Secure Pseudo-Random Linear Binary Sequences Generators Based on Arithmetic Polynoms

Oleg Finko and Sergey Dichenko

Abstract We present a new approach to construction of pseudo-random binary sequences (PRBS) generators for the purpose of cryptographic data protection, secured from the perpetrator's attacks, caused by generation of masses of hardware errors and faults. The new method is based on the use of linear polynomial arithmetic for the realization of systems of boolean characteristic functions of pseudo-random sequences (PRS) generators. "Arithmetization" of systems of logic formulas has allowed to apply mathematical apparatus of residue systems for multisequencing of the process of PRS generation and organizing control of computing errors, caused by hardware faults. This has guaranteed high security of PRS generator's functioning and, consequently, security of tools for cryptographic data protection based on those PRSs.

Keywords Cryptographic data protection · Pseudo-random binary sequences · Residue number systems

1 Introduction

Pseudo-random linear sequences generators play an important role in building of communication with cryptographic data protection [1, 2]. From the list of known attacks on information security is important type of attacks, based on the generation of hardware errors and functioning of the nodes forming the binary PRS [3]. To ensure the required level of interference and fault tolerance of digital devices

O. Finko (✉) · S. Dichenko
Institute of Computer Systems and Information Security of Kuban State Technological University, Moskovskaya St. 2, Krasnodar, Russia
e-mail: ofinko@yandex.ru; ofinko@member.ams.org

S. Dichenko
e-mail: dichenko.sa@yandex.ru

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_24

developed many methods, the most common of which are backup methods and methods of error-correcting coding [4]. However, allocation methods do not provide the required levels of fault tolerance for restrictions on hardware costs, and methods of error-correcting coding is not adapted to the specifics of construction and operation means of data protection (MDP), in particular, the generators of the PRS.

2 Analysis of Attacks Based on Hardware Faults Generation

Currently, the following types of attacks on sites of formation of binary PRS are considered (attack on) [5]:

- Analysis of results of power consumption measurements;
- Analysis of results of operations performance duration;
- Analysis of accidental hardware faults;
- Analysis of intentionally generated hardware faults, etc.

The last two types of faults are not investigated enough currently and thus are threatening to the information security of the functioning of modern and perspective MDP. The origin of those attacks lies in the use of thermal, high frequency, ionizing, and other types of external influences onto MDP for the purpose of creation of masses of faults in hardware functioning by initializing of computing errors.

Hardware attacks can be divided into two classes:

1. **Direct hardware attacks** The consequences of those attacks are failures of data protection tools. There is a method of analysis of the consequences of those failures. These types of attacks mean that in distortion in the certain places of algorithm of transformation, which results in computing errors. Those errors can lead, for example, to repeated generation of the elements of PRS or in generation of faulty elements of PRS, which is unacceptable.
2. **Attacks on postfailure recovery means** Some systems do not recovery means. If the system protection is destroyed, it is impossible to restore the operational mode. That is why such systems need to have means of protection against attacks of the malefactor and to support the possibility of updating the security system without stopping the program running.

Attacks, based on errors generation by means of external influence are highly efficient for the majority of currently known and used algorithms of PRS generation. It is known that probability of error generation is proportional to the time corresponding registers has been affected by the radiation, if the registers are in favorable condition for error occurrence, and to the quantity of bits, in which the error occurrence is expected. The most widely used and proven means of creating PRS are algorithms and structures—Linear feedback shift register (LFSR)—of PRS generation, based on the use of feedback functions of logic [1, 2].

