HARDWARE IMPLEMENTATION OF RANK CODEC

Igor Sysoev, Ernst Gabidulin

Abstract: *The authors present a hardware implementation of the codec for rank codes. Parameters of rank code are (8,4,5). Algorithm was implemented on FPGA Spartan 3. Code rate is 1/2. The codec operates with elements from Galois field $GF(2^8)$. The device can process informational data stream up to 77 MiB/s. Proposed results should help understanding rank code structure and simplify the problem of its application.*

Keywords: *rank codes, codec, error correction code, weak self-orthogonal bases, rank metric, FPGA, key equation, Euclidean algorithm, fast computing.*

ACM Classification Keywords*: B.2.4 High-Speed Arithmetic*

MSC*: 12Y05, 65Y20, 68W35.*

Introduction

Rank codes were developed in 1985 [Gabidulin, 1985]. They are an analogue of Reed-Solomon codes. The key difference between them lies in the definition of the metric. Unlike Reed-Solomon codes, which use Hamming metric, rank codes use rank metric. A new technique of data transmission, called "Network coding" was proposed in 2000 [Ahlswede, 2000]. According to this technique several packets can be combined together by a node for transmission and the rank codes are the most suitable method of correcting errors in such systems. There are many theoretical papers dedicated to rank coding. However none of them consider hardware implementation.

In this work we will concentrate on our algorithms for rank codes and describe how we implemented them. Next part of the paper is organized as follows. At first it will be discussed coding procedure and its complexity. Secondly we are going to discuss our implementation of decoder block, talk about calculating the syndrome, solving key equation, computing error occured and correcting an informational vector. Thirdly we will give the information about resource requirements of the coder. And finally we are going to emphasize and discuss the main results.

Coding scheme

Coding scheme is implemented through multiplication informational vector u by generating matrix G. Coding scheme is similar to rank code syndrome calculating procedure. Entries of the matrix are the elements of the extended field $GF(q^N)$. Let rank code has such parameters – (n, k, d). Then we can write an informational vector

$$u = (u_1, u_1, \ldots .u_k), u_i \in GF(q^N). \tag{1}$$

We assume $n = N$ and $n = 2k$ unless otherwise stated. So current code rate equals $1/2$. Generating matrix G of rank code are

$$G_{(k\times n)} = \begin{Vmatrix} g_1 & g_2 & \cdots & g_n \\ g_1^{[1]} & g_2^{[1]} & \cdots & g_n^{[1]} \\ \vdots & \vdots & \ddots & \vdots \\ g_1^{[k-1]} & g_2^{[k-1]} & \cdots & g_n^{[k-1]} \end{Vmatrix}, \tag{2}$$

where $[i]$ means Frobenius power or q^i. Calculated code vector is equal to the product

$$g = u \cdot G = (g_1, g_2, \ldots, g_n), g_i \in GF(q^N). \tag{3}$$

Papers [Gabidulin, 2010] and [Sysoev, 2011] are describe how to perform similar operation using weak self-orthogonal bases [Gabidulin, 2006]. In this case asymptotic complexity of the operation can be evaluated as ($N = n$)

$$\mathcal{C}_{code} = \mathcal{O}\left((\log N)^2 N\right) + \mathcal{O}(N^{\log_2 3}). \tag{4}$$

Decoding scheme

Code vector g can be distorted during data transmission. So write

$$y = g + e, \tag{5}$$

where

$$e = (e_1, e_2, \ldots, e_n), e_i \in GF(q^N). \tag{6}$$

If there are no more information apart from received vector y then decoder will be able to correct this word if and only if

$$rank(e) \leq \frac{d-1}{2}. \tag{7}$$

Syndrome computing

There are some ways to decode rank codes. Decoding using syndrome (i.e. syndrome decoding) are best investigated. Syndrome is the product of received vector y and check matrix H

$$s = y \cdot H. \tag{8}$$

If all entries of the syndrome (8) are equal to zero elements then rank decoder makes a decision that there are no errors. It associates the received vector $c = y$ with an appropriated informational vector. The check matrix,

$$H_{(d-1\times n)} = \begin{Vmatrix} h_1 & h_2 & \cdots & h_n \\ h_1^{[1]} & h_2^{[1]} & \cdots & h_n^{[1]} \\ \vdots & \vdots & \ddots & \vdots \\ h_1^{[k-1]} & h_2^{[k-1]} & \cdots & h_n^{[k-1]} \end{Vmatrix}, \tag{9}$$

is connected to the generating matrix (2) and must fulfil next condition

$$GH^T = 0. \tag{10}$$

The complexity of the operation (8) may be estimated in similar way as the complexity of coding scheme. So we can use the evaluation (4).

