Computing Nonlinear Polynomial Functions on FPGA-Class PLD Arrays

Sergei Shalagin
Dept. of Computer Systems
Kazan National Research Technical
University – KAI (KNRTU-KAI)
Kazan, Russian Federation
0000-0002-2591-2749

***Abstract*— This paper estimates the quantities of the resources of FPGA-class PLD generators of Boolean functions, *D*-triggers, and input/output units that are required for the distributed computing of nonlinear polynomial functions over a Galois field of a certain power in a preset number of variables. Computations were performed considering the usage factor of the relevant PLD/FPGA resources, as well as the number of the variables for the above functions. We also calculated the PLD/FPGA array size required for computing the predefined polynomial function over a Galois field on both existing and potential distributed computing systems.**

Keywords—polynomial function, FPGA-based PLD architecture, distributed computing systems.

I. INTRODUCTION

In implementing a broad class of systems intended for the real-time processing of digital information, distributed computations can be used, i.e., performing the same parallel and/or pipelined operations on a flow of data implementable in multiprocessing systems (MPSs) [1]. MPS computational nodes are both general-purpose and special-purpose microprocessors (MPs) implemented on VLSIs, particularly on the FPGA-class programmable logic devices [2]. Programmable-architecture MPSs that include elements, such as FPGA-class PLDs [3, 4], can be used as aircraft systems and/or embedded systems that implement various high-reliability devices at different times. In this case, hardware IP cores inside the PLD/FPGA can act as MPs, such as generators of Boolean functions of a given number of variables within configurable logic blocks [2, 5, 6]. Many various-purpose computing devices can be implemented based on the above IP cores [7-13].

This paper defines the estimates of how many resources of a certain FPGA-class PLD are required for a distributed computation of a nonlinear polynomial function over a Galois field of a certain power and of a preset number of variables.

II. PROBLEM STATEMENT

Problem of implementing a broad class of digital signal generating/processing devices on a programmable-architecture MPS reduces to the problem of synthesizing similar computing devices implementing a nonlinear polynomial function of m variables over a Galois field represented as $GF(2^k)$ (NPF(m)) [14]. [15] shows that NPF(m) defined over $GF(2^2)$ can be synthesized on a Virtex-4 PLD based on similar IP cores having a high degree of conformity with this family's PLD architecture. The high conformity degree is achieved due to the configurability of the generators of general Boolean functions of four variables, GBF(4), within the PLD/Virtex-4. Two GBFs(4) allow implementing either the operation of multiplying two elements or the modulo 2 bitwise operation of summing four elements $GF(2^2)$. In general case, similar IP cores, software or schematic-based ones, that implement operations over $GF(2^k)$ have a high degree of conformity with the architecture of an MPS that includes similar elements, i.e., hardware IP cores, due to implementing a general Boolean function of $2{\cdot}k$ variables, GBF($2{\cdot}k$). True is

Statement 1. k GBFs($2{\cdot}k$) allow implementing the operation of multiplying two elements or a bitwise operation of modulo 2 summing of k elements of field $GF(2^k)$.

In [15], estimates of temporary and hardware complexity were obtained for the distributed computation within the PLD/FPGA architecture of an NPF(m) defined over $GF(2^k)$ and represented as:

$$f(x_1,...,x_m)=\sum_{i_1=0}^{w}...\sum_{i_m=0}^{w}a_{i_1...i_m}x_1^{i_1}...x_m^{i_m}, \qquad (1)$$

where $a_{i_1...i_m}, x_1^{i_1}, ..., x_m^{i_m} \in GF(2^k)$ is an elementary polynomial (EP), $i_j=\overline{0,2^k-1}$, $j=\overline{1,m}$, $w=2^k-1$, symbol $\sum$ denotes the bitwise operation of modulo 2 summing. Expressions $a_{i_1...i_m}x_1^{i_1}...x_m^{i_m}$ are defined in (1) as elementary polynomials (EPs). Presence of factors $a_{i_1...i_m}=0$ allows non-computing the values of the relevant EPs. PLDs/FPGAs include MPs that implement GBF($2{\cdot}k$) and the parallel registers for k bits – RG(k), k = 2, 3, … .