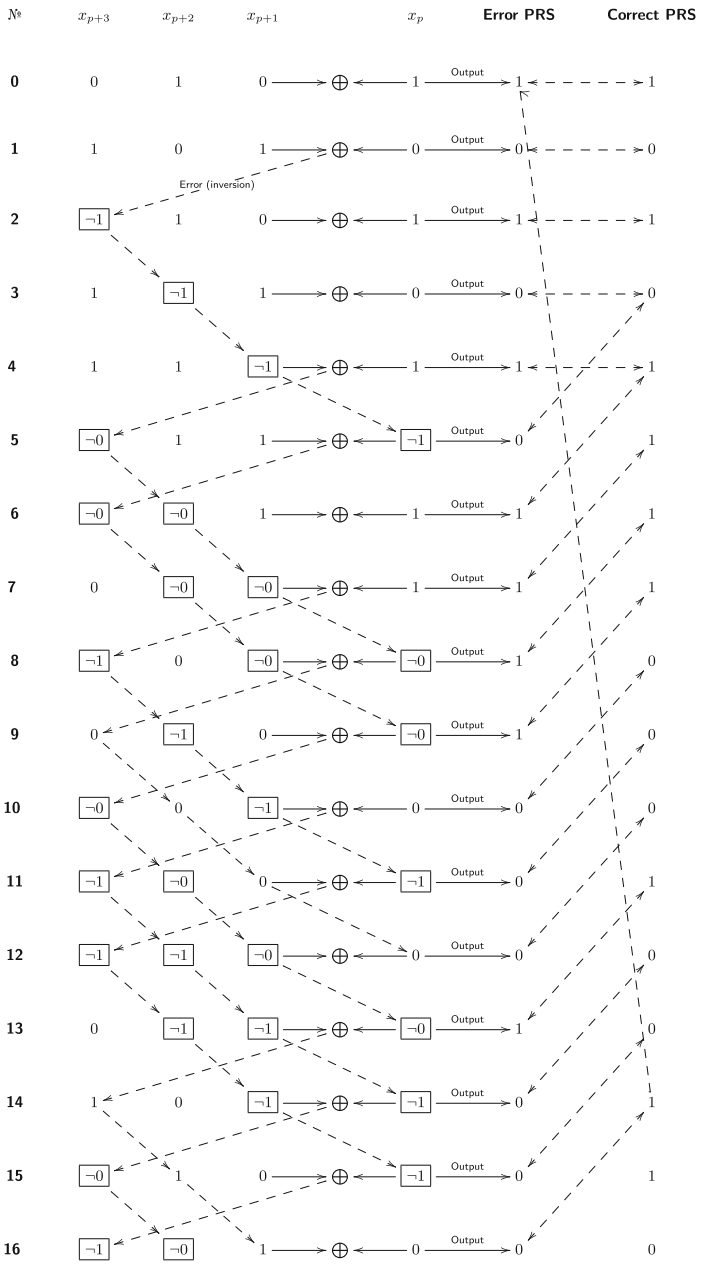

Fig. 1 Example of operation of the LFSR when an error occurs (¬x—logical inversion x)

The structure of LFSR is determined by the forming polynomial:

$$D(\chi) = \chi^{\tau} + \chi^{t_l} + \cdots + \chi^{t_2} + \chi^{t_1} + 1,$$

where $\tau, t_i \in N$ and characteristic equation based on it:

$$\begin{aligned} x_{p+\tau} &= x_p \oplus x_{p+t_1} \oplus x_{p+t_2} \oplus \cdots \oplus x_{p+t_l} \\ &= c_0 x_p \oplus c_1 x_{p+1} \oplus \cdots \oplus c_{\tau-2} x_{p+\tau-2} \oplus c_{\tau-1} x_{p+\tau-1}, \end{aligned} \tag{1}$$

where $x_p, c_i \in \{0, 1\}$; $p \in N$; $i = 0, 1, \ldots, \tau - 1$; $c_{i \in \{0, t_1, t_2, \ldots, t_l\}} = 1$.

In linear algebra the next element of PRS $x_{p+\tau}$ is calculated as the following multiplication:

$$\begin{Vmatrix} x_{p+1} \\ x_{p+2} \\ \cdots\cdots \\ x_{p+\tau-1} \\ x_{p+\tau} \end{Vmatrix}^{\top} = \begin{Vmatrix} 0 & 1 & \ldots\ldots & 0 & 0 \\ & & \cdots\cdots & & \\ 0 & 0 & \ldots\ldots & 1 & 0 \\ 0 & 0 & \ldots\ldots & 0 & 1 \\ c_0 & c_1 & \ldots\ldots & c_{\tau-2} & c_{\tau-1} \end{Vmatrix}^{\top} \cdot \begin{Vmatrix} x_p \\ x_{p+1} \\ \cdots\cdots \\ x_{p+\tau-2} \\ x_{p+\tau-1} \end{Vmatrix}^{\top}.$$

When the described attack is performed the conditions arise for PRS modification or its repeated generation. The effect of repeated generation of a site of PRS is explained by means of Fig. 1 (the forming polynomial: $D(\chi) = \chi^4 + \chi + 1$; the characteristic equation: $x_{p+4} = x_{p+1} \oplus x_p$; the initial conditions: $x_p = 1, x_{p+1} = 0$, $x_{p+2} = 1, x_{p+3} = 0$).

Thus, those attacks, which are based on creating the conditions under which mass hardware errors occur, are threatening for MDP. One of the ways of solving this problem is development of methods for increasing the reliability of the functioning of sites of data protection tools, mostly subjected to attacks of the described type, in particular the sites of forming of the encryption algorithm (cipher), based on PRS generation.

