Entropy **2015**, *17*, 3692-3709; doi:10.3390/e17063692

OPEN ACCESS
entropy
ISSN 1099-4300
www.mdpi.com/journal/entropy

Article

The Switching Generator: New Clock-Controlled Generator with Resistance against the Algebraic and Side Channel Attacks

Jun Choi [1], Dukjae Moon [2], Seokhie Hong [2] and Jaechul Sung [3,*]

[1] The 2nd branch, Defense Security Institute, Seosomun-ro Jung-gu, Seoul 100-120, Korea; E-Mail: choijun1014@hotmail.com

[2] Center for Information Security Technologies (CIST), Korea University, Seoul 136-701, Korea; E-Mails: djmoon17@hotmail.com (D.M.); shhong@korea.ac.kr (S.H.)

[3] Department of Mathematics, University of Seoul, Seoul 130-743, Korea

* Author to whom correspondence should be addressed; E-Mail: jcsung@uos.ac.kr; Tel.: +82-2-6490-2614.

Academic Editors: James Park and Wanlei Zhou

Received: 18 March 2015 / Accepted: 1 June 2015 / Published: 4 June 2015

Abstract: Since Advanced Encryption Standard (AES) in stream modes, such as counter (CTR), output feedback (OFB) and cipher feedback (CFB), can meet most industrial requirements, the range of applications for dedicated stream ciphers is decreasing. There are many attack results using algebraic properties and side channel information against stream ciphers for hardware applications. Al-Hinai *et al.* presented an algebraic attack approach to a family of irregularly clock-controlled linear feedback shift register systems: the stop and go generator, self-decimated generator and alternating step generator. Other clock-controlled systems, such as shrinking and cascade generators, are indeed vulnerable against side channel attacks. To overcome these threats, new clock-controlled systems were presented, e.g., the generalized alternating step generator, cascade jump-controlled generator and mutual clock-controlled generator. However, the algebraic attack could be applied directly on these new systems. In this paper, we propose a new clock-controlled generator: the switching generator, which has resistance to algebraic and side channel attacks. This generator also preserves both security properties and the efficiency of existing clock-controlled generators.

Keywords: clock-controlled generator; algebraic attack; side channel attack; switching generator; period and linear complexity

1. Introduction

Since the National Institute of Standards and Technology (NIST) announced that Rijndael was proclaimed the winner of the Advanced Encryption Standard (AES) competition, no critical security weaknesses have been demonstrated for this algorithm until now [1]. In addition, it has been reported that its performance implemented in software and hardware seems to be excellent across a wide range of computing environments [2,3]. Especially, AES in counter (CTR) or output feedback (OFB) mode can meet most requirements for stream cipher systems, so that there has been an unavoidable long-term decline for stream ciphers.

At the State of the Art in Stream Ciphers (SASC) 2004 and Asiacrypt 2004, A. shamir presented an invited lecture titled "Stream Ciphers: Dead or Alive?" [4]. In this lecture, he pointed out that stream ciphers would survive only in some niche applications. These applications are as follows:

- software-oriented applications with exceptionally high speed (e.g., routers)
- hardware-oriented applications with an exceptionally small footprint (e.g., RFID, smart cards)

Stream ciphers for hardware-oriented applications with restricted resources are often designed using a linear feedback shift register (LFSR) [5]. LFSRs are well suited for hardware implementation. They can produce sequences with large periods and good statistical properties. To generate keystreams with good statistical properties, designers usually used LFSR output sequences as follows [6]:

- use multiple LFSRs and a nonlinear combining function
- use a single LFSR and nonlinear filter function
- use irregular clocking of the LFSRs (clock-controlled generators)

However, these generators with LFSRs are potentially very vulnerable to algebraic attacks and side channel attacks.

An algebraic attack that uses over-defined systems of multivariate equations to recover the secret key has gained much attention. These equations are derived relating the output bits to the internal state bits for the generators. Let the size of the internal state be l and the low degree of the equations be d; then, the time complexity of the algebraic attack is $O(l^d)^\omega$, where ω is the exponent of Gaussian reduction ($\omega < 3$). In the paper [7], the value of ω is $\log_2 7$. Various algebraic attacks were first applied to block ciphers and public key cryptosystems [8–10]. Thereafter, these attacks have been effectively applied to LFSR-based systems [11–14]. In particular, Al-Hinai *et al.* showed that algebraic attacks were effective against the irregularly-clocked LFSR systems in 2006 [15]. To be secure against the algebraic attack, Kanso presented the generalized clock-controlled alternating step generator in 2009 [16]. However, Hassanzadeh and Helleseth described an algebraic attack against this generator [17].

Irregularly-clock-controlled shift registers are highly vulnerable against side channel attacks, such as simple power analysis and timing attacks [18] (Chapter 7). In particular, this attack could be strongly applied to the shrinking generator and cascade generator, which had resistance to an algebraic attack. To overcome these attacks, new clock-controlled generators were proposed in the European ECRYPT eSTREAM project, such as cascade the jump-controlled generator (named Pomaranch) and the mutual clock-controlled generator (named Mickey) [19]. However, the algebraic attack could be applied on these generators [20].

In this paper, we present a new clock-controlled generator: the switching generator. This generator can be resistant to algebraic attack by maintaining both secure properties and the efficiency of traditional clock-controlled generators. Especially, we can protect this generator against side channel attacks by choosing some parameters easily.

2. Clock-Controlled Generators

LFSRs are known to allow fast implementation on hardware and produce sequences with large periods and good statistical properties. However, the inherent linearity of these sequences results in susceptibility to algebraic attacks. That is the fundamental reason why LFSRs are not used directly for keystream generation. A well-known method for increasing the linear complexity with preserving the same properties as those sequences is a nonlinear transformation, such as a filter generator and a combination generator. An alternative method to achieve the same goal is to control LFSR clocks, that is a clock-controlled generator.

The components of the clock-controlled generator can be grouped into two subsystems based on functions: a clock generation and a data generation. In general, each component consists of maximum-length LFSRs (m-LFSRs) that produce an output sequence with a maximum possible period. The generator works in the following way:

1. A clock-controlling LFSR is clocked regularly.
2. Output bits of the controlling LFSR are used to determine the clocking.
3. By determining the clocking, a data-generating LFSR is clocked.