III. ESTIMATES OF NPF COMPUTING COMPLEXITY

Suppose the NPF(m) factors in (1), $a_{i_1...i_m}$, $i_j=\overline{0,2^k-1}$, and $j=\overline{1,m}$, are constants. What is required is to find out, at which value of m the function represented as (1) can be implemented on a PLD/FPGA of a given type, provided that it includes $Q_{\mathrm{GBF}(2k)}$ GBFs($2{\cdot}k$), Q_{D} D-triggers, and Q_{IOB} input-output blocks (IOBs), respectively. Solving this problem reduces to computing for an NPF(m) represented as (1), at a given m, the values of $N_{\mathrm{GBF}(2k)}$, N_{D}, and N_{IOB}, for which constraints, such as $N_{\mathrm{GBF}(2k)}$, are true.

978-1-6654-3941-1 @2021 IEEE

$$\begin{cases} N_{\text{GBF}} \le k_{\text{GBF}} \cdot Q_{\text{GBF}} \\ N_{\text{D}} \le k_{\text{D}} \cdot Q_{\text{D}} \\ N_{\text{IOB}} \le k_{\text{IOB}} \cdot Q_{\text{IOB}} \end{cases}, \quad (2)$$

According to [15], the values of k_{GBF}, k_{D}, and k_{IOB} are usually taken as equal to 0.5-0.7. Let us find the values of N_{GBF}, N_{D}, and N_{IOB}.

To calculate NPF(m) represented as (1), the following operations should be executed over the elements of field $GF(2^k)$: 1) Raising the values of x_j, $j = \overline{1, m}$, to powers $p = \overline{2, w-1}$; 2) multiplying two variables (or two variables and a constant); and 3) summing k variables (or k variables with a constant). To execute each of operations (1)-(3), k GBFs($2 \cdot k$) are required.

According to [15], to calculate NPF(m) represented as (1), required are $m \cdot (w-2)$ operations represented as (1), $\sum_{d=1}^{m} \left(2^{m-d} \cdot (w-1)^d \cdot \left(C_m^d - Z_{m,d} \right) \right)$ operations represented as (2), where $Z_{m,d}$ is the number of EPs of d variables, for which $a_{i_1 \ldots i_m} = 0$, and $\sum_{d=0}^{m} \left(C_m^d - Z_{m,d} \right)$ operations represented as (3). As a result,

$$\begin{aligned} N_{\text{GBF}(2k)} &= k \cdot \Big(m \cdot (w-2) + \\ &+ \sum_{d=1}^{m} \left(2^{m-d} \cdot (w-1)^d \cdot \left(C_m^d - Z_{m,d} \right) \right) + , \\ &+ \sum_{d=1}^{m} \left(C_m^d - Z_{m,d} \right) \Big). \end{aligned} \quad (3)$$

Number of IOBs to be involved to ensure computing the value of NPF(m) represented as (1) is

$$N_{\text{IOB}} = k \cdot (m+1). \quad (4)$$

The lower estimate of the number of D-triggers required for computing NPF(m) represented as (1) in the FPGA-class PLD architecture is defined as:

$$N_{\text{D}} \ge N_{\text{GBF}} + N_{\text{IOB}}. \quad (5)$$

In view of the above, true is the following

Statement 1. NPF(m) represented as (1), at the predefined values of factors $a_{i_1 \ldots i_m}$, $i_j = \overline{0, 2^k - 1}$, and $j = \overline{1, m}$, can be implemented on a PLD/FPGA, for which Q_{GBF}, Q_{D}, and Q_{IOB} are defined, provided that constraints (2) are met in computing parameters N_{GBF}, N_{D}, and N_{IOB} according to (3), (5) and (4), respectively.

