Therefore (5) has exactly two solutions for all h and h', namely $[a_1; a_2; a_3; b_1; b_2; b_3]^T$ and $[a_1; a_2; a_3; b_1; b_2; b_3]^T + [100111]^T$. Equivalently, (4) has exactly two solutions, namely $h'' = h + a_1v_1 + a_2v_2 + a_3v_3$ and $h'' + v_1 = h'' + w_1 + w_2 + w_3 = h'' + [00101]^T$ for some 5-bit vector h''.

Similar arguments hold for the other intersections. □

Proposition 4 *The triples $\{a, b, c\}$ of $\mathcal{P}$ comprise all sets of three distinct highlighted 5-bit values appearing in the same row or column of Table 3.*

Proof: The 5-bit values of Table 3 are arranged in pairs, the members of each pair differing by $[00101]^T$. By Lemma 1, only the 12 highlighted values (arranged in 6 pairs) can belong to a triple $\{a, b, c\}$ of $\mathcal{P}$, since these are the 5-bit values that are not contained in $\mathrm{col}(E_1N_0)$, $\mathrm{col}(E_0N_1)$ or $\mathrm{col}(E_0N_{01})$. Suppose a is such a highlighted value and belongs to a triple $\{a, b, c\}$ of $\mathcal{P}$; we now determine the allowable values of b and c for this triple.

Since E_1 has full rank 5, we can find $j(x) \in P_7$ satisfying $E_1j = a$. In order to satisfy the definition of $\mathcal{P}$, as specified in (P1), b cannot take any value E_0k for which there is some $k(x) \in P_7$ satisfying $E_1k = a$. The 8 values k satisfying $E_1k = a$ are given by $k \in j + \mathrm{col}(N_1)$, so this means that b cannot take any of the 4 pairs of values in $E_0j + \mathrm{col}(E_0N_1)$; these values comprise some column of Table 3. Likewise we can find $j'(x) \in P_7$ satisfying $E_0j' = a$ and, by a similar argument, b cannot take any of the 4 pairs of values in $E_1j' + \mathrm{col}(E_1N_0)$; these values comprise some row of Table 3.

We next determine which of the 6 highlighted pairs of data values are removed from consideration by the exclusion of the values in this column and row of Table 3. We claim that the excluded column $E_0j + \mathrm{col}(E_0N_1)$ intersects the row $a + \mathrm{col}(E_1N_0)$ containing the value a in a pair of values lying on the main diagonal $\mathrm{col}(E_0N_{01})$ of Table 3. To establish this, suppose

$$k \in (E_0j + \mathrm{col}(E_0N_1)) \cap (a + \mathrm{col}(E_1N_0))\,. \tag{6}$$

We know from Lemma 3 that the right-hand side of (6) comprises a pair of values contained in some row and column of Table 3. Now from (6) we can write

$$k = E_0j + E_0n_1 = a + E_1n_0$$

for some $n_1 \in N_1$ and $n_0 \in N_0$. Substitution of $a = E_1j$ and $E_1n_1 = E_0n_0 = 0$ gives

$$k = E_0(j + n_0 + n_1) = E_1(j + n_0 + n_1).$$

Setting $n = j + n_0 + n_1$ gives $k = E_0n = E_1n$ and so, by definition of N_{01}, we obtain $k \in \mathrm{col}(E_0N_{01})$ as claimed. By a similar argument, the excluded row $E_1j' + \mathrm{col}(E_1N_0)$

intersects the column $a + \mathrm{col}(E_0 N_1)$ containing the value a in a pair of values lying on the main diagonal of Table 3.

In summary, given a highlighted value a belonging to a triple $\{a, b, c\}$ of $\mathcal{P}$, each of b and c is restricted to one of the 5 highlighted values lying in either the same row or the same column of Table 3 as a. Application of this result to the possible values of a and c associated with a given value of b then forces any triple $\{a, b, c\}$ of $\mathcal{P}$ to be formed by taking three out of four highlighted values in the same row or column of Table 3.

It remains to show that any such choice does indeed give a triple $\{a, b, c\}$ satisfying the definition of $\mathcal{P}$. From Lemma 2 and the proof of Lemma 1, it is sufficient to show that any ordered pair of highlighted values (x, y) taken from the same row or column of Table 3 (where x and y need not be distinct) does not equal $(E_1 k, E_0 k)$ for any $k(x) \in P_7$. We will assume (x, y) are taken from the same row; the argument when they are taken from the same column is similar. Suppose, for a contradiction, that $(E_1 k, E_0 k) = (x, y)$ where

$$x, y \in (E_1 j + \mathrm{col}(E_1 N_0)) \setminus (E_0 j + \mathrm{col}(E_0 N_1)) \,. \tag{7}$$

Then $x = E_1 j + E_1 n_0$ for some $n_0 \in N_0$. Comparison with $x = E_1 k$ shows that

$$k = j + n_0 + n_1 \quad \text{for some } n_1 \in N_1.$$

Applying E_0 to both sides we obtain $y = E_0 j + E_0 n_1$, which contradicts (7). □