In order to describe this process in detail, we label registers A and B with length m and n. The i-th bit of register A at time t is denoted by A_i^t. Suppose that the i-th bit of A controls the clocking of B in such a way that if it is "0", B is not clocked, and if it is "1", B is clocked j times. In this case, we can express the change to the k-th position of B as Equation (1):

$$B_k^t \;=\; B_k^{t-1} \cdot (A_i^{t-1} \oplus 1) \;\oplus\; B_{k-j}^{t-1} \cdot A_i^{t-1} \tag{1}$$

Let f_A and f_B be a characteristic polynomial and M_A and M_B be a companion matrix for each register, A and B; the above equation can be represented as Equation (2):

$$B_k^t \;=\; [M_B \cdot B^{t-2}]_k \cdot ([M_A \cdot A^{t-2}]_i \oplus 1) \oplus\; [M_B \cdot B^{t-2}]_{k-j} \cdot [M_A \cdot A^{t-2}]_i \tag{2}$$

In this section, we describe existing clock-controlled generators and summarize an algebraic attack approach to these generators.

2.1. Description of Clock-Controlled Generators

Clock-controlled generators are described in this subsection. These generators produce sequences of large periods, large linear complexity and good statistical properties.

2.1.1. Stop and Go Generator

The stop and go generator, proposed by Beth and Piper in 1984, has two LFSRs A and B of length m and n, respectively [21]. The output sequences z^t are generated as follows:

1. Register A is clocked.
2. If the output of A is "0", then register B is not clocked.
3. If the output of A is "1", then register B is clocked.
4. The most significant bit of B forms part of the output sequence.

We can represent the above process as Equation (3).

$$z^t = B_n^{t-1} \cdot (A_m^{t-1} \oplus 1) \oplus B_{n-1}^{t-1} \cdot A_m^{t-1} \tag{3}$$

In [22], Gollmann and Chambers proposed the Step 1/Step 2 generator, which may be seen as a generalization of the stop and go generator. Register A of this generator controls the clocking of register B on the principle that if the output is "0", then B clocks d times before producing the output sequence, and if the output is "1", B clocks k times before doing so. This process is represented as Equation (4).

$$z^t = B_{n-d}^{t-1} \cdot (A_m^{t-1} \oplus 1) \oplus B_{n-k}^{t-1} \cdot A_m^{t-1} \tag{4}$$

2.1.2. Self-Decimated Generator

The self-decimated generator consists of a single LFSR A of length m [23]. LFSR A controls the shifting of itself in the following way:

1. Register A is clocked.
2. If the output of A is "0", then register A is clocked d times.
3. If the output of A is "1", then register A is clocked k times.
4. The most significant bit of A forms part of the output sequence.

This process is called the (d, k) self-decimated generator and represented as Equation (5).

$$z^t = A_{m-d}^{t-1} \cdot (A_m^{t-1} \oplus 1) \oplus A_{m-k}^{t-1} \cdot A_m^{t-1} \tag{5}$$

2.1.3. Alternating Step Generator

The alternating step generator employs three LFSRs A, B and C of length m, n and l, respectively [24]. The output bits are generated as follows:

1. Register A is clocked.
2. If the output of A is "0", then register B is clocked.
3. If the output of A is "1", then register C is clocked.
4. The output bit is the XOR of the most significant bit of registers B and C.

We can represent the process as Equation (6).

$$\begin{aligned} B_n^t &= B_n^{t-1} \cdot (A_m^{t-1} \oplus 1) \oplus B_{n-1}^{t-1} \cdot A_m^{t-1} \\ C_l^t &= C_{l-1}^{t-1} \cdot (A_m^{t-1} \oplus 1) \oplus C_l^{t-1} \cdot A_m^{t-1} \\ z^t &= B_n^t \oplus C_l^t \end{aligned} \tag{6}$$

In 2009, Kanso presented the generalized alternating step generator [16]. This generator is similar to the original alternating step generator, except that it is clocked in a variable number of steps. He used the values of W_B and W_C and the sets $\{i_0, i_1, \cdots, i_{W_B-1}\}$ and $\{i_0, i_1, \cdots, i_{W_C-1}\}$ to compute a various clocking number. In this generator, these values and sets are considered as a part of the key to be secure against the algebraic attack. The output bits are generated as follows:

1. Register A is clocked.
2. If the output of A is "0", then register B is clocked $r(t)$ times, where $r(t) = 1 + 2^0 \cdot A_{i_0}^t + 2^1 \cdot A_{i_1}^t + \cdots + 2^{W_B-1} \cdot A_{i_{W_B-1}}^t, W_B < m$.
3. If the output of A is "1", then register C is clocked $s(t)$ times, where $s(t) = 1 + 2^0 \cdot A_{j_0}^t + 2^1 \cdot A_{j_1}^t + \cdots + 2^{W_C-1} \cdot A_{j_{W_C-1}}^t, W_C < l$.
4. The output bit is the XOR of the most significant bit of registers B and C.

2.1.4. Cascade Generator

The cascade generator consists of N LFSRs and works in the same way as the stop and go generator [25]. To explain the process of the cascade generator, we describe the cascade generator of three stages. This generator has three LFSRs A, B and C of length m, n and l, respectively, and works in the following way:

1. Register A is clocked.
2. If the output of A is "0", then register B is not clocked; else B is clocked.
3. If the output of the XOR of the most significant bit of registers A and B is "0", then register C is not clocked; else C is clocked.
4. The output sequence is the XOR of the most significant bit of A, B and C.

The above process is represented as Equation (7) and can be expanded into the cascade generator of N stages easily.

$$\begin{aligned} B_n^t &= B_n^{t-1} \cdot (A_m^{t-1} \oplus 1) \oplus B_{n-1}^{t-1} \cdot A_m^{t-1} \\ C_l^t &= C_l^{t-1} \cdot (A_m^{t-1} \oplus B_n^{t-1} \oplus 1) \oplus C_l^{t-1} \cdot (A_m^{t-1} \oplus B_n^{t-1}) \\ z^t &= A_m^t \oplus B_n^t \oplus C_l^t \end{aligned} \tag{7}$$

2.1.5. (Self-)Shrinking Generator

The shrinking generator [26], proposed by Coppersmith *et al.* in 1993, employs two LFSRs A and B of length m and n, respectively. The output sequences are generated as follows:

1. Registers A and B are clocked.
2. If the output of A is "0", then the output of B is discarded.
3. If the output of A is "1", then the output of B forms part of the output sequence.