3 Analysis of Methods for Reliable Binary PRS Generation

Currently, the required level of functional reliability of the sites of binary PRS generation is reached both by using excessive devices (reservation) and timely access by various repetitions of the calculations. In digital schemotechnics there are solutions known based on the use of methods of error-correction coding [4]. In order to use those methods for PRS generators it is necessary preliminary to solve the issue multisequencing the process of PRS calculations. The solution is based on the use of classic parallel algorithms of recursion [6].

For example, for the characteristic equation:

$$x_{p+\tau} = x_{p+t} \oplus x_p, \tag{2}$$

corresponding to treen $D(\chi) = \chi^\tau + \chi^t + 1$, it is possible to build a system of characteristic equations:

$$\begin{cases} x_{q,\tau-1} = x_{q-1,\tau-1} \oplus x_{q-1,\tau+t-1}, \\ x_{q,\tau-2} = x_{q-1,\tau-2} \oplus x_{q-1,\tau+t-2}, \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ x_{q,1} = x_{q-1,1} \oplus x_{q-1,t+1}, \\ x_{q,0} = x_{q-1,0} \oplus x_{q-1,t}. \end{cases}$$

Similarly, for the general Eq. (1):

$$\begin{cases} x_{q,\tau-1} = c_0^{(\tau-1)} x_{q-1,0} \oplus c_1^{(\tau-1)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(\tau-1)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(\tau-1)} x_{q-1,\tau-1}, \\ x_{q,\tau-2} = c_0^{(\tau-2)} x_{q-1,\,0} \oplus c_1^{(\tau-2)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(\tau-2)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(\tau-2)} x_{q-1,\tau-1}, \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ x_{q,1} = c_0^{(1)} x_{q-1,0} \oplus c_1^{(1)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(1)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(1)} x_{q-1,\tau-1}, \\ x_{q,0} = c_0^{(0)} x_{q-1,0} \oplus c_1^{(0)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(0)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(0)} x_{q-1,\tau-1}, \end{cases} \tag{3}$$

where $c_i^{(j)} \in \{0, 1\}$ $(i, j = 0, 1, \ldots, \tau - 1)$. The principle of parallel lasing elements PRS based on (3) is illustrated by a graph (see Fig. 2).

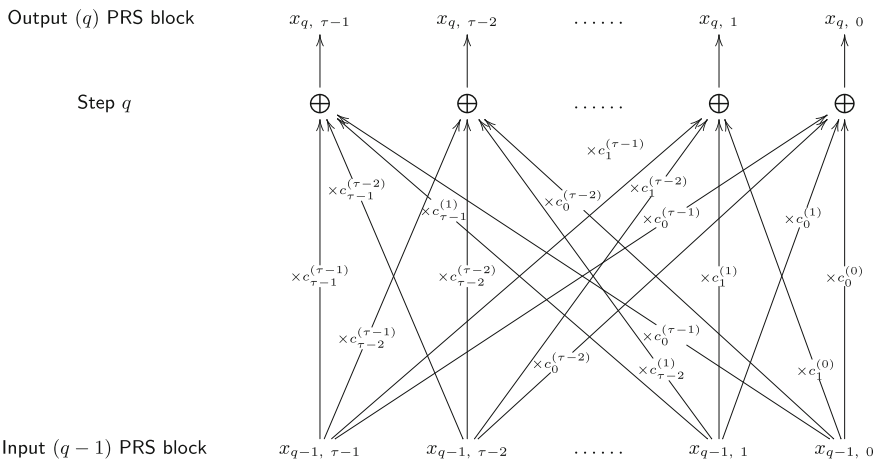

Fig. 2 Graph generating elements parallel PRS based on (3)

System (3) forms an information matrix:

$$\mathbf{G_{Inf}} = \begin{Vmatrix} c_0^{(\tau-1)} & c_1^{(\tau-1)} & \ldots\ldots & c_{\tau-2}^{(\tau-1)} & c_{\tau-1}^{(\tau-1)} \\ c_0^{(\tau-2)} & c_1^{(\tau-2)} & \ldots\ldots & c_{\tau-2}^{(\tau-2)} & c_{\tau-1}^{(\tau-2)} \\ & & \ldots\ldots\ldots\ldots & & \\ c_0^{(1)} & c_1^{(1)} & \ldots\ldots & c_{\tau-2}^{(1)} & c_{\tau-1}^{(1)} \\ c_0^{(0)} & c_1^{(0)} & \ldots\ldots & c_{\tau-2}^{(0)} & c_{\tau-1}^{(0)} \end{Vmatrix}^{\top}.$$