We illustrate the proof of Proposition 4 by means of an example. Suppose we fix the element a to be $[00001]^T$. Take $j = [00111000]^T$ and $j' = [00001011]^T$ to satisfy $E_1 j = E_0 j' = a$. Then $E_0 j = [01000]^T$ and $E_1 j' = [11011]^T$. From Table 3, we can express the forbidden offsets $E_0 j + \mathrm{col}(E_0 N_1)$ and $E_1 j' + \mathrm{col}(E_1 N_0)$ as $[11100]^T + \mathrm{col}(E_0 N_1)$ and $[11110]^T + \mathrm{col}(E_1 N_0)$ respectively. In other words, we exclude the 6 highlighted values $[00010]^T$, $[00111]^T$, $[01101]^T$, $[01000]^T$, $[01100]^T$ and $[01001]^T$ contained in the third column and second row of data values. This leaves the 5 highlighted values $[00100]^T$, $[11111]^T$, $[11010]^T$, $[11101]^T$ and $[11000]^T$ as possible values for b and c. It follows that the elements b and c occurring with a in a triple must belong either to $\{[00100]^T, [11101]^T, [11000]^T\}$ or to $\{[00100]^T, [11111]^T, [11010]^T\}$.

From Proposition 4 we determine the triples $\{a, b, c\}$ of $\mathcal{P}$ to be all sets of three distinct elements taken from a single row of Table 4. By construction, each of these rows has the form

$$\{u, u + [00101]^T, v, v + [00101]^T\}. \tag{8}$$

5.1.2 Determination of $\mathcal{P}'$

In this subsection we determine the possible triples (α, β, γ) of $\mathcal{P}'$, as specified in (P2). The analysis is similar to that for the triples $\{a, b, c\}$ of $\mathcal{P}$, so we will simply state the results without proof.

Lemma 5 *No vector contained in* $\mathrm{col}(E_7 N_6)$, $\mathrm{col}(E_6 N_7)$ *or* $\mathrm{col}(E_6 N_{67})$ *can appear in a triple of* $\mathcal{P}'$.

The matrices E_7N_6, E_6N_7, and E_6N_{67} each have rank 3:

$$E_7N_6 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}, E_6N_7 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, E_6N_{67} = E_7N_{67} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix}.$$

Lemma 6 *If $\{\alpha, \beta, \gamma\}$ is a triple in $\mathcal{P}'$, and $\alpha+[01000]^T \notin \{0, \beta, \gamma\}$, then $\{\alpha+[01000]^T, \beta, \gamma\}$ is a triple in $\mathcal{P}'$.*

Table 5 is a 4×4 arrangement of the 32 possible 5-bit induced data error values, with values that differ by $[01000]^T$ grouped in pairs. Offsets of $\mathrm{col}(E_7N_6)$ appear horizontally, offsets of $\mathrm{col}(E_6N_7)$ appear vertically, and the main diagonal contains $\mathrm{col}(E_6N_{67})$.

Lemma 7 *For any 5-bit vectors h and h', there is a 5-bit vector h'' for which*

$$(h + \mathrm{col}(E_6N_7)) \cap (h' + \mathrm{col}(E_7N_6)) = \{h'', h'' + [01000]^T\}.$$

Similar statements hold for the intersection of offsets of $\mathrm{col}(E_6N_7)$ and $\mathrm{col}(E_6N_{67})$, and for the intersection of offsets of $\mathrm{col}(E_7N_6)$ and $\mathrm{col}(E_6N_{67})$.

Proposition 8 *The triples $\{\alpha, \beta, \gamma\}$ of $\mathcal{P}'$ comprise all sets of three distinct highlighted 5-bit values appearing in the same row or column of Table 5.*

The triples $\{\alpha, \beta, \gamma\}$ of $\mathcal{P}'$ comprise all sets of three distinct elements taken from a single row of Table 6. Each of these rows has the form

$$\{\mu, \mu + [01000]^T, \nu, \nu + [01000]^T\}. \tag{9}$$

5.1.3 Code selection as specified in (P3)

In this subsection we complete the analysis of configurations A and C by selecting a 5-bit/6-bit code mapping as specified in (P3). In fact we shall identify a large set of mappings, all of which deal with configurations A and C, from which we can select one in Section 5.2 that also deals with configuration B. In contrast to the study of linear coding, here we are not concerned with minimising the weight of the induced data errors but rather with ensuring that the sets M_1 and L_1 are small and controllable: specifically, no larger than some set $\{a, b, c\} \in \mathcal{P}$ and some set $\{\alpha, \beta, \gamma\} \in \mathcal{P}'$ respectively. We shall build up the 5-bit/6-bit mapping in stages, keeping track of M_1 and L_1 as each group of codewords is assigned.

Figure 3 shows a template assignment of 4 balanced and 4 unbalanced code words to 6 data words, with the value of x yet to be specified. The associated graph shows the induced data errors arising from change of the most significant code bit position (left-pointing arcs) and least significant code bit position (right-pointing arcs) of the codewords. We see that the entries of M_1 and L_1 resulting just from this template are $\{a, b\}$ and $\{\alpha, \beta\}$ respectively.

For example, change of the least significant bit position of the codeword 101100 results in the codeword 101101, which induces the data error $(x + a) + (x + a + \alpha) = \alpha$; this is represented by a right-pointing arc labelled α that connects the vertices associated with these two codewords.

Starting from the codeword 001100 in Figure 3, there is a path to the codeword 101101 that induces the data error $a + \alpha$; and starting from the codeword 110011, there is a path to the codeword 010010 that induces the data error $b + \beta$. Since the initial two codewords of these paths are both assigned to the same data word (x) and the final two codewords are both assigned to the same data word, for consistency we must impose the constraint $a + \alpha = b + \beta$, or equivalently

$$a + b = \alpha + \beta. \tag{10}$$

Also we can interchange the data word labels $x + a$, $x + b$, or the data word labels $x + \alpha$, $x + \beta$, or both, without changing the current sets M_1 and L_1. This allows additional freedom of choice for Section 5.2 and is represented in Figure 3 by arcs connecting these pairs of data words.