By modifying this generator, Meier and Staffelbach proposed the self-shrinking generator in 1994 [27]. This generator consist of a single LFSR A and works in the following way:

1. Register A is clocked two times.
2. If the first output of A is "0", then the second output is discarded.
3. If the first output of A is "1", then the second output forms part of the output sequence.

In 2010, Kanso presented the modified self-shrinking generator [28]. The output sequences are generated as follows:

1. Generate the bit-triple $(A_m^t, A_m^{t+1}, A_m^{t+2})$ through three times clocking of register A.
2. If $A_m^t \oplus A_m^{t+1}$ = "0", then the third output is discarded.
3. If $A_m^t \oplus A_m^{t+1}$ = "1", then the third output forms part of the output sequence.

The output bit of these generators cannot be represented as a determinate equation.

2.2. Algebraic Attacks on Clock-Controlled Generators

In paper [15,20], the authors described an algebraic attack against clock-controlled generators with the following properties:

- For each register, the transitions from one state to the next must be described by a matrix product of some copies of the given companion matrix.
- For each bit "0" and "1" of the controlling LFSR, the number of copies is fixed. For example, if the control bit is "0", then the number of copy is zero, otherwise the number of copies is one in the stop and go generator.

Using these properties, they produced a family of equations relating the output bits to the internal state bits for generators. The degree of this family of equations could be bounded by two. Therefore, they could obtain the initial state of the generators within a significantly faster time than any other known attack. Since the stop and go generator, self-decimated generator and alternating step generator had the above properties, the algebraic attacks were applied to these generators easily.

As an example, they produced equations for two consecutive output bits of the stop and go generator as follows:

$$z^t = B_n^{t-1} \cdot (A_m^{t-1} \oplus 1) \oplus B_{n-1}^{t-1} \cdot A_m^{t-1} = B_n^t \quad (8)$$

$$z^{t+1} = B_n^t \cdot (A_m^t \oplus 1) \oplus B_{n-1}^t \cdot A_m^t \quad (9)$$

Adding Equations (8) and (9), they could obtain Equation (10) of degree two.

$$\begin{aligned} z^t \oplus z^{t+1} &= B_n^{t-1} \cdot (A_m^{t-1} \oplus 1) \oplus B_{n-1}^{t-1} \cdot A_m^{t-1} \oplus B_n^t \cdot (A_m^t \oplus 1) \oplus B_{n-1}^t \cdot A_m^t \\ &= B_n^t \oplus B_n^t \cdot (A_m^t \oplus 1) \oplus B_{n-1}^t \cdot A_m^t \\ &= (B_n^t \oplus B_{n-1}^t) \cdot A_m^t \end{aligned} \tag{10}$$

Therefore, the time complexity of this generator for the algebraic attack was $O(n^{2\cdot\omega})$.

By the same method, equations bounded by degree two of the self-decimated generator and alternating step generator could be produced easily. In 2010, the generalized alternating step generator, designed for having resistance against the algebraic attack, was analyzed by Hassanzadeh and Helleseth [17]. They used the algebraic attack technique. However, these attacks could not be applied to the (self-)shrinking generator and cascade generator, which did not have the above properties.

3. Switching Generator

In this section, we present a new clock-controlled generator: the switching generator with resistance to an algebraic attack and side channel attack.

3.1. Switching Generator Specification

The switching generator, shown in Figure 1, has two LFSRs A and B of length m and n, respectively. In Figure 1, a MUX (multiplexer) is a logic that selects one of several inputs [29]. The data-generating LFSR B works as Equation (11) using two companion matrices M_1 and M_2. These matrices are companion matrices of two primitive polynomials.

$$B^t = \begin{cases} (M_1)^i \cdot B^{t-1} & \text{if } A_m^{t-1} = 0, \\ (M_2)^j \cdot B^{t-1} & \text{if } A_m^{t-1} = 1. \end{cases} \tag{11}$$

The data-generating LFSR B is updated by bits of the controlling LFSR A and two companion matrices M_1 and M_2. We describe the generating method for output sequences as follows:

1. Register A is clocked.
2. If the output of A is "0", then the register B is clocked by using $(M_1)^i$, then the output bit is the most significant bit of this register.
3. If the output of A is "1", then the register B is clocked by using $(M_2)^j$, then the output bit is the most significant bit of this register.

The period of the output sequences generated by this generator can be up to a maximum $(2^m - 1) \cdot (2^n - 1)$ if the matrix M_2 is applied repetitively in the clocking process.

We define the ordered pair (i, j) of repetition as the switching index. Through using two distinct companion matrices and a carefully chosen switching index in this generator, we can guarantee that this generator can generate the maximum length sequence. For having resistance to the algebraic attack, this

switching index is considered as a part of the key. If we choose this index carefully, this generator can be also resistant to the side channel attack.

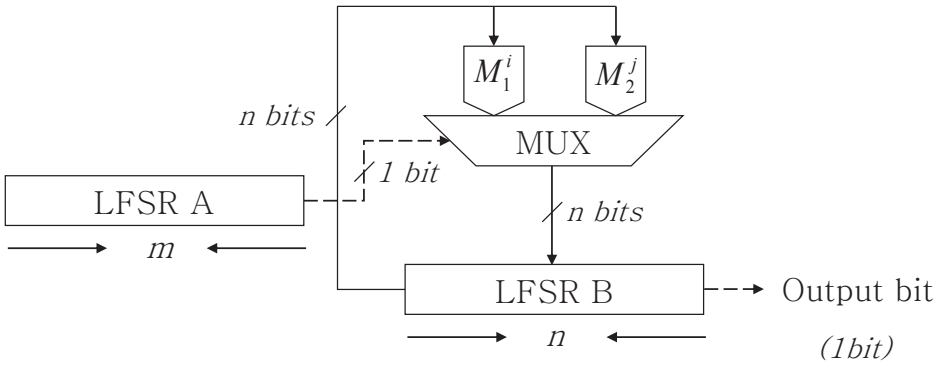

Figure 1. The structure of the switching generator.