Thus we obtain the qth block of the PRS:

$$\mathbf{X}_q = \mathbf{G_{Inf}} \cdot \mathbf{X}_{q-1},$$

where

$$\mathbf{X}_q = \left[x_{q,\tau-1}\ x_{q,\tau-2}\ \ldots\ x_{q,1}\ x_{q,0}\right]^{\top},$$
$$\mathbf{X}_{q-1} = \left[x_{q-1,\tau-1}\ x_{q-1,\tau-2}\ \ldots\ x_{q-1,1}\ x_{q-1,0}\right]^{\top}.$$

Adding to the system (3) checking the equations: $\mathbf{G_{Gen}}$, consisting of the information and the check matrix by adding (3) validation expressions:

$$\begin{cases} x_{q,\tau-1} = c_0^{(\tau-1)} x_{q-1,0} \oplus c_1^{(\tau-1)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(\tau-1)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(\tau-1)} x_{q-1,\tau-1}, \\ x_{q,\tau-2} = c_0^{(\tau-2)} x_{q-1,\,0} \oplus c_1^{(\tau-2)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(\tau-2)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(\tau-2)} x_{q-1,\tau-1}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_{q,1} = c_0^{(1)} x_{q-1,0} \oplus c_1^{(1)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(1)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(1)} x_{q-1,\tau-1}, \\ x_{q,0} = c_0^{(0)} x_{q-1,0} \oplus c_1^{(0)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(0)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(0)} x_{q-1,\tau-1}, \\ x_{q,r-1}^{*} = a_0^{(r-1)} x_{q-1,0} \oplus a_1^{(r-1)} x_{q-1,1} \oplus \cdots \oplus a_{\tau-2}^{(r-1)} x_{q-1,\tau-2} \oplus a_{\tau-1}^{(r-1)} x_{q-1,r-1}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_{q,0}^{*} = a_0^{(0)} x_{q-1,0} \oplus a_1^{(0)} x_{q-1,1} \oplus \cdots \oplus a_{\tau-2}^{(0)} x_{q-1,\tau-2} \oplus a_{\tau-1}^{(0)} x_{q-1,\tau-1}, \end{cases}$$

where r—the number of redundant symbols used linear code, $a_i^{(j)} \in \{0, 1\}$ $(i = 0, 1, \ldots, \tau - 1;\ \ j = 0, \ldots, r - 1)$.

A generator matrix takes the form:

$$\mathbf{G_{Gen}} = \begin{Vmatrix} c_0^{(\tau-1)} & c_1^{(\tau-1)} & \ldots\ldots & c_{\tau-2}^{(\tau-1)} & c_{\tau-1}^{(\tau-1)} \\ c_0^{(\tau-2)} & c_1^{(\tau-2)} & \ldots\ldots & c_{\tau-2}^{(\tau-2)} & c_{\tau-1}^{(\tau-2)} \\ & & \ldots\ldots\ldots\ldots & & \\ c_0^{(1)} & c_1^{(1)} & \ldots\ldots & c_{\tau-2}^{(1)} & c_{\tau-1}^{(1)} \\ c_0^{(0)} & c_1^{(0)} & \ldots\ldots & c_{\tau-2}^{(0)} & c_{\tau-1}^{(0)} \\ a_0^{(r-1)} & a_1^{(r-1)} & \ldots\ldots & a_{\tau-2}^{(r-1)} & a_{\tau-1}^{(r-1)} \\ & & \ldots\ldots\ldots\ldots & & \\ a_0^{(0)} & a_1^{(0)} & \ldots\ldots & a_{\tau-2}^{(0)} & a_{\tau-1}^{(0)} \end{Vmatrix}^{\top}.$$

Then the qth block of the PRS with the control numbers (linear block code):

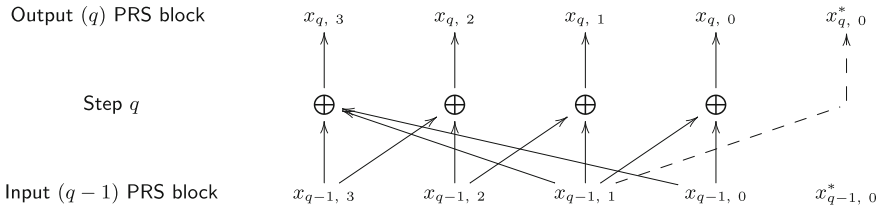

Fig. 3 Example graph parallel generation elements PRS (the characteristic equation: $x_{p+4} = x_{p+1} \oplus x_p$) error control computations (parity control)

$$\mathbf{X}^*_q = \left[x_{q,\tau-1}\ x_{q,\tau-2} \cdots x_{q,1}\ x_{q,0}\ x^*_{q,r-1} \cdots x^*_{q,0}\right]^\top$$

is calculated by:

$$\mathbf{X}^*_q = \mathbf{G}_{\mathbf{Gen}} \cdot \mathbf{X}_{q-1}.$$

Procedure error-correcting decoding is performed using the known rules [4]. The application of linear redundant codes and methods "hot" standby is not the only option for the implementation of functional diagnostics and fault tolerance of digital devices. Example graph parallel generation elements PRS error control computations is shown in Fig. 3.