Assume that

$$\text{the } 5 \times 3 \text{ matrix } [a; \alpha; b] \text{ has rank } 3, \tag{11}$$

and write G for the eight data words of $\mathrm{col}([a; \alpha; b])$. Taking account of (10), the six data words of Figure 3 can then be represented as $(x + G) \setminus \{x + a + b, x + \alpha + b\}$, so we have assigned six of the eight data words contained in the offset $x + G$ of G. Assume further that

$$\text{the } 5 \times 5 \text{ matrix } [a; \alpha; b; c; \gamma] \text{ has rank } 5, \tag{12}$$

so that we can consider the 32 5-bit data words to comprise four offsets of G with offset representatives x_0, x_1, x_2 and x_3, where:

$$\left.\begin{array}{rcl} x_0 & \in & G, \\ x_1 & \in & c + G, \\ x_2 & \in & \gamma + G, \\ x_3 & \in & c + \gamma + G. \end{array}\right\} \tag{13}$$

We can then apply the template assignment of Figure 3 to the offsets whose representatives are x_0, x_1 and x_2, in each case assigning codewords to six out of the eight data words of the offset. Figure 4 shows this assignment, where the most and least significant bit of each set of eight codewords follows the pattern of the template while the central four bits always have weight 2. This assigns 24 codewords (all of whose central four bits have weight 2) to 18 data words while restricting the current sets M_1 and L_1 to $\{a, b\}$ and $\{\alpha, \beta\}$ respectively.

Figure 5 shows an assignment of a further 12 codewords to all 8 data words of the offset whose representative is x_3. This assignment does not enlarge the sets M_1 and L_1, and admits the transpositions of data words indicated by arcs.

Before assigning codewords to the remaining six data words, we shall apply the constraint (10), followed by the constraint (12) (which implies (11)). From (8), each row of Table 4 gives rise to three possible values of $a + b$, independently of which distinct three

of the four row entries are chosen for the triple $\{a, b, c\}$. Similarly, from (9), each row of Table 6 gives rise to three possible values of $\alpha + \beta$. Comparison of these values for $a + b$ and $\alpha + \beta$ yields four possible pairings of a row from Table 4 with a row from Table 6 such that (10) is satisfied, as shown in Table 7. From each row of Table 7, we form a triple $\{a, b, c\} \in \mathcal{P}$ by taking any three distinct elements from the first set of four data values. This is paired with a triple $\{\alpha, \beta, \gamma\} \in \mathcal{P}'$ formed by taking any three distinct elements from the second set of four values in that row; the values of c and γ are thereby determined. Constraint (12) removes the second row of Table 7 from consideration, leaving $3 \cdot 4 \cdot 4 = 48$ possible sets of values $\{a, b, c, \alpha, \beta, \gamma\}$.

We now assign eight codewords to the remaining six data words, and thereby fix the values of x_0, x_1 and x_2. By translation of all the data words we can take x_0 to be 0. The ten unassigned codewords are

001110, 110001, 011100, 100011, 010001, 101110, 100010, 011101, 100001, 011110,

of which one codeword of weight 2 and one of weight 4 must be excluded (see Section 3). Of these codewords, the pairs

$$(010001, 110001) \text{ and } (001110, 101110) \tag{14}$$

differ only in the most significant bit position, while the pairs

$$(100010, 100011) \text{ and } (011100, 011101) \tag{15}$$

differ only in the least significant bit position. The six unassigned data words are

$$a + b, \quad \alpha + b, \quad x_1 + a + b, \quad x_1 + \alpha + b, \quad x_2 + a + b, \quad x_2 + \alpha + b. \tag{16}$$

Let d_1 and d_2 be distinct data words in (16). From (13), the only value of $d_1 + d_2$ lying in G is $a + \alpha$ and the only values of $d_1 + d_2$ lying in $c + G$ are x_1 and $x_1 + a + \alpha$. It follows from (P3) that if either the first or second pair of codewords in (14) is contained in the code then $d_1 + d_2 = c$ for the corresponding data words d_1, d_2 and

$$x_1 = c \text{ or } c + a + \alpha. \tag{17}$$

Similarly, if either the first or second pair of codewords in (15) is contained in the code then $d_1 + d_2 = \gamma$ and

$$x_2 = \gamma \text{ or } \gamma + a + \alpha. \tag{18}$$

Until this point all unbalanced codeword pairs have been chosen to be complements of each other, for convenience of implementation. Suppose, for a contradiction, that we can satisfy (P3) while maintaining this property. Then either all four codewords of (14) or all four codewords of (15) will be retained in the code (or both); suppose the former. By assumption the retained codewords 010001 and 101110 must both be assigned to some data word d_1 of (16). The codewords 110001 and 001110 must be assigned to distinct data words d_2, d_3 of (16), but then by the reasoning leading to (17) we must have $d_1 + d_2 = c$ and $d_1 + d_3 = c$ which is a contradiction. (One might attempt to avoid this contradiction

by relaxing (P3) to allow $M_1 = \{a, b, c, d\}$, where the set $\{a, b, c, d\}$ comprises all four entries of one of the rows of Table 4 and then, from (8),

$$c + d = a + b. \tag{19}$$

But this would result in $d_1 + d_2 = c$ and $d_1 + d_3 = d$, which from (19) would imply $d_2 + d_3 = a + b$, which is not possible from (16).) The argument when the four codewords of (15) are retained instead of those of (14) is similar (and there is likewise no advantage in allowing $L_1 = \{\alpha, \beta, \gamma, \delta\}$ in (P3) for some row $\{\alpha, \beta, \gamma, \delta\}$ of Table 6).