3.2. Switching Index

The switching index (SI) (i, j) is determined by bits of the controlling register A, where $1 \leq i \leq ord(M_1) - 1$ and $1 \leq j \leq ord(M_2) - 1$. We define the smallest multiplication order of a given matrices M_1 and M_2 as $ord(M_1)$ and $ord(M_2)$, when $M_1^{ord(M_1)} = M_2^{ord(M_2)} = I$, where I is the identity matrix. This index can be found through following steps:

1. Generate output sequences of clock-controlling LFSR A as long as the period of this LFSR.
2. For each index (i, j),
   (a) multiply two companion matrices depending on the output sequences of LFSR A;
   (b) compute the characteristic polynomial of the computed matrix in Step (a).
   (c) If the characteristic polynomial is a primitive polynomial, then the pair (i, j) is a switching index.

In these steps, we call the switching matrix (SM) the multiplications of two companion matrices in Step 2 (a) and the detailed process is shown in Example 1.

Example 1. *Let f_c be a characteristic polynomial of degree two for the clock-controlling LFSR and f_{g_1}, f_{g_2} be two distinct polynomials of the same degree three for the data generating LFSR.*

$$f_c = x^2 + x + 1,\ f_{g_1} = x^3 + x + 1,\ f_{g_2} = x^3 + x^2 + 1$$

For each polynomial, companion matrices are as follows:

$$M_c = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix},\ M_1 = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix},\ M_2 = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

Suppose that an initial state of the clock-controlling LFSR is (0,1); the output sequences are the three-periodic sequences with cycle (0,1,1). Using these sequences, the switching matrix SM is computed as follows:

$$SM = (M_2)^j \cdot (M_2)^j \cdot (M_1)^i = (M_2)^{2j} \cdot (M_1)^i$$

In the case of $(i, j) = (1, 6)$*, SM is represented as the below matrix. The characteristic polynomial of this matrix is* $x^3 + x^2 + 1$ *and is primitive. Therefore,* $(1, 6)$ *is the switching index. We can also find other indexes, such as* $(2, 3)$*,* $(3, 2)$*,* $(4, 5)$*,* $(5, 4)$ *and* $(6, 1)$*.*

$$SM_{(1,6)} = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix}^6 \cdot \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix}^6 \cdot \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \end{pmatrix}$$

To obtain the switching index for given sizes m and n, we have to compute many operations as follows:

1. Generate the output sequences of clock-controlling A as long as the period of this LFSR.
   - The time complexity is $2^m - 1$.
2. For each index (i, j),
   - the time complexity is $(2^n - 1) \times (2^n - 1)$.
   (a) Multiply two companion matrices depending on the output sequences of A.
      - The time complexity is $(2^m - 1) \times O(n^3)$, where $O(n^3)$ is the complexity of the multiplication between two $n \times n$ matrices and $O(\bullet)$ is the asymptotic upper bound.
   (b,c) Compute the characteristic polynomial of the computed matrix and check that this characteristic polynomial is a primitive.
      - The time complexity is $O(n^3)$, where $O(n^3)$ is the complexity for factoring polynomials over the given fields [6].
3. The total time complexity is $(2^m-1)+(2^n-1)(2^n-1)\{(2^m-1)O(n^3)+O(n^3)\}$ by 1 and 2.

If the size m and n is large, then the time complexity is very high. Therefore, we expressed this situation as a hard problem.

3.3. Application of Switching Generator

To provide strong robustness, stream ciphers generally use many LFSRs or generators based on LFSR. There are two methods for designing stream ciphers: a combination generator and a filter generator [6]. A5 (GSM) [30] and E0 (Bluetooth) [31] were constructed with a combination generator, and Sober (Nessie project) [32] and Grain eStream project) [33] were designed with a filter generator. It is

not good for security that our generator uses many LFSRs with the controlling LFSR because of the guess-and-determine attacks. We can suggest that a stream cipher use many of our generators to meet a design goal, such as 128-bit security and resistance against side channel attacks and algebraic attacks.

4. The Switching Generator's Properties

In this section, we present the switching generator's properties. The generator has resistance to algebraic attacks with maintaining a large period, linear complexity and good randomness.

4.1. Existence of the Switching Index

For given initial states of length m and n, the existence proof of the switching index is a hard problem. Therefore, we consider the probabilistic existence of the switching index, where this probability is the chance that the characteristic polynomial of the switching matrix computed by clock control bits and index (i, j) is primitive. To compute this probability, we use the following facts:

- Since the primitive polynomial must have a non-zero constant term, the number of candidates for the primitive polynomial is 2^{n-1}
- Over GF(2^n), there are exactly $\phi(2^n - 1)/n$ primitive polynomials of degree n, where ϕ is Euler's totient function [34].

Let $\mathrm{Pr}_{f,n}$, shown in Equation (12), be the probability that a polynomial f of degree n in GF(2^n) is primitive. These values for given degrees n are listed in Table 1.

$$\mathrm{Pr}_{f,n} = \frac{\frac{\phi(2^n-1)}{n}}{2^{n-1}} = \frac{\phi(2^n-1)}{n \cdot 2^{n-1}} \tag{12}$$

Table 1. The probability $Pr_{f,n}$.

State Size(n)	Probability	State Size(n)	Probability
2	0.5	32	0.03125
4	0.25	64	0.0156
8	0.125	128	0.0078
16	0.0625	256	0.0039

From Equation (12) and the bounds of index (i, j), we can compute the expectation $\mathrm{Exp}_{f,i,n}$ of the existence of the switching index for degree n and a fixed i as Equation (13). These expectation values for given degrees n are listed in Table 2.

$$\mathrm{Exp}_{f,i,n} = (2^n - 2) \cdot \frac{\phi(2^n-1)}{n \cdot 2^{n-1}} = \frac{\phi(2^n-1) \cdot (2^{n-1}-1)}{n \cdot 2^{n-2}} \tag{13}$$

By Equation (13), we can compute the expectation for degree $n = 3$ (see Example 1). The expectation is 3.5. Therefore, there are at least three candidates from a theoretical standpoint. However, there are six candidates, such as $(1, 6)$, $(6, 1)$, $(2, 3)$, $(3, 2)$, $(4, 5)$ and $(5, 4)$, in Example 1.