Key equation solving

An important decoding step is the key equation solving. We shall denote the basis of an error space by b_E, that is

$$b_E = (E_1, E_2, \ldots, E_m), \tag{11}$$

where m – rank of the error occurred with $E_i \in GF(q^N)$. For next discussion we will need the definition

Definition 1: Polynomial $L(z)$ over $GF(q^N)$ called *linearized* if $L(z) = \sum_i L_i z^{p^i}$.

Let $\Delta(z)$ denotes a linearized polynomial which roots are superposition of the vectors from (11), so called *error span polynomial*,

$$\Delta(z) = \sum_{p=0}^{m} \Delta_p z^{[p]}. \tag{12}$$

Then

$$\Delta(E) = 0, \tag{13}$$

and E is any linear superposition of the basis vectors from (11)

$$E = \sum_{i=1}^{m} \beta_i E_i, E_i \in b_E, \forall \beta_i \in GF(q). \tag{14}$$

Then let us write the key equation for rank code

$$F(z) = \Delta(z) \cdot S(z) \mod z^{[d-1]}. \tag{15}$$

In (15) $F(z)$ are defined as

$$F(z) = \sum_{i=0}^{m-1} F_i z^{[i]}, \tag{16}$$

where

$$F_i = \sum_{p=0}^{i} \Delta_p s_{i-p}^{[p]}, i = 0, 1, \ldots, m-1 \tag{17}$$

and $S(z)$ denotes linearized syndrome polynomial

$$S(z) = \sum_{j=0}^{d-2} s_j z^{[j]}. \tag{18}$$

In paper [Gabidulin, 1985] the author showed that for solving the key equation it is sufficiently to divide $z^{[d-1]}$ by syndrome polynomial (18). One of the way is using the method of successive divisions (i.e. Euclidean algorithm). In papers [Sysoev, 2011] and [Gabidulin, 2011] it was proposed optimized Euclidean algorithm for linearized polynomials. This algorithm allow us to use recurrence scheme and are most suitable for hardware implementation. Its asymptotic complexity are evaluated as

$$\mathcal{C}_{euclid} = \mathcal{O}(N^{3.585}). \tag{19}$$

Finding roots of error span polynomial

After the last operation the decoder should find roots of key equation $\Delta(z) = 0$

$$\Delta(z) = \sum_{p=0}^{m} \Delta_p z^{[p]}. \tag{20}$$

Remember that this solution should satisfy (14). To solve key equation we can use the method, described in [Berlekamp, 1968]. For this, we will need the following theorem

Theorem 1: *Let $L(z)$ – linearized polynomial and $z = \sum_k Z_k \alpha^k$, where $Z_k \in GF(q)$, then $L(z) = \sum_k Z_k L(\alpha^k)$*

Proof. See [Berlekamp, 1968]. ■

From Theorem 1 it follows that for finding roots of error span polinomial 12, it is necessary to calculate value of this polynomial in certain points and express the results in standard basis

$$b_{std} = (\alpha^0, \alpha^1, \alpha^2, \ldots, \alpha^N), \alpha \in GF(q^N). \tag{21}$$

That is

$$\Delta(\alpha^k) = \sum_{i=0}^{m} \lambda_{ki} \cdot \alpha^i, k = 0, 1, \ldots, m, \lambda_{ki} \in GF(2^N). \tag{22}$$

After this calculation we can construct a matrix $(m+1) \times (m+1)$, which entries are from $GF(q)$

$$\Omega = \begin{Vmatrix} \lambda_{00} & \lambda_{01} & \cdots & \lambda_{0m} \\ \lambda_{10} & \lambda_{11} & \cdots & \lambda_{1m} \\ \cdots & \cdots & \cdots & \cdots \\ \lambda_{m0} & \lambda_{m1} & \cdots & \lambda_{mm} \end{Vmatrix}. \tag{23}$$