Therefore the code must contain at least one pair of unbalanced codewords that are not complements of each other. Figure 6 shows an assignment of eight codewords to two possible listings of the remaining six data words (shown on the left and on the right of the diagram), both of which will be considered in Section 5.2. The assignment uses $x_1 = c$ and $x_2 = \gamma$, in accordance with (17) and (18), to give final sets $M_1 = \{a, b, c\}$ and $L_1 = \{\alpha, \beta, \gamma\}$ satisfying (P3), and excludes the codewords 010001 and 011101. It contains only one pair of unbalanced codewords that are not complements of each other. The boxed data words in Figure 6 can be arbitrarily arranged since the corresponding codewords do not map to any other valid codewords under change of the most significant or least significant bit.

Note that we can make alternative assignments to that of Figure 6 that satisfy (P3) (though these will not be considered in Section 5.2), for example we can:

- set $x_1 = c + a + \alpha$ and interchange the labels $x_1 + a + b$ and $x_1 + \alpha + b$ in Figure 6, or set $x_2 = c + a + \alpha$ and interchange the labels $x_2 + a + b$ and $x_2 + \alpha + b$, or both
- exclude the codewords 100010 and 101110 and have only one non-complementary unbalanced codeword pair; this is equivalent to the assignment of Figures 4, 5 and 6 under complementation of all the codewords of the 5-bit/6-bit mapping
- exclude the codewords 010001 and 101110 from (14) and arrange to have final sets $M_1 = \{a, b\}$ and $L_1 = \{\alpha, \beta, \gamma\}$; however this results in two non-complementary unbalanced codeword pairs. Similarly we can exclude codewords 100010 and 011101 and have final sets $M_1 = \{a, b, c\}$ and $L_1 = \{\alpha, \beta\}$.

5.2 Configuration B

In Section 5.1 we used linear algebra and combinatorial reasoning to control the sets M_1 and L_1 and so deal with configurations C and A respectively of Table 2. In this section we use computer search to select one of the large class of 5-bit/6-bit mappings already identified, in order to deal with configuration B. Under configuration B, only the two most significant bit positions of C_1 and C_0, and only the two least significant bit positions of C_7 and C_6, can change. Our objective here is to arrange for the sets M_2 and L_2 (as defined in Section 5.1) *jointly* to avoid all 255 undetectable errors in S — whereas for configurations C and A we relied on the sets M_1 and L_1 individually.

We begin by reviewing the set of 5-bit/6-bit mappings previously identified in Section 5.1. There are 48 possible sets of values $\{a, b, c, \alpha, \beta, \gamma\}$, represented by the first,

third and fourth rows of Table 7. For each of these, there are: 2^6 possible rearrangements of data words in Figure 4; 2^2 possible rearrangements of data words in Figure 5; 2 possible rearrangements of data words arising from interchange of a with b; 3! possible rearrangements of data words for each of the 2 listings (left-hand and right-hand) in Figure 6; and 8 possible choices of $x_3 \in c+\gamma+G$. In addition, for each of the above possibilities there are 3! possible further rearrangements not previously mentioned, arising from permutation of the labels x_0, x_1, x_2 in Figure 4. (There is no need to consider interchange of α with β because it is equivalent to a combination of other rearrangements.) This gives a total of $2^{19} \cdot 3^3 > 10^7$ code mappings across which the sets M_2 and L_2 (and hence the behaviour under configuration B) can vary.

We deal with configuration B as follows:

(P4) Determine which of the $2^{19} \cdot 3^3$ identified 5-bit/6-bit mappings, all of which deal with configurations A and C, also have the property that

$$E_1 j, E_0 j \in M_2 \cup \{0\} \text{ and } E_7 j, E_6 j \in L_2 \cup \{0\} \tag{20}$$

holds for $j(x) \in P_7$ only when $j(x) = 0$.

By computer search, (P4) gives rise to 192 code mappings associated with 5 of the 48 sets of values $\{a, b, c, \alpha, \beta, \gamma\}$, as summarised in Table 8. The mapping chosen for the IEEE 802.12 code presented in Table 1 has

$$a = [11101]^T, \quad b = [00001]^T, \quad c = [00100]^T, \quad \alpha = [00011]^T, \quad \beta = [11111]^T, \quad \gamma = [01011]^T.$$

The values $(x_0, x_1, x_2) = (0, c, \gamma)$ are assigned to the three offsets of Figure 4 without permutation. The value x_3 is set as $c+\gamma+b$. The members of data word pairs (x_0+a, x_0+b), (x_2+a, x_2+b) and $(x_3+\alpha, x_3+\beta)$ are interchanged. The right-hand listing of data words in Figure 6 is used, with the boxed data words occurring without permutation. These choices result in the sets

$$\begin{aligned} M_2 &= \{[00001]^T, [00100]^T, [01001]^T, [01111]^T, [10001]^T, [10100]^T, [10101]^T, \\ &\quad [11000]^T, [11001]^T, [11100]^T, [11101]^T\}, \end{aligned} \tag{21}$$

$$\begin{aligned} L_2 &= \{[00011]^T, [00100]^T, [00101]^T, [00110]^T, [00111]^T, [01001]^T, [01011]^T, \\ &\quad [10101]^T, [10110]^T, [11000]^T, [11011]^T, [11100]^T [11110]^T, [11111]^T\}. \end{aligned} \tag{22}$$