Table 2. The expectation $Exp_{f,i,n}$.

State Size(n)	Expectation	State Size(n)	Expectation
2	1	32	2^{27}
4	3.5	64	$2^{57.99}$
8	$2^{4.99}$	128	$2^{120.99}$
16	$2^{11.99}$	256	$2^{247.99}$

4.2. Properties of the Switching Index

Since the matrix multiplication is not commutative, switching matrices would be changed depending on the output bits of the clock-controlling LFSR. This property means that the period of the output sequence generated by the switching generator would be changed depending on the initial state for the clock controlling LFSR. However, the below Theorem 1 indicates that the period of the switching generator is constant, independent of the initial state for the clock controlling LFSR [35].

Theorem 1. *For two square matrices M_1, M_2, the characteristic polynomial CP has a property as follows:*

$$CP_{M_1 \cdot M_2}(x) = CP_{M_2 \cdot M_1}(x)$$

Proof. If λ is a root of $CP_{M_1 \cdot M_2}(x)$, then it is an eigenvalue of $M_1 \cdot M_2$. Therefore, there is a non-zero vector v, such that $M_1 \cdot M_2 \cdot v = \lambda \cdot v$. We can multiply $M_1 \cdot M_2 \cdot v = \lambda \cdot v$ by M_2 on both sides, then obtain the below equation:

$$\underline{M_2 \cdot M_1 \cdot (M_2 \cdot v)} = M_2 \cdot (M_1 \cdot M_2 \cdot v) = M_2 \cdot (\lambda \cdot v) = \underline{\lambda \cdot (M_2 \cdot v)}$$

Since $M_2 \cdot v$ is non-zero, it is an eigenvector of $M_2 \cdot M_1$ with the same eigenvalue that $M_1 \cdot M_2$ had. In this case, λ would also be a root of $M_2 \cdot M_1$'s characteristic polynomial. Conversely, if λ is a root of $CP_{M_2 \cdot M_1}(x)$, then it also is a root of $M_1 \cdot M_2$'s characteristic polynomial by the same method. Therefore, two characteristic polynomials are the same. □

This theorem means that the characteristic polynomials of a circular multiplication for given matrices are the same. Hence, if the clock-controlling LFSR and switching index is fixed, then the switching generator has the maximum period independent of the initial vector for the clock-controlling LFSR.

4.3. Choice of the Switching Index

There are many candidates for the switching index. If we can choose this index considering the same number of XOR operations on two matrices $(M_i)^i$ and $(M_2)^j$, the switching generator has resistance to the side channel attack. By using Example 1, we describe this process of choosing a good switching index. We already found indexes, such as $(1, 6)$, $(6, 1)$, $(2, 3)$, $(3, 2)$, $(4, 5)$ and $(5, 4)$. For each index, we compare the number of XOR operations. The comparison results are listed in Table 3.

Table 3. The comparison results of the number of XOR operation.

SI (i,j)	$(M_1)^i$	No. **of XORs: (a)**	$(M_2)^j$	No. **of XORs: (b)**	**difference between (a) and (b)**
$(1,6)$	$\begin{pmatrix}0&0&1\\1&0&1\\0&1&0\end{pmatrix}$	1	$\begin{pmatrix}0&1&0\\1&0&1\\1&0&0\end{pmatrix}$	1	0
$(6,1)$	$\begin{pmatrix}1&1&0\\0&0&1\\1&0&0\end{pmatrix}$	1	$\begin{pmatrix}0&0&1\\1&0&0\\0&1&1\end{pmatrix}$	1	0
$(2,3)$	$\begin{pmatrix}0&1&0\\0&1&1\\1&0&1\end{pmatrix}$	2	$\begin{pmatrix}1&1&1\\0&1&1\\1&1&0\end{pmatrix}$	4	2
$(3,2)$	$\begin{pmatrix}1&0&1\\1&1&1\\0&1&1\end{pmatrix}$	4	$\begin{pmatrix}0&1&1\\0&0&1\\1&1&1\end{pmatrix}$	3	1
$(4,5)$	$\begin{pmatrix}0&1&1\\1&1&0\\1&1&1\end{pmatrix}$	4	$\begin{pmatrix}1&0&1\\1&1&0\\0&1&0\end{pmatrix}$	2	2
$(5,4)$	$\begin{pmatrix}1&1&1\\1&0&0\\1&1&0\end{pmatrix}$	3	$\begin{pmatrix}1&1&0\\1&1&1\\1&0&1\end{pmatrix}$	4	1

Using the comparison results in this table, we can choose the good switching index as $(1,6)$ or $(6,1)$ from six candidates.

4.4. Period and Linear Complexity [36]

By the definition of switching index, the period of the switching generator is $(2^m-1)\cdot(2^n-1)$. The generator's linear complexity LC is shown in Theorem 2.

Theorem 2. *For the given switching generator consisting of two LFSRs of length m and n, if m divides n, then this generator has the linear complexity as follows:*

$$LC = n\cdot(2^m-1)$$

Proof. In paper [37], Chambers *et al*. have considered a type of BRM (binary rate multiplier) whose control and generating registers, A and B, are primitive LFSRs of periods 2^m-1 and 2^n-1. They have shown that its output sequence has linear complexity $n\cdot(2^m-1)$ under two conditions as follows:

1. every prime factor of 2^m-1 divides 2^n-1.
2. GCD(α_{BRM}, 2^n-1) = 1, where α_{BRM} is the number of clocks applied to Register B after clocking A 2^m-1 times.