On a given PLD/FPGA, suppose that condition (2) is met for NPF(m) represented as (1), while it is not met for NPF(m+1) represented as (1). Then computing NPF represented as (1) of q variables, $q > m$, requires involving a programmable-architecture MPS that includes more than one PLD/FPGA, the type of which was predefined initially. Let us re-write expression (1) as:

$$\begin{aligned} &f\left(x_1, \ldots, x_q\right) = \\ &= \sum_{i_{m+1}=0}^{w} \ldots \sum_{i_q=0}^{w} f_{i_{m+1} \ldots i_q}\left(x_1, \ldots, x_m\right) \cdot x_{m+1}^{i_{m+1}} \cdot \ldots \cdot x_q^{i_q}, \end{aligned} \quad (6)$$

$w = 2^k - 1$. If $f_{i_{m+1} \ldots i_q}\left(x_1, \ldots, x_m\right)$ is a variable, then parameters N_{GBF}, N_{D}, and N_{IOB} in inequality (2) are computed, considering this fact according to expressions:

$$\begin{aligned} N_{\text{GBF}} &= k \cdot \Big((q-m) \cdot (w-2) + \\ &+ \sum_{d=0}^{q-m} \left(2^{q-m-d} \cdot (w-1)^d \cdot \left(C_{q-m}^d - Z_{q-m,d} \right) \right) + \\ &+ \sum_{d=0}^{q-m} \left(C_{q-m}^d - Z_{q-m,d} \right), \end{aligned} \quad (7)$$

$$N_{\text{IOB}} = k \cdot (2^{k(q-m)} + m + 1). \quad (8)$$

In view of the above, true is

Statement 2. NPF(q) represented as (6), $q > m$, with the given values of factors $a_{i_1 \ldots i_q}$, $i_j = \overline{0, 2^k - 1}$, and $j = \overline{1, q}$, can be implemented on multiple PLDs/FPGAs with the power of $2^{k(q-m)} + 1$, for which Q_{GBF}, Q_{D}, and Q_{IOB} are defined, provided that constraints (2) are fulfilled in computing parameters N_{GBF}, N_{D}, and N_{IOB} according to (7), (5), and (8), respectively.

Statements 1 and 2 allow defining the maximum number of the variables of NPF represented as (1) and as (6), which can be calculated when using a PLD/FPGA of a given type within the programmable-architecture MPS.

Note 1. Constant values presented in (6) instead of $f_{i_{m+1} \ldots i_q}\left(x_1, \ldots, x_m\right)$ allow considerably reducing the spend of the MPS MPs on computing the value of NPF(q).

The lower estimate of the total number of FPGA-class PLDs required for implementing NPF(q) is defined as:

$$Q_{\text{FPGA}} \ge \left\lceil \max \left(\begin{array}{c} \dfrac{N_{\text{GBF}(2k)}}{k_{\text{GBF}} \cdot Q_{\text{GBF}(2k)}} \\ \dfrac{N_{\text{D}}}{k_{\text{D}} \cdot Q_{\text{D}}} \\ \dfrac{N_{\text{IOB}}}{k_{\text{IOB}} \cdot Q_{\text{IOB}}} \end{array} \right) \right\rceil. \quad (9)$$

At the same time, symbols in the formula are similar to those in (2). There is

Statement 3. The lower estimate of the number of the FPGA-class programmable logic devices required to

implement NPF(q) represented as (6) is defined according to (9).

According to Statement 3 above, we can define the lower estimate of the PLD/FPGA array size, which should be used to implement NPF represented as (6) of a given number of variables over field $GF(2^k)$. Estimate (9) can be improved by implementing on one FPGA array both an IP core implementing NPF(m) and an IP core implementing k multiplexer. This allows reducing the number of IOBs required to transfer among FPGAs the intermediate results obtained in computing the values of NPF represented as (1).