It is possible to verify by hand that the sets (21) and (22) are in accordance with (P4), as we now outline. From (20) and (21) there are 12 values of $E_0 j$ to be checked; for example consider $E_0 j = [00001]^T$. By inspection of the explicit representation of E_0 in Section 5.1 we see that $E_0[00001011]^T = [00001]^T$, therefore

$$j \in [00001011]^T + \mathrm{col}(N_0). \tag{23}$$

Applying E_1 to (23) we find that $E_1 j \in [11011]^T + \mathrm{col}(E_1 N_0)$, whereas according to (20) we impose $E_1 j \in M_2 \cup \{0\}$. It follows that $E_1 j = [01001]^T$ or $[10101]^T$. If the former, then comparison with (23) forces $j = [00001011]^T + [11000000]^T = [11001011]^T$ so that

$E_6 j = [00001]^T \notin L_2 \cup \{0\}$; if the latter, then $j = [00001011]^T + [01000000]^T = [01001011]^T$ so that $E_6 j = [01000]^T \notin L_2 \cup \{0\}$. In either case the result is in accordance with (P4). The checking for the other 11 possible values for $E_0 j$ is similar.

We mention in closing that the subset of the 5-bit/6-bit mappings of Section 5.1 that also deal with configuration B could have more than the 192 elements described above. The reason is that, for ease of computer implementation and hand checking, (P4) does not attempt to take advantage of the alternation rule (strengthened by the use of alternative end delimiters as described in Section 4). For example, (P4) requires the CRC to detect a 7-bit error burst having configuration B that corrupts codeword C_3 from weight 3 to weight 4 without changing the weight of codeword C_7, whereas such an error burst is guaranteed to be detected by the alternation rule.

6 Conclusion

We have presented the first complete explanation of the design principles of the IEEE 802.12 coding scheme. This code has near-perfect DC balance and small run length, and allows the detection within an encoded data packet of any three single-bit errors or any 7-bit error burst that arbitrarily corrupts codewords carried on four parallel channels. A major part of the design effort was directed towards burst error detection, combining: geometrical insight (offset transmission of codewords); linear algebra and combinatorial reasoning (to deal with configurations A and C); and computer search (to deal with configuration B).

Acknowledgements

We are grateful to David Cunningham and Pat Thaler for their assistance in bringing this coding scheme to standardisation in IEEE 802.12.

References

[1] G. Watson, A. Albrecht, J. Curcio, D. Dove, S. Goody, J. Grinham, M. Spratt, and P. Thaler, "The Demand Priority MAC protocol," *IEEE Network*, pp. 28–34, Jan/Feb 1995.

[2] LAN/MAN Standards Committee of the IEEE Computer Society, *IEEE Std* 802.12 – 1995*: Demand Priority Access Method, Physical Layer and Repeater Specification for* 100 *Mb/s Operation*. New York: IEEE, Nov 1995.

[3] A. Albrecht and P. Thaler, "Introduction to 100VG-AnyLAN and the IEEE 802.12 Local Area Network standard," *Hewlett-Packard Journal*, vol. **46**, pp. 6–12, August 1995.

[4] E. Rabinovitch, "(Barely) managing ATM," *SunWorld Online*, vol. **10**, Aug 1996, `<http://sunsite.uakom.sk/sunworldonline/swol-08-1996/swol-08-atm.html>`.

[5] LAN/MAN Standards Committee of the IEEE Computer Society, *IEEE Std* 802.3 – 2002*: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications*. New York: IEEE, Mar 2002.

[6] A. Coles, D. Cunningham, J. Curcio, Jr, D. Dove, and S. Methley, "Physical signalling in 100VG-AnyLAN," *Hewlett-Packard Journal*, vol. **46**, pp. 18–26, August 1995.

[7] S. Crouch and J. Jedwab, "Coding in 100VG-AnyLAN," *Hewlett-Packard Journal*, vol. **46**, pp. 27–32, August 1995.

[8] Local and Metropolitan Area Networks Standards Committee, *Functional Requirements, IEEE Project 802*. IEEE, draft 6.10, revised 12 November 1991.

[9] R. Jain, "Error characteristics of fiber distributed data interface (FDDI)," *IEEE Trans. Comm.*, vol. **38**, pp. 1244–1252, 1990.

Data word	Weight 3 codeword
00001	101100
00011	001101
00101	010101
00110	001110
00111	001011
01000	000111
01001	100011
01010	100110
01101	011010
01111	101001
10001	100101
10011	010110
10100	111000
10110	011001
11000	110001
11001	101010
11011	110100
11100	011100
11101	010011
11111	110010

Data word	Weight 2 codeword	Weight 4 codeword
00000	001100	110011
00010	100010	101110
00100	001010	110101
01011	000110	111001
01100	101000	010111
01110	100100	011011
10000	000101	111010
10010	001001	110110
10101	011000	100111
10111	100001	011110
11010	010100	101011
11110	010010	101101

Table 1: The IEEE 802.12 5-bit/6-bit transmission code

Configuration	# most significant positions in C_1 and C_0	# least significant positions in C_7 and C_6
A	3	1
B	2	2
C	1	3

Table 2: Configurations A, B and C for a 7-bit error burst

		Offset of col(E_0N_1)							
		00000		01110		11100		10010	
Offset of col(E_1N_0)	00000	00000,	00101	01110,	01011	11100,	11001	10010,	10111
	11110	11110,	11011	10000,	10101	**00010**,	**00111**	**01100**,	**01001**
	01111	01111,	01010	**00001**,	**00100**	10011,	10110	**11101**,	**11000**
	10001	10001,	10100	**11111**,	**11010**	**01101**,	**01000**	00011,	00110

Table 3: Transpose of induced data error vectors, arranged horizontally by offsets of col(E_1N_0) and vertically by offsets of col(E_1N_0). The main diagonal is col(E_0N_{01}).

