S E S ' 2 0 0 5

Scientific Conference "SPACE, ECOLOGY, SAFETY" with International Participation
10–13 June 2005, Varna, Bulgaria

SOFTWARE IMPLEMENTATION OF P–ADIC SELF–SHRINKING GENERATOR FOR AEROSPACE CRYPTOGRAPHIC SYSTEMS

Zhaneta N. Tasheva, Borislav Y. Bedzhev

NMU "V. Levski", Faculty of Artillery and Air Defense,
Karel Shorpil Str. 1, 9713 Shoumen, e-mail: tashevi86@yahoo.com, bedzhev@mail.pv-ma.bg

Key words: Cryptography, Encryption Algorithm, Stream Ciphers, PRNG, FCSR.

Abstract: *To be suitable for use in aerospace cryptographic systems software-oriented stream ciphers must be fast, uniform, scalable, consistent and unpredictable. With regard in the paper the software implementation of a fast stream cipher, named Self–Shrinking p–adic Generator which produces 8 bits (SSPG-8) in one clock cycle, is proposed. The theoretical base of Self-Shrinking p-adic Generator is recalled. The software implementation of p-adic Self–Shrinking Generator is described. Analysis of more than 300 aerospace images is presented. The results from statistical analysis show that the sequence, generated by p-adic SSPG-8, is appropriate for a particular aerospace cryptographic application.*

INTRODUCTION

The need of software-oriented stream ciphers in modern aerospace cryptographic systems has rapidly grown during the last several years. To be suitable for use in aerospace the stream ciphers must be fast, uniform, scalable, consistent and unpredictable. One of the most used cryptographic systems is the binary additive stream cipher in whitch the keystream, the plaintext and the ciphertext are basic binary sequences. The keystream is generated from a keystream generator, which takes a secret key as a seed, and produces a long pseudorandom sequence. The ciphertext is generated by bitwise modulo 2 additions of the keystream and the plaintext.

The main goal of software-oriented stream cipher design is to generate efficient pseudorandom sequence [3], [4], [5] witx fast software algorithm and with truly random sequences properties. A fast software implementation of pseudo random number generator (PRNG), named Self–Shrinking p-adic Generator – 8 bits (SSPG–8), is proposed and an analysis of encrypted with SSPG–8 images is given in this paper.

THE THEORETICAL BASICS OF SELF-SHRINKING P-ADIC GENERATOR

In this section the theoretical basic of a recently proposed Self-Shrinking p–adic Generator (SSPG) [7] and some basic SSPG properties will be recalled.

A. THE SSPG ARCHITECTURE

In contrast with the classic self-shrinking generator [3] the SSPG architecture (Fig. 1) uses a p-adic FCSR instead of LFSR. This allows the generator to produce a number in the range 0 to p–1 in one step (a_i = [0, 1, …, p–1]). The self-shrinking p-adic generator selects a portion of the output p-adic FCSR sequence by controlling of the p-adic FCSR itself using the following algorithm.

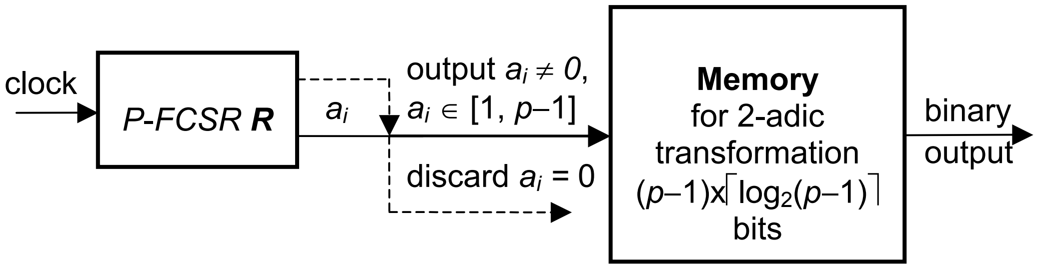

clock
P-FCSR R
a_i
output $a_i \neq 0$, $a_i \in [1, p-1]$
discard $a_i = 0$
Memory
for 2-adic transformation
$(p-1)\times\lceil \log_2(p-1) \rceil$ bits
binary output

Fig. 1. Self-Shrinking p-adic Generator

Definition 1: The algorithm of the Self-Shrinking p-adic Generator (Fig. 1) consists of the following steps:

1. The p-adic FCSR R is clocked with clock sequence with period τ_0.

2. If the p-adic FCSR output number is not equal to 0 ($a_i \neq 0$), the output bit forms a part of the p-adic SSPG sequence. Otherwise, if the output number of the p-adic FCSR is equal to 0 ($a_i = 0$), the p-adic output number of SSPG is discarded.

