sciendo

Journal of ELECTRICAL ENGINEERING, VOL 69 (2018), NO3, 245–249

On run-length limited error control codes constructed from binary product codes

Peter Farkaš[*,], Tomáš Janvars[*], Katarína Farkašová[**], Eugen Ružický[**]**

In this paper it is presented that run-length limited error control codes could be constructed from any two or more-dimensional binary product codes as long as at least one of the one-dimensional binary component codes is or can be converted to a run-length limited error control code. The advantages of this construction are as follows: It does not require any additional redundancy except that which is already contained in the original error control code and that the encoding and decoding procedure used for the underlying error control code do not to be changed.

K e y w o r d s: constrained code, run-length limited error control code, binary product code, control matrix, modifier, more-dimensional code

1 Introduction

Run-length limited (RLL) codes are codes with constraints on lengths of identical symbol runs in sequences of their consecutive codewords. These constraints are invoked by practical system requirements [1]. RLL codes have applications mainly in communications systems and storage systems. Their importance was underlined in 2017 by the fact that the highest IEEE award (The Medal of Honor) was given to K. A. S. Immink. He developed and improved many constrained codes for different optical storage systems [2]. These were recognized as "pioneering contributions to video, audio and data recording technology, including compact disc, DVD and Blu-ray." An overview of RLL codes could be found in his book [3].

Immink and others used RLL Codes in practical applications together with Error Control Codes (ECC) connected in cascade. The reason was that RLL codes are not resilient against errors. They can, in most cases, be corrected by ECCs. The consequence of using two such kinds of codes in cascade was that redundancy was added into the encoding stream twice, once for the ECC and once for the RLL code. The other problem with this arrangement is that the answer to the question of which code should be the outer and which the inner one cannot be answered satisfactory [4]. One way to overcome these difficulties is to use codes which possess both properties (RLL and ECC). Such codes are denoted in literature as Run-Length Limited Error Control Codes (RLL-ECCs). A short overview of RLL-ECCs development and the current state of the art in their construction could be found in recent papers [4, 5], which also form the main background knowledge useful for understanding this short communication.

The novelty of this paper is a proposition that some binary product codes (BPCs) could be converted into RLL-ECCs. Namely such that contain at least one component ECC which could be transformed to an RLL-ECC. It is a generalization of the observation presented in [6] that binary product codes obtained from single parity check codes (SPCs) could be converted to RLL-ECCs when at least one component code has an even codeword length.

2 Basic introduction to product codes

Product Codes are usually constructed from two or more linear block ECCs. A linear block ECC C is defined as a k-dimensional subspace of an n – dimensional vector space over a finite field $GF(q)$. One way it could be specified is via a $k \times n$ generator matrix $\mathbf{G}$. In its rows it contains k linearly independent vectors. Each codeword $\mathbf{c} \in C$ could be obtained as a linear combination of these row-vectors from $\mathbf{G}$. The basic parameters of C are usually given in compact form using a triplet $[n, k, d_m]$ where the numbers denote the codeword length, code dimension and code distance respectively.

Another possibility for defining a linear ECC is to use a parity check matrix. For example, low density parity check (LDPC) codes are usually defined using it as follows

$$C = \{\mathbf{c}; \quad \mathbf{c}\mathbf{H}^\top = \mathbf{0}\} \tag{1}$$

It is a special case of a control matrix $\mathbf{H}_c$, which has to have linearly independent rows and for which

$$\boldsymbol{c}\mathbf{H}_c^\top = \mathbf{0}\,. \tag{2}$$

In other words, each row of $\mathbf{H}_c$ describes one control equation, which each codeword must fulfil.