Then the problem of finding roots of error span polynomial is equivalent to the finding a solution for the next matrix equation:

$$Z \cdot \Omega = \begin{vmatrix} Z_0 & Z_1 & \ldots & Z_m \end{vmatrix} \cdot \begin{Vmatrix} \lambda_{00} & \lambda_{01} & \cdots & \lambda_{0m} \\ \lambda_{10} & \lambda_{11} & \cdots & \lambda_{1m} \\ \cdots & \cdots & \cdots & \cdots \\ \lambda_{m0} & \lambda_{m1} & \cdots & \lambda_{mm} \end{Vmatrix} = \begin{vmatrix} 0 & 0 & \cdots & 0 \end{vmatrix}. \tag{24}$$

The null vector is placed at the right side of the equation (24). This equation may be solved by the means of transformation of the matrix from current form into idempotent one (see example 2.57 in [Berlekamp, 1968]). Then the roots of error span polynomial, or (11), may be derived from matrix rows. Thus we will calculate error values (11).

Find error locator

After calculating (11) it is necessary to calculate error locator. At first step we need to solve a shortened set of equations [Gabidulin, 1985]:

$$\sum_{j=1}^{m} E_j x_j^{[p]} = s_p, p = 0, 1, \ldots, m-1. \tag{25}$$

All the equations transform in such way that at the end of operations we will got unknown variables of equal power, that is

$$\sum_{j=1}^{m} E_j^{[m-p]} x_j^{[m-1]} = s_p^{[m-p]}, p = 0, 1, \ldots, m-1. \tag{26}$$

Complexity of the operation (25) are evaluated as

$$\mathcal{C}_{conv} = \mathcal{C}_{mult} \left((m-1)^2 + \frac{(m-1)m}{2} \right), \tag{27}$$

where $\mathcal{C}_{mult}$ is the multiplication complexity for the pair of elements from the extended finite field $GF(q^N)$ in the current basis. In paper [Gabidulin, 2010] and [Sysoev, 2011] It was shown how we may optimize a set of base operations (multiplication, powering, inversion) using weak self-orthogonal bases. The set of equations (26) is the system of linear equations. It can be rewrited into the product of matrix by vector

$$\begin{Vmatrix} E_1^{[m-1]} & E_2^{[m-1]} & \cdots & E_m^{[m-1]} \\ E_1^{[m-2]} & E_2^{[m-2]} & \cdots & E_m^{[m-2]} \\ \vdots & \vdots & \ddots & \vdots \\ E_1^{[0]} & E_2^{[0]} & \cdots & E_m^{[0]} \end{Vmatrix} \begin{Vmatrix} x_1^{[m-1]} \\ x_2^{[m-1]} \\ \vdots \\ x_{m-1}^{[m-1]} \end{Vmatrix} = \begin{Vmatrix} s_0^{[m-1]} \\ s_1^{[m-2]} \\ \vdots \\ s_{m-1}^{[0]} \end{Vmatrix}. \tag{28}$$

To solve a system of equations (26) expressed in (28), we will use Gaussian elimination. So

$$\begin{Vmatrix} 1 & \tilde{E}_{12} & \cdots & \tilde{E}_{1m} \\ 0 & 1 & \cdots & \tilde{E}_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{Vmatrix} \begin{Vmatrix} x_1^{[m-1]} \\ x_2^{[m-1]} \\ \vdots \\ x_{m-1}^{[m-1]} \end{Vmatrix} = \begin{Vmatrix} \tilde{s}_0 \\ \tilde{s}_1 \\ \vdots \\ \tilde{s}_{m-1} \end{Vmatrix}. \tag{29}$$