3. The shrunken p-adic SSPG output sequence is transformed in 2-adic sequence in which every p-adic number is presented with $\lceil \log_2 (p-1) \rceil$ binary digits, where $\lceil x \rceil$ is the smallest integer which is greater or equal to x. Every output number i from 1 to p–1 of p-adic SSPG sequence is depicted with p–adic expansion of the number:

$$i - 1 + \frac{2^{\lceil \log_2 (p-1) \rceil} - (p-1)}{2}. \tag{1}$$

The proposed SSPG uses the generalization of 2-adic FCSRs [1], [2] with stage contents and feedback coefficients in Z/(p) where p is a prime number, not necessarily 2.

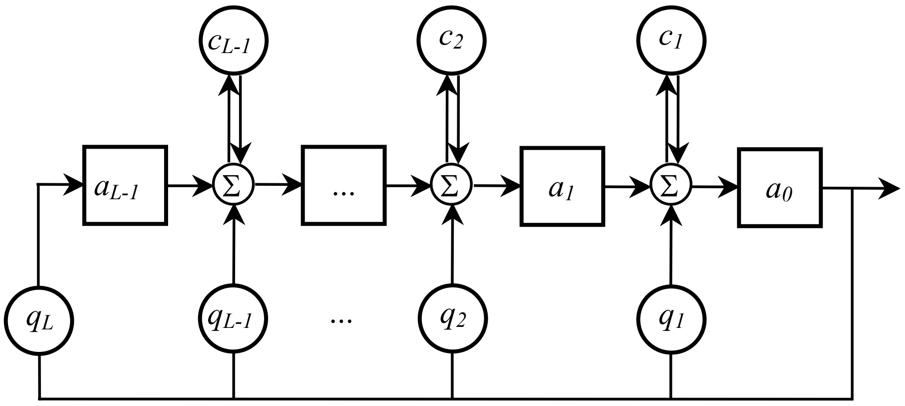


Fig. 2. Galois FCSR

Definition 2: A p-adic feedback with carry shift register with Galois architecture of length L (Fig. 3) consists of L stages (or delay elements) numbered 0, 1, …, L-1, each capable to store one p-adic (0, 1, …, p-1) number and having one input and one output; and a clock which controls the movement of data. During each clock cycle the following operations are performed:

1. The content of stage 0 is output and forms part of the output sequence;

2. The sum modulo p after stage i is passed to stage i - 1 for each i, $1 \leq i \leq L-1$;

3. The output of the last stage 0 is introduced into each of the tapped cells simultaneously, where it is fully added (with carry) to the contents of the preceding stages.

The $q_1, q_2, \ldots, q_L$ are the feedback multipliers and the cells denoted with $c_1, c_2, \ldots, c_{L-1}$ are the memory (or “carry”) bits. If

$$q = -1 + q_1 p + q_2 p^2 + \ldots + q_L p^L \tag{2}$$

is the base p expansion of a positive integer:

$$q \equiv -1 \pmod{p}, \tag{3}$$

then q is a connection integer for a FCSR with feedback coefficients $q_1, q_2, \ldots, q_L$ in Z/(p).

With each clock cycle, the integer sums:

$$\sigma_j = a_j + a_0 q_j + c_j \tag{4}$$

is accumulated.

At the next clock cycle this sum modulo p

$$a'_{j-1} = \sigma_n \pmod{p} \tag{5}$$

is passed on to the next stage in the register, and the new memory values are:

$$c'_j = \sigma_n \ (\operatorname{div} p). \tag{6}$$

B. The SSPG Properties

In this subsection some SSPG properties and theorem will be recalled. The theorem proofs can be found in [7]. The SSPG properties can be generalized as follows:

1. The **unlinearity** of SSPG is guaranteed by the fact that it is unknown at which positions the FCSR-sequence is shrunken. As a result the linear algebraic structure of the original FCSR-sequence is destroyed.

2. The SSPG implementation is very **fast** because the pseudorandom generator produces $\lceil \log_2 (p-1) \rceil$ binary digits in one step.