* Institute of Multimedia ICT, Faculty of Electrical Engineering and Information Technology, Bratislava, Slovakia p.farkas@ieee.org,
** Institute of Applied informatics, Faculty of Informatics Pan European University, Bratislava, Slovakia, eugen.ruzicky@paneurouni.com

DOI: 10.2478/jee-2018–0033, Print (till 2015) ISSN 1335-3632, On-line ISSN 1339-309X

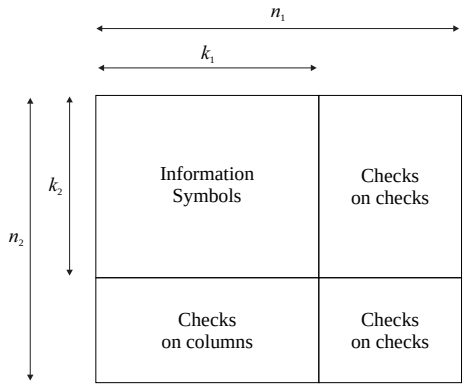

Fig. 1. 2-D product code

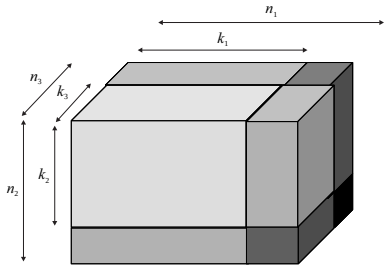

Fig. 2. 3-D product code

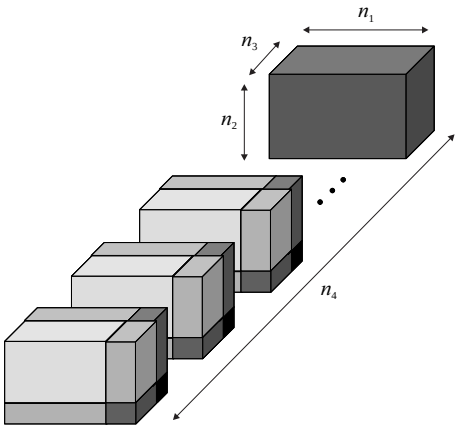

Fig. 3. 4-D product code

2.1 Two-dimensional product codes

Two-dimensional product codes (2-D PCs) were proposed in [8]. They are composed of two linear block codes C_1 and C_2 with parameters $[n_1, k_1, d_{m1}]$ and $[n_2, k_2, d_{m2}]$ respectively.

A 2-D PC is an $[n_1 \times n_2, k_1 \times k_2, d_{m1} \times d_{m2}]$ code illustrated in Fig. 1 via a table in which each row corresponds to a codeword $\mathbf{c}_1 \in C_1$ and each column to a codeword $\mathbf{c}_2 \in C_2$.

The main advantage of 2-D PCs is that decoding their relatively long codewords could be performed using less complex subroutines for decoding the shorter row and column codewords from the component codes [1].

2.2 Multi-dimensional product codes

The 2-D PC could be generalized into more dimensions as illustrated, for example, for 3-D and 4-D in Fig. 2. and Fig. 3 respectively. For the M-D PC, the basic parameters are given by the following triplet

$$[N, K, d_m] = \big[n_1 \times n_2 \times \ldots, \times n_M, \;\; k_1 \times k_2 \times \ldots, k_M, \; d_{m1} \times d_{m2} \times \ldots, \times d_{mM}\big]. \quad (3)$$

In [9] single parity check (SPC) codes were used to obtain multidimensional PCs (M-D PCs). It was also shown there that such codes have surprisingly good error control capabilities with respect to their relative simplicity and not very demanding decoding complexity. Similar results were also obtained independently in [10], where residual bit error rate after transmission with binary antipodal signaling through an AWGN channel and decoding using Pyndiah algorithm [11] was analyzed in dependency on signal to noise ratio for up to 6-D PCs obtained from SPCs.