Important advantages for these purposes have redundant arithmetic codes, in particular, so-called AN-codes and residue number systems (RNS) codes. The application of these codes to monitor logical data types and fault tolerance implementing devices became possible with the introduction of logical operations arithmetic expressions [7], in particular linear numerical polynomials (LNP) and modular forms [8].

4 Error Control Operation of the PRS Generators, Based on "Arithmetization" Logical Account

At the end of the last century there was formed a new direction parallel logic computation by the arithmetic (numeric) polynomials [7]. In particular received position "Modular arithmetic parallel logic computation" of the unification of the theoretical foundations of RNS [9–11] and theoretical foundations of parallel logic computation by the arithmetic of polynomials. The objective of the association is to use advantages of RNS, i.e., parallelization arithmetic, error control calculations [12] in real time and ensure high availability of computing equipment in the field of

parallel logical account. In the following, these provisions were developed in various aspects, in particular, toward the implementation of cryptographic functions [13, 14]. In particular, it was considered parallel generators PRS based, in general, nonlinear (canonical) arithmetic polynomials. Use of LNP proposed by Prof. V.D. Malyugin [7] for the construction of parallel generators PRS possible to reduce the maximum length of realizing polynomial to a value of $n+1$, where n—number of arguments of a Boolean function implemented [14]. In this paper, this method is used as the basis for the construction of safe (self-checking, fault-tolerant) generators on the basis of the excess bandwidth RNS.

It is known [15] that the qth block of land PRS can be represented by a single LNP. The system of characteristic Eq. (3) must submit, as a system of Boolean functions, which in turn must be converted into a system:

$$\begin{cases} L_{\tau-1}(\mathbf{X}_{q-1}) = g_1^{(\tau-1)} x_{q-1,0} + g_2^{(\tau-1)} x_{q-1,1} + \cdots + g_\tau^{(\tau-1)} x_{q-1,\tau-1}, \\ L_{\tau-2}(\mathbf{X}_{q-1}) = g_1^{(\tau-2)} x_{q-1,0} + g_2^{(\tau-2)} x_{q-1,1} + \cdots + g_\tau^{(\tau-2)} x_{q-1,\tau-1}, \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ L_0(\mathbf{X}_{q-1}) = g_1^{(0)} x_{q-1,0} + g_2^{(0)} x_{q-1,1} + \cdots + g_\tau^{(0)} x_{q-1,\tau-1}, \end{cases}$$

where $g_j^{(i)}$ (here and then) takes the value "0" or "1" depending on the entry in the ith LNP $x_{q-1,j}$; $i, j = 0, 1, \ldots, \tau - 1$.

The result of the calculation of i-LNP system appears to be a binary word of length $l_i = \lfloor \log(\sum_{j=\tau-1}^{0} g_j^{(i)}) \rfloor + 1$, where $\lfloor a \rfloor$—the largest integer. Calculated total LNP:

$$\begin{aligned} L(\mathbf{X}_{q-1}) &= L_{\tau-1}(\mathbf{X}_{q-1}) + 2^{\gamma_1} L_{\tau-2}(\mathbf{X}_{q-1}) + \cdots + 2^{\gamma_{\tau-1}} L_0(\mathbf{X}_{q-1}) \\ &= g_1^{(\tau-1)} x_{q-1,0} + g_2^{(\tau-1)} x_{q-1,1} + \cdots + g_\tau^{(\tau-1)} x_{q-1,\tau-1} \\ &\quad + 2^{\gamma_1} (g_1^{(\tau-2)} x_{q-1,0} + g_2^{(\tau-2)} x_{q-1,1} + \cdots + g_\tau^{(\tau-2)} x_{q-1,\tau-1}) \\ &\quad + \cdots + 2^{\gamma_{\tau-1}} (g_1^{(0)} x_{q-1,0} + g_2^{(0)} x_{q-1,1} + \cdots + g_\tau^{(0)} x_{q-1,\tau-1}) \\ &= h_1 x_{q-1,0} + h_2 x_{q-1,1} + \cdots + h_\tau x_{q-1,\tau-1}, \end{aligned}$$