This result is able to be applied into a switching generator directly. If m divides n, then every prime factor of 2^m-1 divides 2^n-1 [38]. Since the characteristic polynomial of the switching matrix (SM) is a primitive polynomial for a given switching index (i,j), the integers $\alpha_{S.G}$ and 2^n-1 are relatively prime, where $\alpha_{S.G}$ is the number of clocks applied to Register B after clocking A 2^m-1 times for the switching generator. Therefore, the linear complexity of this generator is $n\cdot(2^m-1)$. □

Additionally, the minimal polynomial of the output sequence of the switching generator is exactly $f_{SM}(x^{(2^m-1)})$. In Example 1, since $f_{SM_{(1,6)}} = x^3 + x^2 + 1$, the minimal polynomial of the output sequence is $f_{SM_{(1,6)}}(x^3) = (x^3)^3 + (x^3)^2 + 1 = x^9 + x^6 + 1$

4.5. Algebraic Properties

At clock t, the internal state B^t of the data-generating LFSR is updated according to the bit information of the clock-controlling LFSR and two companion matrices M_1 and M_2. This process is represented as Example 2 in detail.

Example 2. *We use characteristic polynomials for clock-controlling and data-generating LFSRs, defined in Example 1. Then, the switching index is "(1,6)", and the switching matrix SM is as follows:*

$$SM = (M_2)^6 \cdot (M_2)^6 \cdot M_1 = (M_2)^{12} \cdot M_1$$

Let the initial state of the data generating LFSR be (0,0,1). the internal states B^t are as follows:

$$\begin{aligned}
(0,0,1) &\xrightarrow{M_1} (1,1,0) \xrightarrow{(M_2)^6} (1,1,1) \xrightarrow{(M_2)^6} (1,0,1)\\
&\xrightarrow{M_1} (1,0,0) \xrightarrow{(M_2)^6} (0,1,1) \xrightarrow{(M_2)^6} (1,1,0)\\
&\xrightarrow{M_1} (0,1,1) \xrightarrow{(M_2)^6} (1,1,0) \xrightarrow{(M_2)^6} (1,1,1)\\
&\xrightarrow{M_1} (1,0,1) \xrightarrow{(M_2)^6} (0,0,1) \xrightarrow{(M_2)^6} (0,1,0)\\
&\xrightarrow{M_1} (0,0,1) \xrightarrow{(M_2)^6} (0,1,0) \xrightarrow{(M_2)^6} (1,0,0)\\
&\xrightarrow{M_1} (0,1,0) \xrightarrow{(M_2)^6} (1,0,0) \xrightarrow{(M_2)^6} (0,1,1)\\
&\xrightarrow{M_1} (1,1,1) \xrightarrow{(M_2)^6} (1,0,1) \xrightarrow{(M_2)^6} (0,0,1)
\end{aligned}$$

We can represent the update process for these internal states as the matrix product of some copies of the matrix M_1 or M_2. The number of copies is defined as exponents. The exponents for each matrix M_1 or M_2 are listed in Table 4. In this table, we can notice that the exponents are not constant for each bit "0" or "1" of the clock controlling LFSR. This is due to the following mathematical property: for two companion matrix M_1 and M_2 of two distinct primitive polynomials, the matrix M_1 cannot be represented as the matrix product of some copy of the matrix M_2. This explains that the switching generator cannot be implemented as the stop and go generator. Thus, the algebraic attack applied to the stop and go generator cannot be applied to the switching generator.

In Section 3.2, we presented that there are many candidates for the switching index. Additionally, we described that this switching index is considered as a part of the key. This condition is used to guarantee that the switching generator has resistance against the algebraic attack. Even though attackers assume the clock-controlling LFSR, they cannot find the exact switching index among many candidates. Therefore, they cannot construct algebraic equations for the target generator easily.

Table 4. The exponents for each matrix (the period is 21).

M_1 **Case**	state	(0,0,1)	(1,1,0)	(1,1,1)	(1,0,1)	(1,0,0)	(0,1,1)
	exponent	-	1	2	1	1	4
	state	(1,1,0)	(0,1,1)	(1,1,0)	(1,1,1)	(1,0,1)	(0,0,1)
	exponent	5	1	6	2	1	2
	state	(0,1,0)	(0,0,1)	(0,1,0)	(1,0,0)	(0,1,0)	(1,0,0)
	exponent	6	1	6	6	1	6
	state	(0,1,1)	(1,1,1)	(1,0,1)	(0,0,1)	-	-
	exponent	3	1	1	3	-	-
M_2 **Case**	state	(0,0,1)	(1,1,0)	(1,1,1)	(1,0,1)	(1,0,0)	(0,1,1)
	exponent	-	3	6	6	4	6
	state	(1,1,0)	(0,1,1)	(1,1,0)	(1,1,1)	(1,0,1)	(0,0,1)
	exponent	6	1	6	6	6	6
	state	(0,1,0)	(0,0,1)	(0,1,0)	(1,0,0)	(0,1,0)	(1,0,0)
	exponent	6	1	6	6	1	6
	state	(0,1,1)	(1,1,1)	(1,0,1)	(0,0,1)	-	-
	exponent	6	5	6	6	-	-

Attackers have to be able to construct algebraic equations for the target algorithm to attack this algorithm with the basic algebraic attack. If they can attack, then they can apply other advanced algebraic attacks, such as fast algebraic attacks [13] and higher order algebraic attacks [39]. However, they cannot construct any algebraic equations in our generator.

4.6. Other Security Evaluation

In clock-controlled generators, the basic idea of guess-and-determine attacks is to guess m bits of the clock-controlling LFSR state and derive other n bits through the relationship between the keystream bits and the internal state bits. The time complexity is $(2^m - 1) \times O(n^3)$, where $O(n^3)$ is the complexity of the Gaussian elimination method for $n \times n$ matrix [40].

When the keystream bits in the given generator have a significant correlation with the output bits of some LFSRs, we can apply the correlation attacks. Our generator is derived from the alternating step generator. The best time complexity of a correlation attack against the alternating step generator is $O(n^2 \times 2^n)$ [41]. The switching generator is more robust than the alternating step generator, because attackers cannot know the exact switching index, even though they can know all primitive polynomial for all LFSRs. Therefore, we can argue the complexity for being dominated by the time complexity of the alternating step generator. If we can choose the two parameters m and n, such that $(2^m - 1) \times n \leq n^2 \times 2^n$, namely $n \leq m \leq n \times (log_2 n)$, then the complexity against these attacks is more than the linear complexity. Therefore, the time complexity against all basic attacks is dominated by the time complexity of the linear complexity.