3 Product code's control and parity check matrices

In case that the codes are binary, the graphical representation of M-D PCs, illustrated for 2-D, 3-D and 4-D in Figs. 1, 2 and 3 respectively, could be represented using a control matrix $\mathbf{H}_c$ from which the parity check matrix $\mathbf{H}$ could be obtained in a the following form

$$\mathbf{H} = \begin{bmatrix} h_{1,1} & h_{1,2} & \ldots & h_{1,N} \\ h_{2,1} & h_{2,2} & \ldots & h_{2,N} \\ \vdots & \vdots & \vdots & \vdots \\ h_{(N-K),1} & h_{(N-K),2} & \ldots & h_{(N-K),N} \end{bmatrix}. \quad (4)$$

The approach how it could be done is most easily understandable explaining it on a 2-D PC.

The first step is to choose a bijective mapping between each coordinate of a codeword from the 2-D PC and each column of matrix $\mathbf{H}_c$. For example an index $j = 1, 2, \ldots, N$, (where in this case $N = n_1 \times n_2$) could be used for indexing $\mathbf{H}_c$. matrix columns and codeword symbols.

In the second step a bijective mapping between rows and columns of all control equations in the component codes, each represented by one row or one column in the 2-D table (illustrating the 2-D PC) and rows of matrix $\mathbf{H}_c$, has to be selected. The total number of control equations E_c, equivalent to a total number of rows in $\mathbf{H}_c$, is

$$E_c = n_2 \times (n_1 - k_1) + n_1 \times (n_2 - k_2). \quad (5)$$

c_1	c_2	c_3	c_4	c_5	c_6	c_7
c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}
c_{15}	c_{16}	c_{17}	c_{18}	c_{19}	c_{20}	c_{21}
c_{22}	c_{23}	c_{24}	c_{25}	c_{26}	c_{27}	c_{28}
c_{29}	c_{30}	c_{31}	c_{32}	c_{33}	c_{34}	c_{35}
c_{36}	c_{37}	c_{38}	c_{39}	c_{40}	c_{41}	c_{42}
c_{43}	c_{44}	c_{45}	c_{46}	c_{47}	c_{48}	c_{49}

Fig. 4. 2-D product code composed from two [7,4,3] Hamming codes

In the next step a symbol 1 or 0 is assigned to each element in matrix $\mathbf{H}_c$ following a simple rule. If there is a symbol with index i in row j in the table representing the 2-D PC participating in the control equation corresponding to the considered row, then the corresponding element of $\mathbf{H}_c$, is equal to 1 otherwise it is zero.

In case we also need the parity check matrix $\mathbf{H}$, then in the last step the linearly dependent rows in the control matrix $\mathbf{H}_c$, have to be identified and deleted.

For example, we will show how to get $\mathbf{H}_c$ for a 2-D PC illustrated in Fig. 4.

Let us suppose that it was obtained as a product of two identical $[7, 4, 3]$ Hamming codes, each with the following parity check matrix

$$\mathbf{H}_1 = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}, \tag{6}$$

which we can also express in a more compact form as

$$\mathbf{H}_1 = [\,\mathbf{h}_1 \quad \mathbf{h}_2 \quad \mathbf{h}_3 \quad \mathbf{h}_4 \quad \mathbf{h}_5 \quad \mathbf{h}_6 \quad \mathbf{h}_7\,], \tag{7}$$

where $\mathbf{h}_\iota, \iota = 1, \ldots, 7$ are columns of $\mathbf{H}_1$ in (6).
Let us denote a codeword of this code as

$$\mathbf{c} = [\,c_1, \quad c_2, \quad \ldots, \quad c_{49}\,], \tag{8}$$

then the control matrix is

$$\mathbf{H}_c = \begin{bmatrix} \mathbf{H}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{H}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{H}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{H}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{H}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{H}_1 \\ \mathbf{H}_{11} & \mathbf{H}_{12} & \mathbf{H}_{13} & \mathbf{H}_{14} & \mathbf{H}_{15} & \mathbf{H}_{16} & \mathbf{H}_{17} \end{bmatrix}, \tag{9}$$