where $\gamma_k = \sum_{i=0}^{k-1} (l_i + 1)$, $k = 1, 2, \ldots, \tau - 1$; $h_j \in Z$, or

$$L(\mathbf{X}_{q-1}) = \sum_{i=1}^{\tau} h_i x_{q-1,i-1}. \tag{4}$$

The final result is formed by implementing operator masking $\Xi^{\varphi}\{U\}$, which is used to determine the values of the φth Boolean function representation $U = (b_v \ldots b_\varphi \ldots b_2 b_1)_2$ (record $(\ldots)_2$ means representing a nonnegative U in a binary number), that is, $\Xi^{\varphi}\{U\} = b_\varphi$.

In RNS a nonnegative coefficient LNP (4) h_j is uniquely represented by a set of residues on the grounds RNS ($m_1, m_2, \ldots, m_n < m_{n+1} < \cdots < m_k$—pairwise simple):

$$h_j = (\alpha_1, \alpha_2, \ldots, \alpha_n, \alpha_{n+1}, \ldots, \alpha_k)_{\mathrm{MA}}, \tag{5}$$

where $\alpha_t = |h_j|_{m_t}; t = 1, 2, \ldots, n, \ldots, k$; $|\centerdot|_m$—the smallest nonnegative deduction number $\centerdot$ on the modulo m. Operating range $M_n = m_1 m_2 \ldots m_n$ must meet $M_n > 2^s$, where $s = \sum_{1 \le \varepsilon \le \tau} l_\varepsilon$—the number of binary bits required to represent the result of a calculation LNP (4).

The remains $\alpha_1, \alpha_2, \ldots, \alpha_n$ are informational, and $\alpha_{n+1}, \ldots, \alpha_k$—are control. RNS in this case is called the extended and covers the complete set of states represented all k residues. This area is full range RNS $[0, M_k)$, where $M_k = m_1 m_2 \ldots m_n m_{n+1} \ldots m_k$, and consists of the operating range $[0, M_n)$, defined information bases RNS, and range identified redundant bases $[M_n, M_k)$, unacceptable region for the results of a calculation. This means that operations on numbers h_j are in the range $[0, M_k)$. Therefore, if the result of the operation RNS beyond M_n, it should output error calculation.

Consider RNS specified grounds $m_1, m_2, \ldots, m_n, m_{n+1}$. Each coefficient LNP h_j can be written as (5) and get redundant code RNS represented by the LNP system:

$$\begin{cases} U^{(1)} = L^{(1)}(\mathbf{X}_{q-1}) = \alpha_1^{(1)} x_{q-1,0} + \alpha_2^{(1)} x_{q-1,1} + \cdots + \alpha_\tau^{(1)} x_{q-1,\tau-1}, \\ U^{(2)} = L^{(2)}(\mathbf{X}_{q-1}) = \alpha_1^{(2)} x_{q-1,0} + \alpha_2^{(2)} x_{q-1,1} + \cdots + \alpha_\tau^{(2)} x_{q-1,\tau-1}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ U^{(n)} = L^{(n)}(\mathbf{X}_{q-1}) = \alpha_1^{(n)} x_{q-1,0} + \alpha_2^{(n)} x_{q-1,1} + \cdots + \alpha_\tau^{(n)} x_{q-1,\tau-1}, \\ U^{(n+1)} = L^{(n+1)}(\mathbf{X}_{q-1}) = \alpha_1^{(n+1)} x_{q-1,0} + \alpha_2^{(n+1)} x_{q-1,1} + \cdots \\ +\alpha_\tau^{(n+1)} x_{q-1,\tau-1}. \end{cases} \tag{6}$$

Substituting in (6) values of RNS residue on the appropriate grounds for each coefficient (4) and the values of the variables $x_{q-1,0}, \ldots, x_{q-1,\tau-1}$, get the values of LNP system (6), where $U^{(1)}, U^{(2)}, \ldots, U^{(n)}, U^{(n+1)}$—nonnegative integer. In accordance with the Chinese remainder theorem solve the system of equations:

$$\begin{cases} U^* = |U^{(1)}|_{m_1}, \\ U^* = |U^{(2)}|_{m_2}, \\ \ldots\ldots\ldots\ldots\ldots \\ U^* = |U^{(n)}|_{m_n}, \\ U^* = |U^{(n+1)}|_{m_{n+1}}. \end{cases} \tag{7}$$