IV. Distributed Computation of NPF

Suppose a given PLD/FPGA, existing or potential, includes MPs implementing an arbitrary GBF(a). Let us consider the implementation of NPF(q) represented as (1) over $GF(2^k)$, provided that $x_{m+1}, ..., x_q$ take the predefined constant values of $\overline{0, w}$, while $x_1, ..., x_m$ vary [9]. We have

$$f\left(x_1, ..., x_q\right) = \psi_{i_{m+1}...i_q}\left(x_1, ..., x_m\right) \qquad (10)$$

with the given $i_j = \overline{0, 2^k - 1}$ and $j = \overline{m+1, q}$. Let us consider the implementation on the above PLD/FPGA of the set $\psi_{i_{m+1}...i_q}\left(x_1, ..., x_m\right)$ as similar IP cores. It is required to compute $\left(2^k\right)^{q-m} - Z$ values of $\psi_{i_{m+1}...i_q}\left(x_1, ..., x_m\right)$, having implemented the relevant number of IP cores and k multiplexers $\left(2^k\right)^{q-m}$-in-1, where Z is the constants used in (10) instead of $\psi_{i_{m+1}...i_q}\left(x_1, ..., x_m\right)$, $i_j = \overline{0, 2^k - 1}$, and $j = \overline{m+1, q}$, according to Note 1.

The language of Statement 1 can be strengthened at implementing on a given PLD/FPGA a set of $\left(2^k\right)^x$ IP cores implementing $\psi_{i_{m+1}...i_q}\left(x_1, ..., x_m\right)$ at predefined $i_j = \overline{0, 2^k - 1}$, $j = \overline{m+1, q}$, as well as k multiplexers $\left(2^k\right)^x$-in-1. In total, the above modules allow implementing NPF($m+x$) over $GF(2^k)$. Let us introduce the following definitions:

Definition 1. An arbitrary $\psi_{i_{m+1}...i_q}\left(x_1, ..., x_m\right)$ defined over $GF(2^k)$ according to (10) can be implemented as pipeline as an IP core when using k GBFs(a), one RG(k) (or k D-triggers): $m \cdot k \le a$, to save the value at the output, as well as m RGs(k) to save the input parameters.

Definition 2. Multiplexer 2^b-in-1 can be pipelined when using a GBF(a): $2^b + b \le a$, a D-trigger to save the values at the output, and $2^b + b$ D-triggers to save input variables.

Definition 3. Multiplexer $\left(2^b\right)^e$-in-1, $e = 1, 2, ...$, can be pipelined when using $2^e - 1$ GBFs(a) and $2^e - 1$ D-triggers to save intermediate and final values, as well as $2^{be} + b \cdot e - Z$ D-triggers to save input variables.

According to Definitions 1-3, implementing $\left(2^k\right)^x - Z$ $\psi_{i_{m+1}...i_q}\left(x_1, ..., x_m\right)$ with predefined $i_j = \overline{0, 2^k - 1}$, $j = \overline{m+1, q}$, as well as k multiplexers $\left(2^k\right)^x$-in-1, requires $(m+x)$ inputs and one output, k bits each; $k\left(2^k\right)^x - Z$ GBFs(a) to implement $\psi_{i_{m+1}...i_q}\left(x_1, ..., x_m\right)$ and $k\left(2^e - 1\right)$ GBFs(a) to implement k multiplexers $\left(2^k\right)^x$-in-1, provided that $e =]k \cdot x/b[$; to save intermediate and final results in computing $\psi_{i_{m+1}...i_q}\left(x_1, ..., x_m\right)$ and multiplexing $\left(2^b\right)^e$-in-1, $k\left(2^k\right)^x - Z$ and $k\left(2^e - 1\right)$ D-triggers, $e =]k \cdot x/b[$ are required, respectively; and to save the values at inputs, only $(m+x)$ RGs(k) (or $k \cdot (m+x)$ D-triggers) are required, the values at the input of multiplexers are saved at the output of the preceding elements. This allows considerably save the number of D-triggers used for distributed computation of NPF($m+x$) over $GF(2^k)$. In view of the above, there is

Statement 4. For distributed computation of NPF($m+x$) over $GF(2^k)$ on a given FPGA implementing GBF(a), $N_{\text{GBF}} = k\left(2^k\right)^x + k\left(2^e - 1\right) - Z$, $e =]k \cdot x/b[$, $N_{\text{D}} = N_{\text{GBF}} + k \cdot (m+x)$, and $N_{\text{IOB}} = k \cdot (m+x+1)$ are required.