where

$$\mathbf{0} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{10}$$

and

$$\mathbf{H}_{1\iota} = \begin{bmatrix} \mathbf{h}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota \\ \mathbf{0}_\iota & \mathbf{h}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota \\ \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{h}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota \\ \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{h}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota \\ \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{h}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota \\ \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{h}_\iota & \mathbf{0}_\iota \\ \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{0}_\iota & \mathbf{h}_\iota \end{bmatrix} \tag{11}$$

in which

$$\mathbf{0}_\iota = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{12}$$

The matrix $\mathbf{H}_c$ is a 42×49-matrix (9). Observing its top 21 rows, it is obvious that the 49 resulting PC's codeword coordinates could be distributed into 7 disjunctive (concatenated) sets corresponding to the different codewords of the component code. The union of these sets contains all 49 coordinates of the codeword. This observation will be important in later explanations in this paper.

For a M-D PC, which is composed of component ECCs with parameters

$$[n_1, k_1, d_{m1}], [n_2, k_2, d_{m2}], \ldots, [n_M, k_M, d_{M1}],$$

the approach is similar. Let us suppose that one of these codes is the RLL-ECC and it has parameters: $[n_i, k_i, d_{iM}]$. We can start the construction by first writing all rows corresponding to control equations of this code into the rows of $\mathbf{H}_c$ starting from the top. This code is used in the same direction in the M-D code exactly $(n_i - k_i).n_1 \times n_2 \times \ldots \times n_{(i-1)} \times n_{(i+1)} \times \ldots \times n_M)$ "times in a parallel manner".

In the next step we can continue with writing rows corresponding to appropriately interleaved columns of the control matrix ECC $\mathbf{H}_1, \mathbf{H}_2, \ldots \mathbf{H}_{(i-1)}, \mathbf{H}_{(i+1)}, \ldots, \mathbf{H}_M$ which we will denote $\Xi\{\mathbf{H}_{1,j}\}, \Xi\{\mathbf{H}_{2,j}\}, \ldots, \Xi\{\mathbf{H}_{M,j}\}$. (In order to save space we will also use $(n_i - k_i) \times n_i$ submatrices $\mathbf{0}_{i,j}$ composed of all zeros in the next text).

Applying this approach and notation we will get the control matrix in the following form

$$\mathbf{H}_c = \begin{bmatrix} \mathbf{H}_i & \mathbf{0}_{i,j} & \ldots & \mathbf{0}_{i,j} \\ \mathbf{0}_{i,j} & \mathbf{H}_i & \ldots & \mathbf{0}_{i,j} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{i,j} & \mathbf{0}_{i,j} & \ldots & \mathbf{H}_i \\ \Xi\{\mathbf{H}_{1,1}\} & \Xi\{\mathbf{H}_{1,2}\} & \ldots & \Xi\{\mathbf{H}_{1,N/n_1}\} \\ \Xi\{\mathbf{H}_{2,1}\} & \Xi\{\mathbf{H}_{2,2}\} & \ldots & \Xi\{\mathbf{H}_{2,N/n_2}\} \\ \vdots & \vdots & \vdots & \vdots \\ \Xi\{\mathbf{H}_{(i-1),1}\} & \Xi\{\mathbf{H}_{(i-1),2}\} & \ldots & \Xi\{\mathbf{H}_{(i-1),(N/n_{(i-1)})}\} \\ \Xi\{\mathbf{H}_{(i+1),1}\} & \Xi\{\mathbf{H}_{(i+1),2}\} & \ldots & \Xi\{\mathbf{H}_{(i+1),(N/n_{(i+1)})}\} \\ \Xi\{\mathbf{H}_{(i+2),1}\} & \Xi\{\mathbf{H}_{(i+2),2}\} & \ldots & \Xi\{\mathbf{H}_{(i+2),(N/n_{(i+2)})}\} \\ \vdots & \vdots & \vdots & \vdots \\ \Xi\{\mathbf{H}_{(M-1),1}\} & \Xi\{\mathbf{H}_{(M-1),2}\} & \ldots & \Xi\{\mathbf{H}_{(i+2),(N/n_{(M-1)})}\} \\ \Xi\{\mathbf{H}_{M,1}\} & \Xi\{\mathbf{H}_{M,2}\} & \ldots & \Xi\{\mathbf{H}_{M,(N/n_M)}\} \end{bmatrix} \tag{13}$$