3. The SSPG sequence have **large period**. It is determined by the next theorem:

Theorem 1: The period of the self-shrunken p-adic generator realized by maximum length p-adic FCSR of length L and connection integer q is:

$$T_{SSPG} = T^* . \lceil \log_2 (p-1) \rceil, \tag{7}$$

where T* is the number of output p-adic FCSR numbers different from 0.

Here will be remind [1], [2] that the period of p-adic FCSR is T = q – 1, and within the period of a p-adic FCSR sequence each of p–adic numbers from 0 to p – 1 appears with approximately equal probability.

4. The SSPG sequences are **balanced**. It follows from the Theorem 2:

Theorem 2: The self-shrunken output SSPG sequence generated by maximum length p-adic FCSR of length L and connection integer q is a balanced sequence. Moreover:

a) If the prime p can be present in form $p = 2^n + 1$, the numbers of 0s and 1s in the self-shrunken output SSPG sequence are balanced and equal to:

$$N_{0s} \approx N_{1s} \approx \left\lceil \frac{q-1}{2^n+1} \right\rceil . 2^{n-1} . n . \tag{8}$$

b) if the prime p satisfies the inequality:

$$p < 2^{\lceil \log 2\,(p-1) \rceil} + 1 \qquad (9)$$

the numbers of 0s and 1s in the self-shrunken output SSPG sequence are balanced and equal to:

$$N_{0s} \approx N_{1s} \approx \left\lceil \frac{q-1}{2^n+1} \right\rceil . \frac{\lceil \log_2(p-1) \rceil . (p-1)}{2} . \qquad (10)$$

5. **No possibility exists to appear subsequences** of 2⌈log2 (p–1)⌉ consecutive 1s or 0s in SSPG sequences. It is guaranteed by step 3 in Definition 1.

SOFTWARE IMPLEMENTATION OF SELF–SHRINKING P–ADIC GENERATOR WHICH PRODUCES 8 BITS IN ONE CLOCK

As mentioned, the software-oriented steam ciphers must be fast. To satisfy this requirement a p-adic Self-Shrinking Generator which produces 8 bits in one clock cycle is picked out. According to this the prime p of SSPG must be in the range [129, 257], where are 24 primes, given in Table 1.

Table 1: The primes in the range [129, 257]

131	137	139	149	151	157	163	167	173	179	181	191
193	197	199	211	223	227	229	233	239	241	251	257

Definition 3: The Self–Shrinking p–adic Generator, which seed depends on primes in the range [129 = 2^7 + 1, 257= 2^8 + 1], generates 8 bits in one clock cycle. This generator will be called SSPG–8 in the rest of the paper.

All binary presentations of numbers from 0 to 255 will appear in the SSPG–8 output only if the seed of SSPG–8 depends of prime p = 257. For the other primes in the Table 1 some binary presentations of the numbers from 0 to 255 will be discarding according to the rule, given in the step 3 of Definition 1. The binary presentations of some shrunken SSPG–8 binary outputs with primes p = 251 and p = 241 according to (1) are shown in Table 2.

Table 2: Binary presentation of SSG–8 output

p-adic number	Binary output		p-adic number	p-adic number	Binary output	
	p = 251	p = 241			p = 251	p = 241
1	00000010	00001000	…	235	11101110	11101110
2	00000011	00001001	…	236	11101111	11101111
3	00000100	00001010	…	237	11110000	11110000
4	00000101	00001011	…	238	11110001	11110001
5	00000110	00001100	…	239	11110010	11110010
6	00000111	00001101	…	240	11110011	11110011
7	00001000	00001110	…	241	11110100	-
8	00001001	00001111	…	242	11110101	-
9	00001010	00010000	…	243	11110110	-
10	00001011	00010001	…	244	11110111	-
11	00001100	00010010	…	245	11111000	-
12	00001101	00010011	…	246	11111001	-
13	00001110	00010100	…	247	11111010	-
14	00001111	00010101	…	248	11111011	-
15	00010000	00010110	…	249	11111100	-
16	00010001	00010111	…	250	11111101	-

The software implementation of SSPG–8 is realized in Visual C++ environment. It uses the base template class FCSR <class T> [6], which provide a straightforward way of abstracting type information. The template class FCSR <class T> models the p–adic FCSR definition 2. Using templates, the design class FCSR can operate on data of many types and the code is generated for a template class and its functions only when they are instantiated.