Example 1. For a Virtex-7 FPGA, XC7V585T, $Q_{\text{GBF}} = 585,720$, $Q_{\text{D}} = 728,400$, $Q_{\text{IOB}} = 850$, and $k_{\text{GBF}} = k_{\text{D}} = k_{\text{IOB}} = 0.5$, GBF implements an arbitrary Boolean function of six variables: $a = 6$, $Z = 0$. It is required to determine the number of variables for NPF($m+x$) over $GF(2^2)$, implemented on the given FPGA according to Statement 4, condition (9) being met.

In Statements 1-3, $k = 2$, $m = 3$, and $b = 2$ in the given conditions. Therefore, to implement NPF($m+x$) over $GF(2^2)$, $N_{\text{GBF}} = 2\left(4^x + 2^e - 1\right)$, $e = x$, $N_{\text{D}} = N_{\text{GBF}} + 2(3+x)$, and $N_{\text{IOB}} = 2(4+x)$ will be required. Condition (9) will be met for the maximum integer $x = 8$. In accordance with Statement 4, NPF(11) can be implemented over $GF(2^2)$ on the given FPGA.

Example 2. For a Virtex-7 FPGA, XC7V585T, all conditions from Example 1 are met. It is required to determine the number of variables for NPF($m+x$) over $GF(2^3)$, implemented on the given FPGA according to Statement 4, condition (9) being met.

In Statements 1-3, $k = 3$, $m = 2$, and $b = 2$ in the given conditions. Therefore, to implement NPF($m+x$) over $GF(2^3)$, $N_{\text{GBF}} = 3\left(8^x + 2^e - 1\right)$, $e =]3 \cdot x/2[$,

$N_{\mathrm{D}} = N_{\mathrm{GBF}} + 3(2+x)$, and $N_{\mathrm{IOB}} = 3(4+x)$ will be required. Condition (9) will be met for the maximum integer $x = 5$. In accordance with Statement 4, NPF(7) can be implemented over $GF(2^3)$ on the given FPGA.

Let us consider how k multiplexers $\left(2^k\right)^x$ -in-1 are implemented on one PLD. According to definitions 2 and 3, there is

Statement 5. For the distributed implementation of k multiplexers $\left(2^k\right)^y$ -in-1 on a given FPGA implementing GBF(a), $N_{\mathrm{GBF}} = k\left(2^e - 1\right)$, $e = \left]k \cdot y / b\right[$, $N_{\mathrm{D}} = N_{\mathrm{GBF}} + k \cdot 2^{ky} + k \cdot y - Z$, and $N_{\mathrm{IOB}} = N_{\mathrm{D}} - N_{\mathrm{GBF}} + k$ are required.

Example 3. Let us consider the implementation of k multiplexers $\left(2^k\right)^y$ -in-1 on a Virtex-7 FPGA, XC7V585T, for which all the conditions from Example 1 are met. It is required to determine the number of variables x that serve for multiplexing a certain number of the values of $\psi_{i_{m+1}\ldots i_x}\left(x_1,\ldots,x_m\right)$, computed over $GF(2^2)$ according to (10) at the given $i_j = \overline{0,\, 2^k - 1}$, $j = \overline{m+1,\, x}$ and at condition (9) being met.