The overall number of columns and rows in the control matrix in (13) is

$$N = (n_1 \times n_2 \times \cdots \times n_M)$$

and $\kappa =$

$$\begin{aligned}&(n_i - k_i)(n_1 \times n_2 \times \cdots \times n_{(i-1)} \times n_{(i+1)} \times \cdots \times n_M) + \\ &+(n_1 - k_1)(n_2 \times n_3 \cdots \times n_M) + \\ &+(n_2 - k_2)(n_1 \times n_3 \cdots \times n_M) + \\ &\cdots + (n_M - k_M)(n_1 \times n_2 \cdots \times n_{(M-1)})\end{aligned}$$

The basic parameters of the resulting code are $\left[N,\, K,\, d_{mMD}\right]$, where:

$$N = n_1 \times n_2 \cdots \times n_M\,, \tag{14}$$

$$K = k_1 \times k_2 \cdots \times k_M\,, \tag{15}$$

$$d_{mMD} = d_1 \times d_2 \cdots \times d_M\,. \tag{16}$$

One can observe that the matrix (13) contains the following submatrix in the first rows from the top

$$\mathbf{H}_{cT} = \begin{bmatrix} \mathbf{H}_i & \mathbf{0}_{i,j} & \dots & \mathbf{0}_{i,j} \\ \mathbf{0}_{i,j} & \mathbf{H}_i & \dots & \mathbf{0}_{i,j} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{i,j} & \mathbf{0}_{i,j} & \dots & \mathbf{H}_i \end{bmatrix}. \tag{17}$$

4 RLL-ECCs construction from product codes

4.1 Overview of previous relevant results

In [6] a method was proposed of how RLL-ECCs could be obtained from ECCs with control matrices which possess the following property.

Property. If the control equation of a linear binary block code C contains a set E with an even number of symbols, then inverting an odd number of symbols from E in all the codewords of C creates a new code C' in which no codeword has symbols which are all equal to zero or all equal to one in E (C' is a coset code of C).

This property could be exploited multiple times for non-overlapping sets of symbols in order to find a modifier $\mathbf{m}$ which could be added to each codeword after it is encoded before transmission or storing:

$$\mathbf{c}' = \mathbf{c} + \mathbf{m}\,. \tag{18}$$

In (18) the $\mathbf{c}'$ denotes the transmitted or stored codeword of the RLL-ECC and the addition is a vector addition over $GF(2)$. (In other words, the corresponding coordinates are added modulo 2).

Before decoding, the influence of the modifier could be eliminated by adding it to the received word or word read out from a storage medium

$$\hat{\mathbf{c}} = \mathbf{c}' + \mathbf{m}\,. \tag{19}$$

In (19) $\hat{\mathbf{c}}$ is the estimated codeword from the original ECC, which can contain errors. Therefore it has to be decoded using the usual procedure for the specific ECC. For example the Pyndiah algorithm could be used for PCs. For more details on decoding the reader is advised to consult [9–11].

Let us present a simple example from [5]. The well-known binary $[7, 4, 3]$ Hamming code has the following $\mathbf{H}$ matrix (6). (In this case it is also the control matrix $\mathbf{H}_{1c}$.) Each codeword $\mathbf{c} = \left(c_6,\, c_5,\, c_4,\, c_3,\, c_2,\, c_1,\, c_0\right)$ must satisfy the following control equations:

$$c_2 + c_3 + c_4 + c_5 = 0\,, \tag{20}$$

$$c_1 + c_3 + c_4 + c_6 = 0\,, \tag{21}$$

$$c_1 + c_2 + c_4 + c_7 = 0\,. \tag{22}$$

In this example the corresponding 3 sets of symbols are not disjunctive and therefore we can choose one of them in order to obtain an appropriate modifier. If we take the set of symbols $\{c_2,\, c_3,\, c_4,\, c_5\}$ which participate in (20), then the modifier can have an odd number of ones in the corresponding set of coordinates, namely in $\{2,\, 3,\, 4,\, 5\}$. For example

$$\mathbf{m} = (0\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0)\,. \tag{23}$$

The sequence composed of concatenated codewords of the resulting RLL-ECC cannot contain more than 12 identical symbols [5].