Since $m_1, m_2, \ldots, m_n, m_{n+1}$ are pairwise prime, then the only solution of (7) gives the expression:

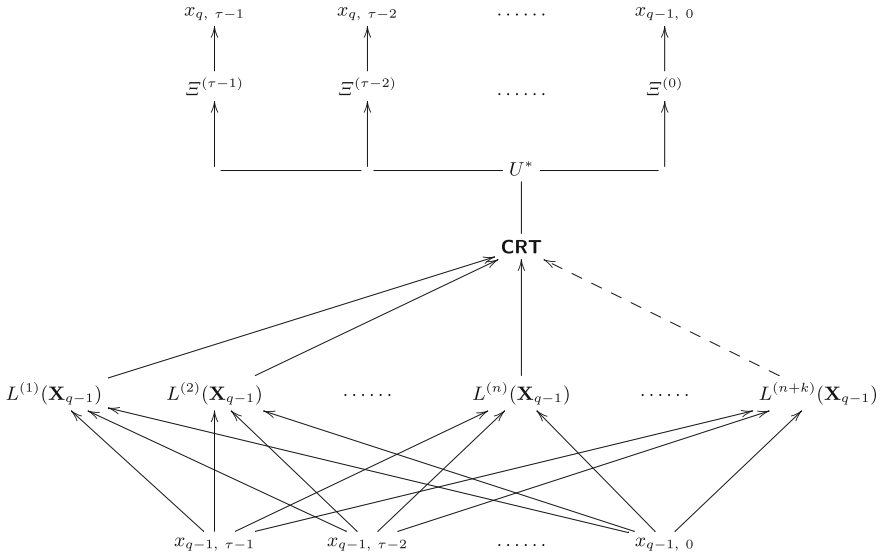

Fig. 4 Graph of parallel generation PRS based on the Chinese remainder theorem (CRT)

$$U^* = \left| \sum_{s=1}^{n+1} M_{s,n+1} \mu_{s,n+1} U^{(s)} \right|_{M_{n+1}}, \tag{8}$$

where $M_{s,n+1} = \dfrac{M_{n+1}}{m_s}$, $\mu_{s,n+1} = |M_{s,n+1}^{-1}|_{m_s}$, $M_{n+1} = \prod_{s=1}^{n+1} m_s$.

Graph parallel generation PRS based on (8) is shown in Fig. 4. The occurrence of the result of the calculation (8) in the range (control expression):

$$0 \le U^* < M_n,$$

means the absence of detectable errors of calculations.

5 Reconfiguration of Equipment

Restore reliable operation of the generator of the PRS in the case of long-term failure is possible by correcting an error or reconfiguration of equipment generator (active redundancy). The first option is unacceptable because it does not guarantee no penetration of undetectable errors in the result of the encryption. By methods of modular redundant coding is made possible to apply a variant of the reconfiguration of the equipment by excluding from the operation of the failed equipment.

Table 1 Calculation table orthogonally bases and modules RNS

j	$B_{1,j}$	$B_{2,j}$	$\cdots$	$B_{n+2,j}$	M_j
1	0	$\frac{M_1\mu_{2,1}}{m_2}$	$\cdots$	$\frac{M_1\mu_{n+2,1}}{m_{n+2}}$	$m_2m_3\ldots m_{n+2}$
2	$\frac{M_2\mu_{1,2}}{m_1}$	0	$\cdots$	$\frac{M_2\mu_{n+2,2}}{m_{n+2}}$	$m_1m_3\ldots m_{n+2}$
.....	.....	.....	...	.....	.....
$n+2$	$\frac{M_{n+2}\mu_{1,n+2}}{m_1}$	$\frac{M_{n+2}\mu_{2,n+2}}{m_2}$	$\cdots$	0	$m_1m_2\ldots m_{n+1}$

After localization of the faulty equipment—for example—a single channel operation RNS, the reconfiguration operation is performed by the calculation U^* from the system:

$$\begin{cases} U^* = |\widetilde{U}^{(1)}|_{m_1}, \\ \ldots\ldots\ldots\ldots\ldots \\ U^* = |\widetilde{U}^{(n)}|_{m_n}, \\ U^* = |\widetilde{U}^{(n+1)}|_{m_{n+1}}, \\ U^* = |\widetilde{U}^{(n+2)}|_{m_{n+2}} \end{cases}$$

on the modules corresponding to the serviceable equipment of the computer:

$$U^* = |\widetilde{U}^{(1)}B_{1,j} + \widetilde{U}^{(2)}B_{2,j} + \cdots + \widetilde{U}^{(n+2)}B_{n+2,j}|_{M_j},$$

where $\widetilde{U}^{(i)}$—are numbers that may contain errors; $B_{i,j}$—orthogonal bases; $i, j = 1, 2, \ldots, n+2$; $i \neq j$; $B_{i,j} = \frac{M_j\mu_{i,j}}{m_i}$; $M_j = \frac{M_{n+2}}{m_j}$; $\mu_{i,j}$ is calculated from the comparison: $\frac{M_j\mu_{i,j}}{m_i} \equiv 1 \pmod{m_i}$. Compiled Table 1 contains the values of the orthogonal bases and modules of the system for the occurrence of a single error for each base RNS.