In Definitions 2 and 3, $k = 2$ and $b = 2$ at the given conditions; condition (9) will be met at the maximum integer $y = 7$. According to Statement 5, 2 multiplexers $\left(2^2\right)^7$ -in-1 can be implemented on the given FPGA. This allows implementing NPF(18) over $GF(2^2)$ according to (10) when using eight FPGAs XC7V585T, of which one is used for the multiplexer, while 7 ones are for implementing NPF(11) over $GF(2^2)$ (see Example 1).

Example 4. Let us consider the implementation of k multiplexers $\left(2^k\right)^y$ -in-1 on Virtex-7 FPGA, XC7V585T, for which all the conditions from Example 1 are met. It is required to determine the number of variables x that serve for multiplexing a certain amount of the values of $\psi_{i_{m+1}\ldots i_x}\left(x_1,\ldots,x_m\right)$, computed over $GF(2^3)$ according to (10) at the given $i_j = \overline{0,\, 2^k - 1}$, $j = \overline{m+1,\, x}$ an at condition (9) being met (9).

In Definitions 2 and 3, $k = 3$ и $b = 2$ at the given conditions; condition (9) will be met at the maximum integer $y = 3$. According to Statement 5, 3 multiplexers $\left(2^3\right)^3$ -in-1 can be implemented on the given FPGA. This allows implementing NPF(10) over $GF(2^3)$ according to (10) when using four FPGAs XC7V585T, of which one is used for the multiplexer, while three ones are for implementing NPF(7) over $GF(2^3)$ (see Example 1).

Note 2. In Examples 1-4 above, IOBs were the limiting factor in implementing NPFs and multiplexers on FPGA XC7V585T.

Considering that modern Russian MPSs include up to 1.5 thousand PLDs/FPGAs, this amount is quite acceptable for implementing on them up to 187 NPFs(18) over $GF(2^2)$ and up to 375 NPFs(10) over $GF(2^3)$.

Number of NPF variables can be increased by multiplexing many values of $\psi_{i_{m+1}\ldots i_q}\left(x_1,\ldots,x_m\right)$ at predefined $i_j = \overline{0,\, 2^k - 1}$, $j = \overline{m+1,\, q}$, when using more than one FPGA. Based on Statements 4 and 6, we determined

Statement 6. For the distributed computation of NPF(q) shown in (1) and represented as (10), it is required $2^u - 1$ FPGAs implementing GBF(a), $u = \left](q - m - x)/y\right[$.

V. Conclution

We obtained the estimates of the complexity of implementing the nonlinear polynomial functions of a preset number of variables on a programmable-architecture MPS. The estimates depend on both the PLD characteristics, i.e., the number of elements – generators of Boolean functions of a given number of variables, *D*-triggers, and input-output blocks, and the characteristics of the nonlinear polynomial functions to be implemented, i.e., numbers of variables and nonzero coefficients. The above estimates are defined for both the number of PLD elements and the number of PLD crystals included in the MPS, both existing and potential ones.

References

[1] V. V. Voevodin, Parallelnyye vychisleniya [Parallel Computing], BKhV-Peterburg, Saint Petersburg, 2002, 600 p. (in Russian). [Online]. Available: https://booksee.org/book/589570.

[2] M.O. Kuzelin, D.A. Knyshev, and V. Yu. Zotov, Sovremennye semeistva PLIS firmy Xilinx: spravochoye posobiye [Modern FPGA Families from Xilinx: a Reference Book], Goriachaia liniya-Telekom Publ., Moscow, 2004, 440 p. (in Russian).

[3] I. A. Kalyaev, I. I. Levin, A. I. Dordopulo and L. M. Slasten, “FPGA-based reconfigurable computer systems,” 2013 Science and Information Conference, London, UK, 2013, pp. 148-155. [Online]. Available: https://ieeexplore.ieee.org/document/6661730.

[4] A. I. Dordopulo, E. A. Semernikov, I. A. Kalyaev, and I. I. Levin, “High-Performance Reconfigurable Computer Systems of New Generation,” Numerical methods and programming, vol. 12, pp. 82-89, Oct.–Dec. 2011. (in Russian). [Online]. Available: https://en.num-meth.ru/index.php/journal/article/view/489.