5. Conclusions

The appearance of fast and efficient block ciphers has caused a diminishing of the importance of stream ciphers due to the convenience of the use of block ciphers in various applications. Therefore, researchers of stream ciphers reviewed some area where stream ciphers would survive: very high speeds in communication links or compact in constrained devices. In the area of constrained devices, LFSRs are used to construct stream ciphers. It has, however, been well known that LFSR-based systems, well suited to low-end hardware implementations, are very vulnerable to algebraic attacks. Furthermore, irregular clock-controlled generators were highly vulnerable against side channel attacks.

In this paper, we proposed a new clock-controlled generator: the switching generator. This generator preserved both the security and efficiency of existing clock-controlled generators. In addition, this system could be resistant to algebraic attacks, because determinate equations for this generator, relating the output bits to the internal state bits for this generator, could not be produced. Especially, we could guarantee the resistance against side channel attacks by choosing a good switching index.

Acknowledgments

This research was supported by the MSIP (Ministry of Science, ICT and Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (NIPA-2014-H0301-14-1004) supervised by the NIPA (National IT Industry Promotion Agency) for Dukjae Moon and Seokhie Hong. Furthermore, this work was supported by the 2014 Research Fund of the University of Seoul for Jaechul Sung.

Author Contributions

All authors have contributed to the study and preparation of the article. The 1st author conceived the idea and wrote the paper. The 2nd author did the security analysis and proved theorems. The 3rd author reviewed and advised for the paper. The corresponding author participated in the security analysis and reviewed for the paper. All authors have read and approved the final manuscript.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Daemen, J.; Rijmen, V. *The Design of Rijndael*; Springer: Berlin/Heidelberg, Germany, 2001.
2. Good, T.; Benaissa, M. AES on FPGA : From the Fastest to the Smallest. In *Cryptographic Hardware and Embedded Systems—CHES 2005*, Proceedings of 7th International Workshop on Cryptographic Hardware and Embedded Systems, Edinburgh, UK, 29 August–1 September 2005; Rao, J.R., Sunar, B., Eds.; Volume 3659, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2005; pp. 427–440.

3. Könighofer, R. A Fast and Cache-Timing Resistant Implementation of the AES. In *Topics in Cryptology—CT-RSA 2008*, Proceedings of The Cryptograhers' Track at the RSA Conference 2008, San Francisco, CA, USA, 8–11 April 2008; Malkin, T., Ed.; Volume 4964, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2008; pp. 187–202.
4. Shamir, A. Stream Ciphers: Dead or Alive? In Proceedings of ASIACRYPT 2004, Jeju Island, Korea, 5–9 December 2004; Lee, P.J., Ed.; Volume 3329, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2004; p. 78.
5. Ronse, C. *Feedback Shift Registers*; Lecture Notes in Computer Science, Volume 169; Springer: Berlin/Heidelberg, Germany, 1984; .
6. Menezes, A.J.; van Oorschot, P.C.; Vanstone, S.A. *Handbook of Applied Cryptography*; CRC Press: Boca Raton, FL, USA, 1996.
7. David H.; Bailey, K.L.; Simon, H.D. Using Strassen's Algorithm to Accelerate the Solution of Linear Systems. *J. Supercomput.* **1990**, *4*, pp. 357–371.
8. Courtois, N.T. The Security of Hidden Field Equations (HFE). In *Topics in Cryptology—CT-RSA 2001*, Proceedings of The Cryptographers' Track at RSA Conference, San Francisco, CA, USA, 8–12 April 2001; Naccache, D., Ed.; Volume 2020, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2001; pp. 266–281.
9. Courtois, N.; Pieprzyk, J. Cryptanalysis of Block Ciphers with overdefined Systems of Equations. In *Advances in Cryptology—ASIACRYPT 2002*, Proceedings of 8th International Conference on the Theory and Application of Cryptology and Information Security, Queenstown, New Zealand, 1–5 December 2002; Zheng, Y., Ed.; Volume 2501, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2002; pp. 267–287.
10. Kipnis, A.; Shamir, A. Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization. In *Advances in Cryptology—CRYPTO 1999*, Proceedings of 19th Annual International Cryptology Conference Santa Barbara, CA, USA, 15–19 August 1999; Wiener, M.J., Ed.; Volume 1666, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1999; pp. 19–30.
11. Armknecht, F.; Krause, M. Algebraic Attacks on Combiners with Memory. In *Advances in Cryptology—CRYPTO 2003*, Proceedings of 23rd Annual International Cryptology Conference, Santa Barbara, CA, USA, 17–21 August 2003; Boneh, D., Ed.; Volume 2729, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2003; pp. 162–175.
12. Courtois, N.T.; Meier, W. Algebraic Attacks on Stream Ciphers with Linear Feedback. In *Advances in Cryptology—EUROCRYPT 2003*, Proceedings of International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, 4–8 May 2003; Biham, E., Ed.; Volume 2656, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2003; pp. 345–359.
13. Courtois, N.T. Fast Algebraic Attacks on Stream Ciphers with Linear Feedback. In *Advances in Cryptology—CRYPTO 2003*, Proceedings of 23rd Annual International Cryptology Conference, Santa Barbara, CA, USA, 17–21 August 2003; Boneh, D., Ed.; Volume 2729, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2003; pp. 176–194.