Set #				
1	00010	00111	01100	01001
2	00001	00100	11101	11000
3	11111	11010	01101	01000
4	00001	00100	11111	11010
5	00010	00111	01101	01000
6	01100	01001	11101	11000

Table 4: The sets of four transposed data error vectors from which the triples $\{a, b, c\}$ of $\mathcal{P}$ are derived

		Offset of col(E_6N_7)							
		00000		00001		10010		10011	
Offset of col(E_7N_6)	00000	00000,	01000	00001,	01001	10010,	11010	10011,	11011
	10000	10000,	11000	10001,	11001	**00010**,	**01010**	**00011**,	**01011**
	00100	00100,	01100	**00101**,	**01101**	10110,	11110	**10111**,	**11111**
	10100	10100,	11100	**10101**,	**11101**	**00110**,	**01110**	00111,	01111

Table 5: Transpose of induced data error vectors, arranged horizontally by offsets of col(E_7N_6) and vertically by offsets of col(E_7N_6). The main diagonal is col(E_6N_{67}).

Set #				
1	00010	01010	00011	01011
2	00101	01101	10111	11111
3	10101	11101	00110	01110
4	00101	01101	10101	11101
5	00010	01010	00110	01110
6	00011	01011	10111	11111

Table 6: The sets of four transposed data error vectors from which the triples $\{\alpha, \beta, \gamma\}$ of $\mathcal{P}'$ are derived

Set # for $\mathcal{P}$					$a+b=$ $\alpha+\beta$	Set # for $\mathcal{P}'$				
2	00001	00100	11101	11000	11100	6	00011	01011	10111	11111
3	11111	11010	01101	01000	10010	2	00101	01101	10111	11111
4	00001	00100	11111	11010	11011	3	10101	11101	00110	01110
6	01100	01001	11101	11000	10100	6	00011	01011	10111	11111

Table 7: Sets of four transposed data error vectors for triples $\{a, b, c\} \in \mathcal{P}$ and $\{\alpha, \beta, \gamma\} \in \mathcal{P}'$ that admit a common value for $a+b$ and $\alpha+\beta$

# code mappings	$\{a, b\}$	c	$\{\alpha, \beta\}$	γ	$a+b=\alpha+\beta$
64	$\{00001, 11101\}$	00100	$\{00011, 11111\}$	01011	11100
64	$\{00001, 11101\}$	11000	$\{00011, 11111\}$	10111	11100
32	$\{00100, 11000\}$	11101	$\{00011, 11111\}$	10111	11100
16	$\{00100, 11000\}$	11101	$\{01011, 10111\}$	11111	11100
16	$\{00100, 11111\}$	00001	$\{01110, 10101\}$	00110	11011

Table 8: Numbers of code mappings arising from (P4), and the associated transposed data error vectors

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
$\cdots$	01111	00110	10001	11110	10111	01001	00111	01011	$\cdots$

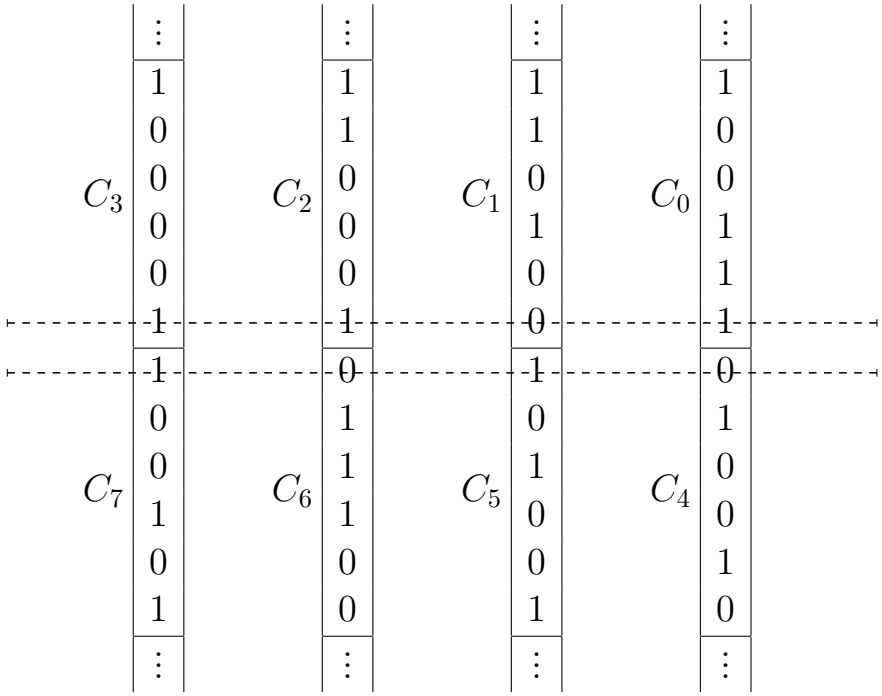

Figure 1: Transmission of codewords over four parallel conductors

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
$\cdots$	01111	00110	10001	11110	10111	01001	00111	01011	$\cdots$