In [6] an answer was given to the question if this construction could be generalized to a more-dimensional SPC-PC by proofing the following proposition. For convenience of the reader we also attach the proof.

Proposition. *From each binary M-D, an SPC-PC with parameters* $\left[n_1 \times n_2 \times \cdots \times n_i \times \cdots \times n_M,\ k_1 \times k_2 \times \cdots \times k_i \times \cdots \times k_M,\ 2^M\right]$, *in which at least one component code has an even codeword length (say* n_i*), an RLL-ECC could be obtained using the construction exploiting a modifier described in [5]. The resulting RLL-ECC has the same parameters and not more than* $2(n_i - 1)$ *identical symbols appear in any string of its codewords.*

P r o o f . It is obvious that if at least one component code has an even codeword length (say n_i), then a control matrix for the M-D SPC-PC, could be obtained in which exactly $n_1 \times n_2 \times \cdots \times n_{i-1} \times n_{i+1} \times \cdots \times n_M$ rows contain an even number n_i of consecutive ones in each of them. For example using the approach described in Section 3 of this paper and mapping the codewords of the code with even codeword lengths into the first rows (from the top) of $\mathbf{H}_c$ without any interleaving. Please see (6) as an example. Therefore the union of the non-overlapping sets of symbols corresponding to these sets of ones contains all $n_1 \times n_2 \times \cdots \times n_i \times \cdots \times n_M$ codeword symbols. Based on the property proven in [5] in each of these sets an unequal number of symbols could be negated via the ones in the modifier. Using identical arguments as in [5], consequently the longest run of identical symbols cannot be longer than $2(n_i - 1)$ in any sequence of consecutive codewords from such a code.

4.2 Novel result

Further generalization of the construction presented in [11] is possible, namely the component codes from the original PC from which the RLL ECC could be constructed do not have to be only SPCs, but they could be also any other binary linear block ECCs. The necessary condition is that one of them has to be an RLL-ECC. More formally it could be expressed as follows.

PROPOSITION. *From any binary M-D PC with parameters* $[n_1 \times n_2 \times \cdots \times n_M,\ k_1 \times k_2 \times \ldots k_M,\ d_1 \times d_2 \times \cdots \times d_M]$ *an RLL-ECC could be obtained using the construction exploiting a modifier based on the property considered in* [5] *if at least one binary 1-D component ECC could be converted to an RLL-ECC. The RLL properties of the constructed RLL-ECC will be identical to the RLL properties of the chosen 1-D component RLL-ECC.*

P r o o f . Let the 1-D component code of a binary M-D PC, which could be brought to an RLL-ECC have codeword length n_i and control matrix $\mathbf{H}_i$. It is obvious that the control matrix for the binary M-D PC, could be obtained in a form given in (13). In it there are N/n_i submatrices identical to $\mathbf{H}_i$ as in the main diagonal of matrix $\mathbf{H}_{cT}$ given in (17). In each column the nonzero components could appear only in the positions containing one of the columns from only one of the submatrices $\mathbf{H}_i$. Therefore the coordinates of the resulting M-D PC codeword symbols have not only a bijective correspondence with the columns of these submatrices $\mathbf{H}_i$, but they could be divided into N/n_i consecutive disjunctive sets each containing exactly n_i coordinates. The union of all sets contains all coordinates of the resulting M-D PC. The sets could be concatenated so that the coordinates are in order. Therefore, the resulting modifier could be formed as a concatenation of the modifiers used to obtain the original 1-D RLL ECC from the original component 1-D ECC. Moreover the RLL properties of the constructed RLL ECC will be identical to the RLL properties of the original 1-D ECC.