14. Lee, D.H.; Kim, J.; Hong, J.; Han, J.W.; Moon, D. Algebraic Attacks on Summation Generators. In *Fast Software Encryption—FSE 2004*, Proceedings of 11th International Workshop, Delhi, India, 5–7 February 2004; Roy, B.K., Meier, W., Eds.; Volume 3017, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2004; pp. 34–48.
15. Al-Hinai, S.; Batten, L.M.; Colbert, B.D.; Wong, K.K.H. Algebraic Attacks on Clock-Controlled Stream Ciphers. In *Information Security and Privacy—ACISP 2006*, Proceedings of 11th Australasian Conference, Melbourne, Australia, 3–5 July 2006; Batten, L.M., Safavi-Naini, R., Eds.; Volume 4058, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2006; pp. 1–16.
16. Kanso, A.A. Modified clock-controlled alternating step generators. *Comput. Commun.* **2009**, *32*, 787–799.
17. Hassanzadeh, M.M.; Helleseth, T. Algebraic attack on the More Generalized Clock-Controlled Alternating Step Generator. In Proceedings of the IEEE 2010 International Conference on Signal Processing and Communications (SPCOM), Bangalore, India, 18–21 July 2010; pp. 1–5.
18. Klein, A. *Stream Ciphers*; Springer: Berlin/Heidelberg, Germany, 2013.
19. Robshaw, M.J.B.; Billet, O. *New Stream Cipher Designs—The eSTREAM Finalists*; Springer: Berlin/Heidelberg, Germany, 2008.
20. Al-Hinai, S.Z.M. Algebraic Attacks on Clock-Controlled Steam Ciphers. Ph.D. Thesis, Doctoral Dissertation, Queensland University of Technology, Queensland, Australia, 2007.
21. Beth, T.; Piper, F. The Stop-and-Go Generator. In *Advances in Cryptology—EUROCRYPT 1984*, Proceedings of Workshop on the Theory and Application of Cryptographic Techniques, Paris, France, 9–11 April 1984; Beth, T., Cot, N., Ingemarsson, I., Eds.; Volume 209, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1985; pp. 88–92.
22. Gollmann, D.; Chambers, W.G. Clock-controlled shift registers: A review. *IEEE J. Sel. Areas Commun.* **1989**, *7*, pp. 525–533.
23. Rueppel, R.A. When Shift Registers Clock Themselves. In *Advances in Cryptology—EUROCRYPT 1987*, Proceedings of Workshop on the Theory and Application of Cryptographic Techniques, Amsterdam, The Netherlands, 13–15 April 1987; Chaum, D., Price, W.L., Eds.; Volume 304, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1988; pp. 53–56.
24. Günther, C.G. Alternating step generators controlled by deBruijn sequences. In *Advances in Cryptology—EUROCRYPT 1987*, Proceedings of Workshop on the Theory and Application of Cryptographic Techniques, Amsterdam, The Netherlands, 13–15 April 1987; Chaum, D., Price, W.L., Eds.; Volume 304, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1988; pp. 5–14.
25. Chambers, W.G.; Gollmann, D. Lock-in Effect in Cascades of Clock-Controlled Shift-Registers. In *Advances in Cryptology—EUROCRYPT 1988*, Proceedings of Workshop on the Theory and Application of Cryptographic Techniques, Davos, Switzerland, 25–27 May 1988; Barstow, D., Brauer, W., Brinch Hansen, P., Gries, D., Luckham, D., Moler, C., Pnueli, A., Seegmuller, G., Stoer, J., Wirth, N., Gunther, C.G., Eds.; Volume 330, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1988; pp. 331–344.

26. Coppersmith, D.; Krawczyk, H.; Mansour, Y. The Shrinking Generator. In *Advances in Cryptology—CRYPTO 1993*, Proceedings of 13th Annual International Cryptology Conference, Santa Barbara, CA, USA, 22–26 August 1993; Stinson, D.R., Ed.; Volume 773, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1994; pp. 22–39.
27. Meier, W.; Staffelbach, O. The Self-Shrinking Generator. In *Advances in Cryptology—EUROCRYPT 1994*, Proceedings of Workshop on the Theory and Application of Cryptographic Techniques, Perugia, Italy, 9–12 May 1994; Santis, A.D., Ed.; Volume 905, Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1995; pp. 205–214.
28. Kanso, A.A. Modified self-shrinking generator. *Comput. Electr. Eng.* **2010**, *36*, 993–1001.
29. Liptak, B. *Instrument Engineers' Handbook: Process Software and Digital Networks*; CRC Press: Boca Raton, FL, USA, 2002.
30. Schneier, B. *Applied Cryptography*; Wiley: Hoboken, NJ, USA, 1996.
31. Vainio, J.T. *Bluetooth Security*; Helsinki University of Techinology: Espoo, Finland, 2000.
32. Hawkes, P.; Rose, G.G. Primitive Specification and Supporting Documentation for SOBER-t32; In Proceedings of the First Open NESSIE (New European Schemes for Signature, Integrity, and Encryption) Workshop, Leuven, Belgium, 13–14 November 2000.
33. Hell, M.; Johansson, T.; Meier, W. Grain: A Stream Cipher for Constrained Environments; *Int. J. Wirel. Mob. Comput.* **2007**, *2*, 86–93.
34. Lidl, R.; Niederreiter, H. *Introduction to Finite Fields and Their Applications*; Cambridge University Press: Cambridge, UK, 1986.
35. Lay, D.C. *Linear Algebra and Its Applications*; Addison Wesley: Boston, MA, USA, 2005.
36. Rueppel, R.A. *Analysis and Design of Stream Ciphers*; Springer: Berlin/Heidelberg, Germany, 1986.
37. Chambers, W.G.; Jennings, S.M. Linear Equivalence of Certain BRM Shift Register Sequences. *Electr. Lett.* **1984**, *20*, 1018–1019.
38. Rosen, K.H. *Elementary Number Theory and Its Applications*; Addison-Wesley: Boston, MA, USA, 2000.
39. Wang, Q.; Johansson, T.; Kan, H. Some results on fast algebraic attacks and higher-order nonlinearities. *Inf. Sec. IET* **2012**, *6*, 41–46.
40. Atkinson, K. A. *An Introduction to Numerical Analysis*; Wiley: Hoboken, NJ, USA, 1989.
41. Khazaei, S.; Fischer, S.; Meier, W. Reduced Complexity Attacks on the Alternating Step Generator. In *Selected Areas in Cryptography—SAC 2007*, Proceeding of 14th International Workshop, Ottawa, Canada, 16–17 August 2007; Adams, C., Miri, A., Wiener, M., Eds.; Volume 4876, Lecture Notes in Computer Science; Springer: Berlin Heidelberg, Germany, 2007; pp. 1–16.