Complexity of solving such the system (29) using Gaussian Elimination evaluated as

$$\mathcal{C}_{Gauss} = \mathcal{C}_{inv} \cdot m + \mathcal{C}_{mult} \left(\sum_{k=1}^{m} k^2 \right) + \mathcal{C}_{add} \left(\sum_{l=1}^{m} l(l-1) \right), \tag{30}$$

where $\mathcal{C}_{add}$ and $\mathcal{C}_{inv}$ are appropriated estimations of complexity for the sum of two elements from extended field and inversion of the element in current basis.
Complexity of the solution (reduction to Gaussian form) equals to

$$\mathcal{C}_{solveX} = (\mathcal{C}_{mult} + \mathcal{C}_{add}) \frac{m(m-1)}{2}. \tag{31}$$

After this operation we will got a set of independent equations

$$\sum_{i=1}^{m} x_p^{[m-1]} = \hat{s}_p, p = 0, 1, \ldots, m-1. \tag{32}$$

Each equation from (32) is solving by means of raising to power $[n-(m-1)]$. Complexity of the powering may be evaluated as

$$\mathcal{C}_{xPow} = \mathcal{C}_{mult} \cdot m(n-(m-1)). \tag{33}$$

After the calculating x_p we may form a system of equation which solution is a set of error locators [Gabidulin, 1985]

$$x_p = \sum_{j=1}^{n} Y_{pj} h_j, p = 1, 2, \ldots, m. \tag{34}$$

In equation (34) h_j are the entries of the first row in check matrix (9) with $h_j \in GF(q^N)$. Y_{pj} are elements of the error locators matrix $(m \times n)$, $Y_{pj} \in GF(q)$. It is worth noting that all the equations (34) are independent and have unique solution. The element $h_j \in GF(q^N)$ can be expressed in the form of vector which entries are the elements h_{ji} from $GF(q)$. Thus each equation (34) can be represented in the next form

$$X_p = \hat{H} \cdot Y_p = \begin{Vmatrix} x_{p1} \\ x_{p2} \\ \vdots \\ x_{pn} \end{Vmatrix} = \begin{Vmatrix} h_{11} & h_{21} & \cdots & h_{n1} \\ h_{12} & h_{22} & \cdots & h_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ h_{1n} & \cdots & \cdots & h_{nn} \end{Vmatrix} \cdot \begin{Vmatrix} Y_{p1} \\ Y_{p2} \\ \vdots \\ Y_{pn} \end{Vmatrix}. \tag{35}$$

To find Y_p it is necessary to multiply X_p by the matrix $\hat{H}^{-1}$. This matrix is independent from input information and may be evaluated at the step of device development. Complexity of the operation should be not more than (upper-bound estimate)

$$\mathcal{C}_Y = \mathcal{C}_{add} \cdot (n-1)m. \tag{36}$$

Error finding and vector correction

Using known error locators matrix Y_{pj} (from (34)), and the obtained error span matrix E_i (11), it will be simple to got error values e_k for each entry from received corrupted code vector y

$$e = EY = \begin{Vmatrix} E_1 & E_2 & \cdots & E_m \end{Vmatrix} \cdot \begin{Vmatrix} Y_{11} & Y_{12} & \cdots & Y_{1n} \\ Y_{21} & Y_{22} & \cdots & Y_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{m1} & Y_{m2} & \cdots & Y_{mn} \end{Vmatrix}. \tag{37}$$

Error correction operation is a simple subtraction

$$g = y - e. \tag{38}$$

A common complexity of the error calculation and code vector correction are evaluated as

$$\mathcal{C}_e = \mathcal{C}_{add} \cdot (nm). \tag{39}$$

After the correction of received vector we need to find informational vector

$$\tilde{u} = g \cdot D = u \cdot G \cdot D. \tag{40}$$

For equality $\tilde{u}$ and u it is necessary to satisfy the condition

$$G_{(k\times n)} \cdot D_{(n\times k)} = I_{(k\times k)}, \tag{41}$$

where I denotes the unity matrix with identity components from $GF(q^N)$ at the diagonal. Matrix D is defined unambiguously and calculates a priori. Complexity of multiplication vector g by matrix D may be estimated as

$$\mathcal{C}_u = \mathcal{C}_{mult} \cdot mn + \mathcal{C}_{add} \cdot (n-1)m. \tag{42}$$

Resource demanding

The authors have implemented described algorithm using hardware description language VHDL. Our goal was the developing a codec for rank code with the following parameters $(8, 4, 5)$. A base field was $GF(2)$. An extended field was $GF(2^8)$, i.e. $N = 8$. Decoder has no information about row and column erasures.