6 Conclusion

It is known that the use of RNS already with two redundant bases allows us to provide a level of fault tolerance modular transmitter that exceeds the tolerance provided by the method of rorovana equipment. These redundant hardware costs are reduced from 200 % (triple) up to 30–40 % (when using RNS) [16]. At the same time it should be noted that the amount of hardware, PRS generator operating in accordance obtained by the method, may exceed the hardware failover LFSR, built in accordance with traditional solutions. So you should make a fundamentally new level of functional flexibility of the designed generator PRS able to implement many other cryptographic functions, which are time-varying, without rebuilding the structure.

This allows for the implementation of the device not only in programmable logic integrated circuit, but also high-tech large custom integrated circuits, in particular used for the implementation of number theoretic transformations in the field of digital signal processing.

The implementation of the PRS generators using LNP and redundant RNS allows to obtain a new class of solutions aimed at the safe implementation of the logical cryptographic functions, in particular parallel generators PRS. This is provided as a functional control equipment (in real time), and its fault tolerance through reconfiguration of the structure of the evaluator in the process of its degradation. Classic LFSR considered in the present work, is the basis and more complex, for example, combining generators PRS. Use of the implementation of the PRS generator modular arithmetic provides the possibility of applying the proposed solutions in the hybrid cryptosystems (including asymmetric) [14]. When this arithmetic calculator that supports the implementation of asymmetric cryptographic algorithms may be used to implement systems of Boolean functions (elements PRS).

References

1. Forouzan, B.A.: Cryptography and Network Security. McGraw Hill (2008)
2. Schneier, B.: Applied Cryptography. Wiley, New York (1996)
3. Yang, B., Wu, K., Karri, R.: Scan based side channel attack on data encryption standard. Report **2004**(324), 114–116 (2004)
4. Hetagurov, J.A., Prudnaya, Y.P.: Improving the reliability of digital devices redundant coding methods. Energiya, Moscow (1974)
5. Kelsey, J.: Protocol interactions and the chosen protocol attack. Security protocols. In: 5th International Workshop, pp. 91–104, Springer New York. (1996)
6. Ortega, J.M.: Introduction to Parallel & Vector Solution of Linear Systems. Plenum Press, New York (1988)
7. Shmerko, V.P.: Malyugin's theorems: a new concept in logical control, VLSI design, and data structures for new technologies. Autom. Remote. Control. **65**(6), 893–912 (2004). June
8. Finko, O.A.: Large systems of boolean functions: realization by modular arithmetic methods. Autom. Remote. Control. **65**(6), 871–892 (2004). June
9. Garner, H.L.: Number systems and arithmetic. Adv. Comput. **6**, 131–194 (1965)
10. Omondi, A., Premkumar, B.: Residue Number System: Theory and Implementation. Imperial College Press, London (2007)
11. Soderstrand, M.A., Jenkins, W.K., Jullien, G.A., Tailor, F.J.: Residue Number System Arithmetic: Modern Application in Digital Signal Processing. IEEE Press, New York (1986)
12. Jenkins, W.K.: The design of error checkers for self-checking residue number arithmetic. IEEE Trans. Comput. **4**, 388–396 (1983)
13. Finko, O.A., Vishnevsky, A.K.: Parallel realization of systems of substitutions by numerical polynoms. In: Papers of the 5th International Conference Parallel Computing and Control Problems, pp. 935–943. Moscow (2010)
14. Finko, O.A., Vishnevsky, A.K.: Standard function hybrid cryptosystem arithmetic and logical multinomial realization. Theory and Techniques of Radio, pp. 32–38. Voronezh (2011)
15. Finko, O.A., Dichenko, S.A., Eliseev, N.I.: Error function generator binary PRS control implemented on arithmetic polynomials. St. Petersburg State Polytechnical University J. Comput. Sci. Telecommun. Control Syst. **176**(4), 142–149 (2013)
16. Krasnobaev, V.A.: Reliable model in the computer residue number system. Electron. Model. **7**(4), 44–46 (1985)