[5] FPGA Leadership across Multiple Process Nodes/ Xilinx Inc. Cop. 2021. URL: https://www.xilinx.com/products/silicon-devices/fpga.html.

[6] 7-Series Produkt Selection Guade/ Xilinx Inc. Cop. 2014-2020. URL: https://www.xilinx.com/support/documentation/selection-guides/7-series-product-selection-guide.pdf.

[7] V. A. Raikhlin, I. S. Vershinin, R. F. Gibadullin, and S. V. Pystogov, “Reliable recognition of masked binary matrices. Connection to information security in map systems,” Lobachevskii J Math., vol. 34, pp. 319–325, Dec. 2013. [Online]. Available: https://doi.org/10.1134/S1995080213040112.

[8] I. S. Vershinin, R. F. Gibadullin, S. V. Pystogov, and V. A. Raikhlin, “Associative Steganography. Durability of Associative Protection of Information,” Lobachevskii J Math., vol. 41, pp. 440–450, July 2020. [Online]. Available: https://doi.org/10.1134/S1995080220030191.

[9] V. M. Zakharov, and S. V. Shalagin, Raspredelennoye vychisleniye nelineynykh mnogochlenov nad polem Galua v arkhitekture FPGA [Distributed Computation of Non-Linear Polynomials over the Galois Field in the FPGA Architecture]. *Herald of Technological University*, vol. 21, no. 11, pp. 146-149, Nov. 2018. (in Russian). [Online]. Available: https://www.elibrary.ru/item.asp?id=36815900.

[10] S. Yu. Melnikov, and K. E. Samouylov, “Probabilistic functions and statistical equivalence of binary shift registers with random Markov

input," CEUR Workshop Proceedings, vol. 2639, pp. 93-99. July 2020. [Online]. Available: http://ceur-ws.org/Vol-2639/paper-08.pdf.

[11] Z. Gizatullin, R. Gizatullin and V. Drozdikov, "Research of Noise Immunity of Computer Equipment of Control Systems Under Action of Pulsed Magnetic Field," 2019 International Russian Automation Conference (RusAutoCon), Sochi, Russia, 2019, pp. 1-5, doi: 10.1109/RUSAUTOCON.2019.8867658. [Online]. Available: https://ieeexplore.ieee.org/document/8867658.

[12] R. F. Gibadullin, G. A. Baimukhametova and M. Y. Perukhin, "Service-Oriented Distributed Energy Data Management Using Big Data Technologies," 2019 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), Sochi, Russia, 2019, pp. 1-7, doi: 10.1109/ICIEAM.2019.8743064. [Online]. Available: https://ieeexplore.ieee.org/document/8743064.

[13] V. Pesoshin, A. Gumirov, V. Kuznetsov and D. Shirshova, "Generators of the Binary Inverse-Segment Pseudo-Random Sequences," 2018 IEEE East-West Design & Test Symposium (EWDTS), Kazan, 2018, pp. 1-8, doi: 10.1109/EWDTS.2018.8524819. [Online]. Available: https://ieeexplore.ieee.org/document/8524819.

[14] V. M. Zakharov and S. V. Shalagin, "Executing discrete orthogonal transformations based on computations on the Galois field in the FPGA architecture," 2016 International Siberian Conference on Control and Communications (SIBCON), Moscow, Russia, 2016, pp. 1-4, doi: 10.1109/SIBCON.2016.7491652. [Online]. Available: https: //ieeexplore.ieee.org/document/7491652

[15] S. V. Shalagin, Realizaciya cifrovyh ustrojstv v arhitekture PLIS/FPGA pri ispol'zovanii raspredelennyh vychislenij v polyah Galua [Implementing digital devices in FPGA architecture when using distributed computing in Galois fields], KNRTU-KAI Press, Kazan, 2016. 228 p. (in Russian). [Online]. Available: https://www.elibrary.ru/item.asp?id=27287609.