Codec was designed as IP-block with simple inputs and outputs. The rest of the world can use signal “LOAD” and “Y[7:0]” for data input. For outputs other blocks should use corrected vector “I[7:0]" with “VALID” and “FAIL" flags. Such way does not constrain data channel. So the codec can be used with any transmitter such as copper (i.e. RS-485/Ethernet), memory (i.e. NAND Flash) or radio (i.e. ZigBee/WiFi) channels.

For more objective results this rank codec does not depend on specialized FPGA blocks (such as multipliers, block ram, I/O blocks). The device may be used not only in FPGA but also in applications specific integrated circuit (ASIC). Block scheme of full decoder you can see at figures 1 and 2.

Figure 1 shows blocks for searching error basis and interconnection between them. One can see two state machines, for “Euclid” and “Error Span” stages (denoted as “SM”). Structure is purposefully pipelined. So next decoding operation starts can begin before previous operation finish. Syndrome block consist of four independent calculators for each transposed check matrix row. Euclid stage block performs operations on linearized polynomials. “Euclid main” block are more complex and more frequently used. It decreases current power at each step of Euclidian algorithm. “Euclid final” are used once per decoding. “Error Span” block is simpler. It performs calculate values of Δ linearized polynomial and outputs its roots.

Figure 2 shows block diagram of algorithm for calculating error locator. One can see state machines chain. These state machines share memory and computational resources between each other. The system also have pipelined structure. Final stage “Data correction" check syndrome value and can decode received vector if there are no errors occured. Otherwise, it performs errors searching and, after that, calculations for finding informational vector. The obtained structure and results may be optimized better. But they are initial point and will help us to more precise evaluate complex systems based upon rank codes.

We used Active-HDL 7.2 Student Edition from Aldec [Aldec inc., 2012] for simulation. Our project was developed for commercial FPGAs XC3S700AN-4FGG484C [Xilinx inc., 2008][Xilinx inc., 2012] and XC6SLX16-3CSG225 [Xilinx inc., 2012]. For generating the configuration bit file for FPGA we use software ISE WebPack by Xilinx inc. 13.1 [Xilinx inc., 2011].