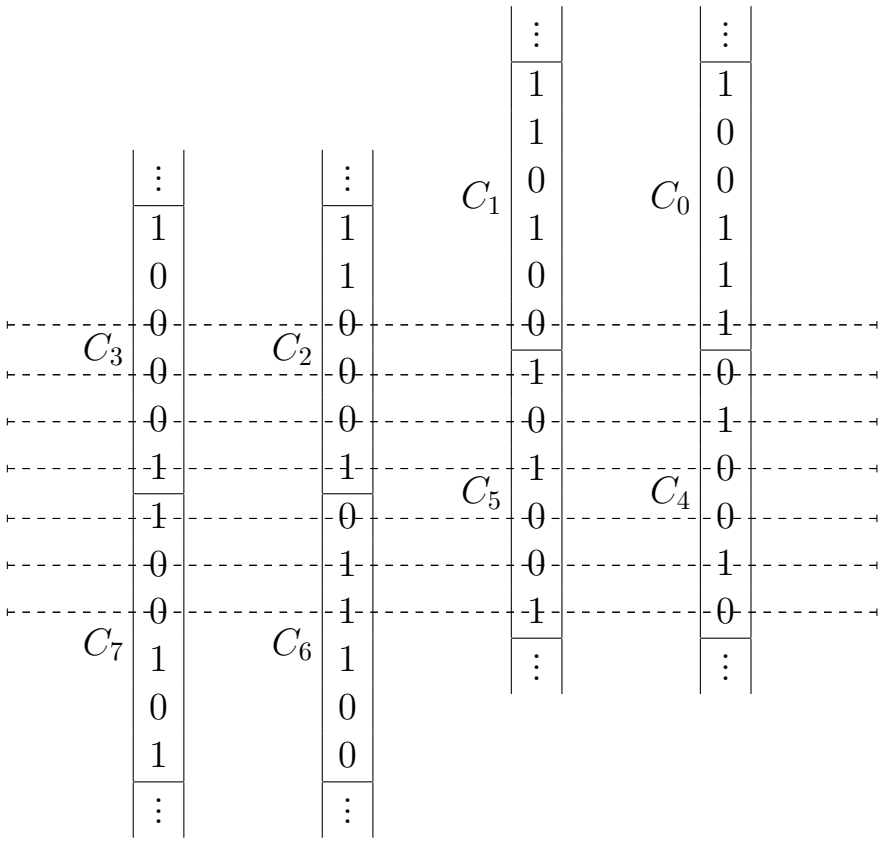

Figure 2: Offset transmission of codewords

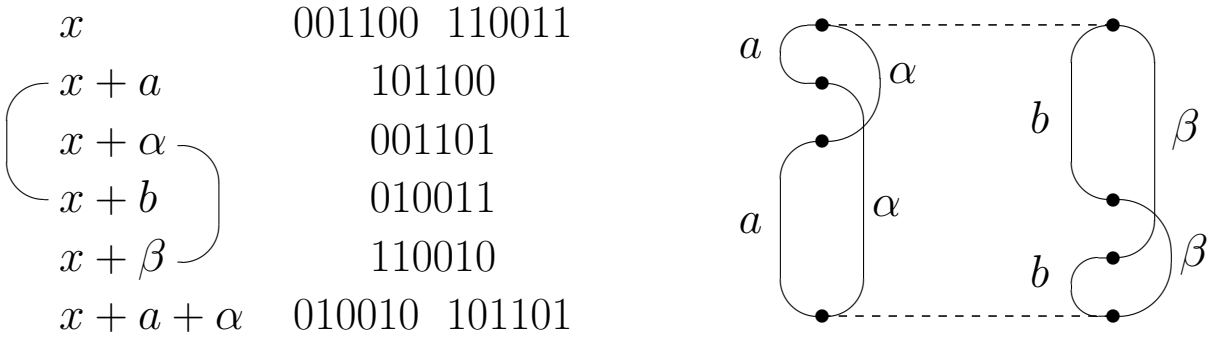

Figure 3: Template assignment of codewords to data words

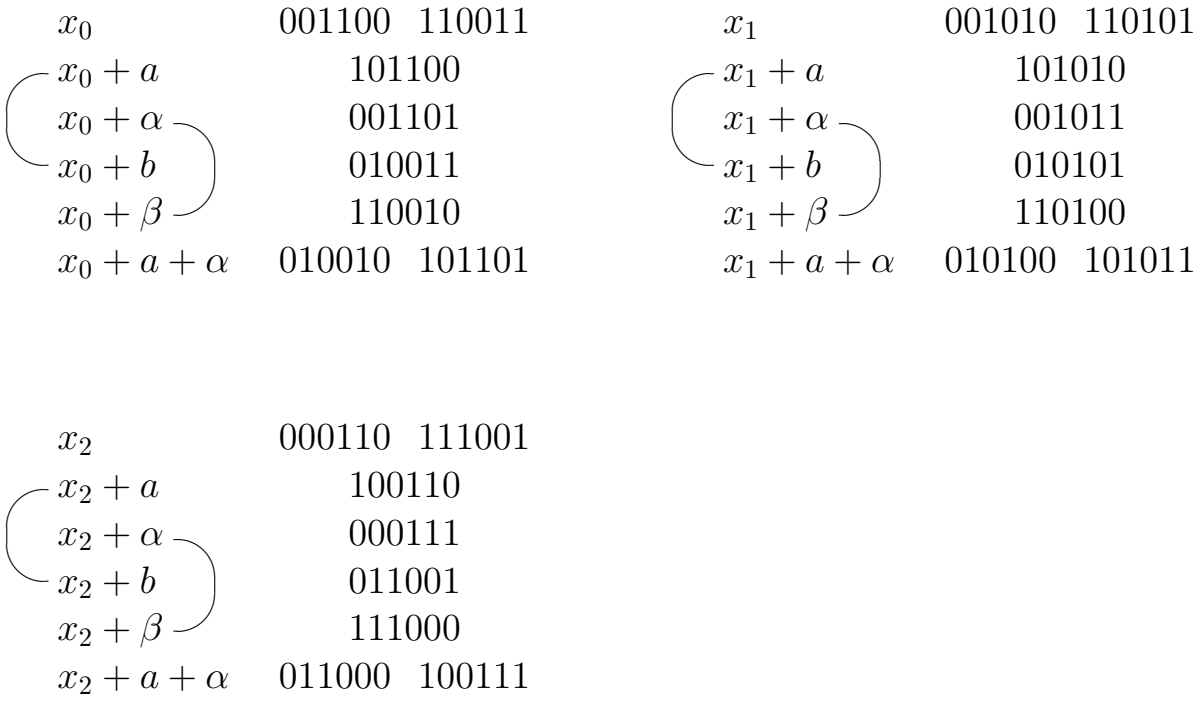