For example if the chosen 1-D RLL ECC has codeword length n_i, then the longest run of identical symbols cannot be longer than $2(n_i - 1)$ in any sequence of consecutive codewords from the resulting RLL-ECC.

5 Conclusions

In this paper it was shown that from a binary M-D PCs it is possible to obtain RLL-ECCs using the method described in [5] if at least one of its 1-D binary component codes could be transformed to an RLL-ECC using the method in [5] and that the RLL properties of the resulting RLL-ECC will be the same as the RLL properties of the 1-D RLL ECC. The main advantages of the construction in [5] will be preserved-namely that no additional redundancy is introduced by the construction and also that the decoding procedure of the original M-D PC does not have to be modified.

Acknowledgements

This work was supported by the European Space Agency (Contract No: 4000117400/16/NL/NDe), Scientific Grant Agency of Ministry of Education of Slovak Republic and Slovak Academy of Sciences (grant VEGA 1/0477/18) and the Grant Agency Academic Alliance (grant GA/2016/10).

Comment: Certain part of this paper was presented at the International Conference Cybernetics & Informatics 2018, Lazy pod Makytou, Slovak Republic, January 31-February 3, 2018.

REFERENCES

[1] R. E. Blahut, "Digital Transmission of Information", *Addison Wesley*, 1990.

[2] T. S. Perry, "The PIT Boss", *IEEE Spectrum*, vol. 54, no. 5, May 2017, pp. 32–50, doi: 10.1109/MSPEC.2017.7906897.

[3] A. Kees and Schouhamer Immink, "Codes for Mass Data Storage Systems", *Shannon Foundation Publisher*, 2004.

[4] P. Farkaš and F. Schindler, "Construction for Obtaining Trellis Run Length Limited Error Control Codes from Convolutional Codes", *Journal of Electrical Engineering*, vol. 68 no. 5, Aug 2017, pp. 401–404.

[5] P. Farkaš and F. Schindler, "Run Length Limited Error Control Codes Construction based on One Control Matrix Property", *Journal of Electrical Engineering*, vol. 68 no. 4, June 2017, pp. 322–324.

[6] P. Farkaš, T. Janvars, K. Farkašová and E. Ružický, "On Run-Length Limited Error Control Codes Constructed from Binary Single Parity Check Product Codes", *International conference Cybernetics & Informatics 2018*, Lazy pod Makytou, Slovakia, 31.1.–3.2.2018.

[7] J. I. Hall, "Notes on Coding Theory – Chapter 3, Linear Codes", Michigan State University, Michigan, 2010, http:// users.math.msu.edu/users/jhall/classes/codenotes/Linear.pdf, accessed March 2017.

[8] L. Calabi, and H. Haefeli, "Class of Binary Systematic Codes Correcting Errors Occurring at Random and Bursts", *IRE Transactions on Circuit Theory*, vol. 6, no. 5, May 1959, pp. 79–94,.

[9] D. M. Rankin and T. A. Gulliver, "Single Parity Check Product Codes", *IEEE Transactions on Communications*, vol. 49, no. 8, Aug 2001, pp. 1354–1362, doi: 10.1109/26.939851.

[10] T. Janvars and P. Farkas, "On Decoding of Multidimensional Single Parity Turbo Product Codes", *Mobile Future and Symposium on Trends Communications*, 2003. SympoTIC '03 Joint First Workshop on, pp. 25–28. doi: 10.1109/TIC.2003.1249080.

[11] R. M. Pyndiah, "Near-Optimum Decoding of Product Codes: Block Turbo Codes", *IEEE Trans. Commun*, vol. 46, Aug 1998, pp. 1003–1010.

Received 6 March 2018