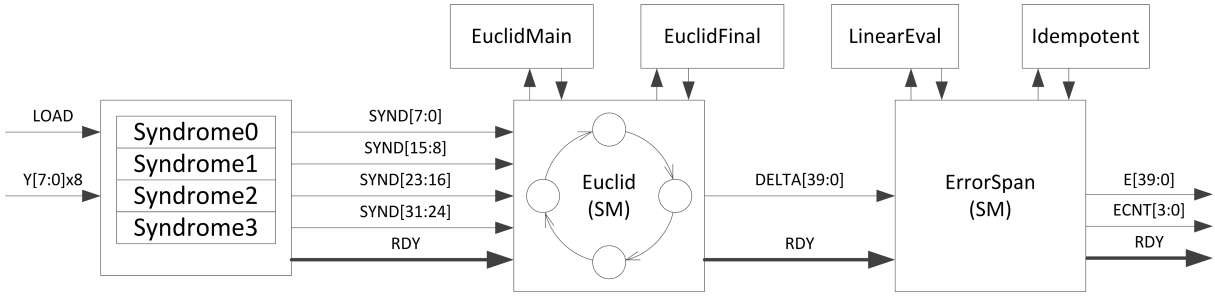

Figure 1: Searching error basis block diagram

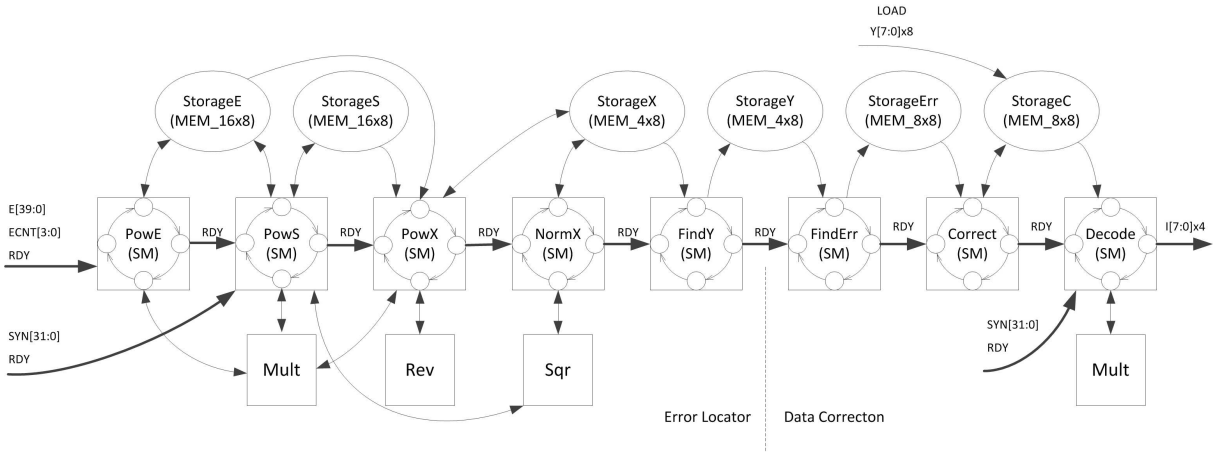

Figure 2: Block diagram searching error locator and decoding received vector

Synthesis, translation, mapping and routing were performed with default settings. One can see authors's results in table 1. The codec can be run at 155 MHz Spartan-6 and 86 MHz for Spartan-3. So if current code rate is 1/2 then codec can perform data processing with 77 MiB/s and 43 MiB/s speed accordingly. When this work started, the best chip for commertial devices was Spartan-3AN. Today FPGA Spartan-6 preferred for new budget designs. It is recent and more popular FPGA [Xilinx inc., 2012]. The key difference between two FPGAs is the using technology and architecture. The Spartan-3 (denotes as "S3") technology is 90 nm, while Spartan-6 (denoted as "S6") one is 45 nm. Such difference in technology impacts on achievable speed. Indeed, table 1 shows speed increasing.

Spartan 3A contain 4-input Look-Up table (LUT). Each LUT implements logic plus storage elements used as flip-flops or latches [Xilinx inc., 2012]. Each configurable logic block in Spartan-6 consists of two slices. Each slice in Spartan-6 contains four 6-input LUTs, eight flip-flops and miscellaneous logic. As one can see in table of results, in new architecture number of used slices has been dramatically decreased. Obtained results show this difference between two FPGAs.

So table describes not only resource demanding for the new device, but also for legacy one. One can see and how new architecture suits for the codec. The main conclusion from the results lies in idea that rank codes may be implemented for industrial and experimental purposes, even in small series-produced devices.

Table 1: Simulation and synthesize results

Operation	Delay,	Slices,		Flip-Flops,		4-input LUTs,		Max. freq,	
	cycles	number (%)		number (%)		number (%)		MHz	
		S3	S6	S3	S6	S3	S6	S3	S6
Full coder	14	850 (14)	209 (9)	921 (8)	630 (3)	1247 (11)	859 (9)	163	282
Syndrome stage	25	728 (12)	196 (9)	897 (8)	620 (3)	933 (8)	614 (7)	176	313
KES (Euclid) stage	2040	1837 (31)	826 (36)	1980 (17)	1956 (11)	2945 (25)	2045 (22)	87	156
Error span stage	977	489 (8)	156 (7)	321 (3)	302 (2)	859 (7)	422 (5)	130	174
Error locator stage	401	877 (15)	262 (12)	927 (8)	784 (4)	1214 (10)	545 (6)	143	202
Data correction stage	15	96 (2)	32 (1)	92 (1)	82 (0)	164 (1)	72 (1)	212	344
Full decoder	3458	3979 (68)	1380 (61)	4420 (38)	4027 (22)	6106 (54)	3564 (40)	86	155
Full codec	3472	4829 (82)	1583 (70)	5341 (45)	4657 (26)	7353 (62)	4423 (49)	86	155

Conclusion

Achieved results are important for further scientific research. They will help to estimate more carefully parameters of the systems based upon rank codes. Understanding the internal structure of the hardware codec should direct research efforts to the most important problems. Likewise, the results may be used to compare rank codes with Reed-Solomon codes from applied aspect.