Figure 4: Template assignment applied to three of the four offsets

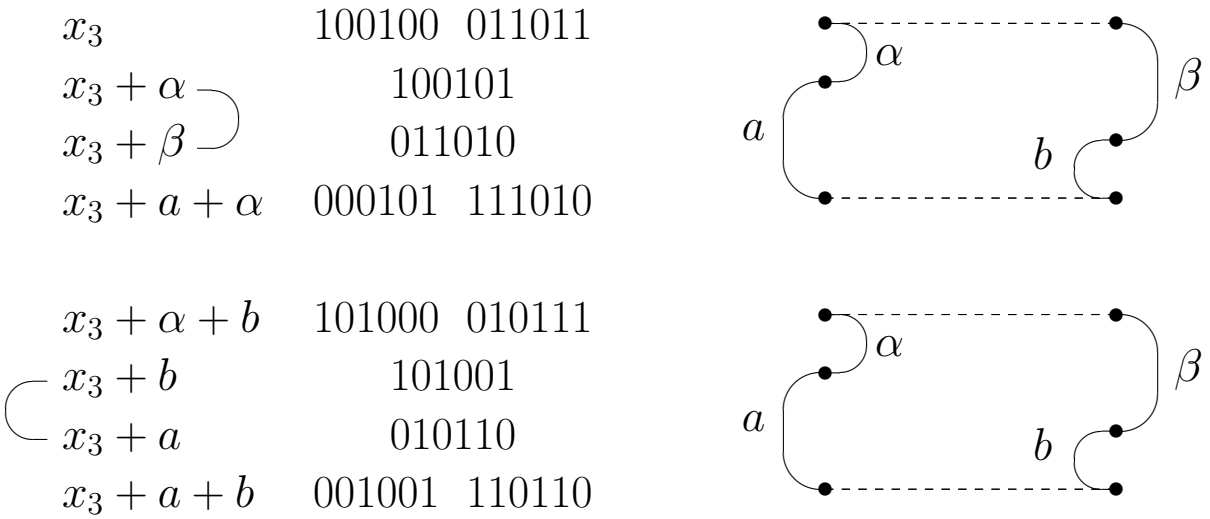

Figure 5: Code structure for fourth offset

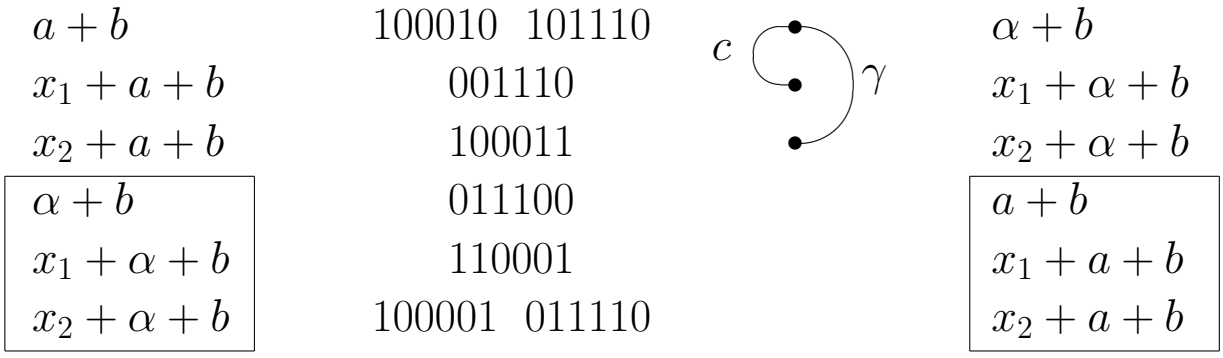

Figure 6: Code structure for remaining six data words