In what follows one should consider more complex codecs, examine dependence of hardware implementation complexity on the main parameters and develope new algorithms for more efficient decoding.

Acknowledgements

The authors would like to thank Nina. I. Pilipchuk, Sergey M. Vladimirov, Alexander I. Kolybelnikov, Alexey V. Ourivsky and Alexander L. Shishkin for constructive critique and valuable advices.

Bibliography

[Gabidulin, 1985] E.M. Gabidulin, Theory of Codes with Maximum Rank Distance, Problems of Information Transmisson, Moscow, Vol. 21, No 1, 1985, pp. 3–16.

[Gabidulin, 2010] E.M. Gabidulin and I.Y. Sysoev Rank Codes Using Weak Self-Orthogonal Bases, Computational Technologies in Electrical and Electronics Engineering (SIBIRCON-2010), Irkutsk, July 2010, pp. 70–71.

[Ahlswede, 2000] R. Ahlswede E.,N. Cai S.-Y. R. Li and R.W.Yeung, Network information flow, IEEE Trans. Inform. Theory, Vol. IT-46, pp. 1204-1216, July 2000.

[Sysoev, 2011] I.Y. Sysoev, Fast Algorithm for Computing Rank Code Syndrome Using Weak Self-Orthogonal Basis in $GF(2^{2^N})$, Proceedings of the 13-th International Conference "Digital signal processing and its applications", Moscow, 2011, Vol.1, pp. 78–81.

[Gabidulin, 2006] E.M. Gabidulin and N.I. Pilipchuk, Symmetric matrices and codes correcting rank errors beyond the $\lfloor (d-1)/2 \rfloor$ bound, Discrete Appl. Math., Feb. 2006, Vol. 154, I. 2, pp. 305–312.

[Gabidulin, 2011] E.M. Gabidulin and I.Y. Sysoev, Modified Algorithm for Decoding Rank Codes (in russian), *Proc. of 54th MIPT Scientific Conference*, 2011, Vol. 2, pp. 55-57.

[Sysoev, 2011] I.Y. Sysoev, Efficient Euclidian Algorithm for Linearized Polynomials (in russian), Proc. of the IX International Scientific Conference "Perspective technologies in the Information Transfer Means 2011", Vladimir, 2011, Vol. 3, pp. 40–45.

[Berlekamp, 1968] E.R. Berlekamp, Algebraic Coding Theory, New-York: McGraw-Hill Book Company, 1968.

[Aldec inc., 2012] Aldec inc., Active-HDL Manual, www.aldec.com, Aldec Web-cite, 2012.

[Xilinx inc., 2008] Xilinx inc, Spartan-3AN FPGA Family Datasheet, www.xilinx.com, 2008, P. 108.

[Xilinx inc., 2012] Xilinx inc., Spartan-3AN FPGA User Guides, www.xilinx.com, 2012.

[Xilinx inc., 2012] Xilinx inc., Spartan-6 Family Overview, www.xilinx.com, 2012.

[Xilinx inc., 2011] Xilinx inc., ISE Design Suite, http://www.xilinx.com/support/download/index.htm, 2011.

Authors' Information

Igor Y. Sysoev *– Post-graduate student, Moscow Institute of Physics and Technology (State University), P.O. Box: 141700, Dept. of Radio Engineering and Telecommunications, Dolgoprudny, Russia; e-mail: Igor.Sisoev@gmail.com*
Major Fields of Scientific Research: Rank Codes, ECC, Complexity, Digital Systems.

Ernst M. Gabidulin *– Doctor of Engineering, Professor, Moscow Institute of Physics and Technology (State University), P.O. Box: 141700, Dept. of Radio Engineering and Telecommunications, Dolgoprudny, Moscow Region, Russia; e-mail: ernst_gabidulin@yahoo.com*
Major Fields of Scientific Research: Coding Techniques, Coding Theory, Cryptography, Complexity, Shannon Theory.