The seed of SSPG-8 depends on 4 components:

- the prime p in the range [129, 257]. Here we will mention that the software implementation can work with all numbers in above range, but when p is not prime number the generated pseudo-random sequence won't be the maximum length sequence [1], [2].
- the initial state of the p-adic FCSR;
- the initial memory of the p-adic FCSR;
- the feedback polynomial of the p–adic FCSR.

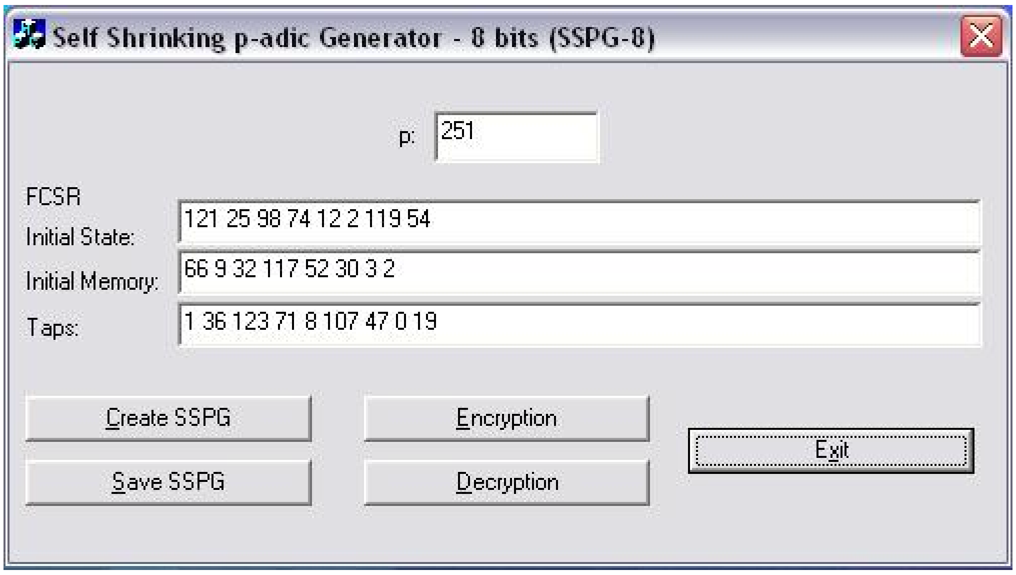
The software application "Self-Shrinking p–adic Generator" (Fig. 3) realizes SSPG–8s with primes $p \in [129, 257]$. Its general opportunities are:

1. The SSPG–8 seed has the length up to 3*256*8*24 = 147 456 bits if the initial state, the initial memory and the feedback polynomial of the control p–adic FCSR are with length 256 p-adic numbers. The coefficient 8 is used to present every p–adic number and the coefficient 24 gives the number of primes in the range [129, 257].

2. The SSPG–8 seed may be changed by modifying the prime p, feedback polynomial, initial state and initial memory of p–adic FCSR.

3. The output text file of SSPG–8 pseudo random number sequence with arbitrary bit length, defined by user can be saved. This file can be used for statistical tests.

4. The arbitrary files (.txt, .bmp, .bin, *.*) can be encrypted and decrypted with SSPG–8.


Self Shrinking p-adic Generator - 8 bits (SSPG-8)

p: 251

FCSR
Initial State: 121 25 98 74 12 2 119 54
Initial Memory: 66 9 32 117 52 30 3 2
Taps: 1 36 123 71 8 107 47 0 19

Fig. 3: Software application "Self–Shrinking p–adic Generator – 8 bits""

The properties of SSPG-8 cryptographic generator were been analyzing by means of more then 300 colour BMP images encrypted with SSPG-8 sequences with different seeds. The two used SSPG–8s with its polynomials are presented in Table 3 and the corresponding images, encrypted with these SSPG–8s, are given on Fig. 4. As one can see from Fig. 4 the image outlines can be identified if the length of used p-adic FCSR is less then 3. This fact follows from the chosen magnitude of prime p.

Table 3: Polynomials for SSPG–8 elements

PRNG	SSPG Elements	Polynomials
SSPG–8_1 p = 251	Initial State	121 25 98 74 12 2 119 54
	Initial Memory	66 9 32 117 52 30 3 2
	Taps	1 3 12 7 8 10 4 0 1
SSPG--8_2 p =241	Initial State	98 200 117
	Initial Memory	57 90 4
	Taps	1 36 123 71 8 107 47 0 19

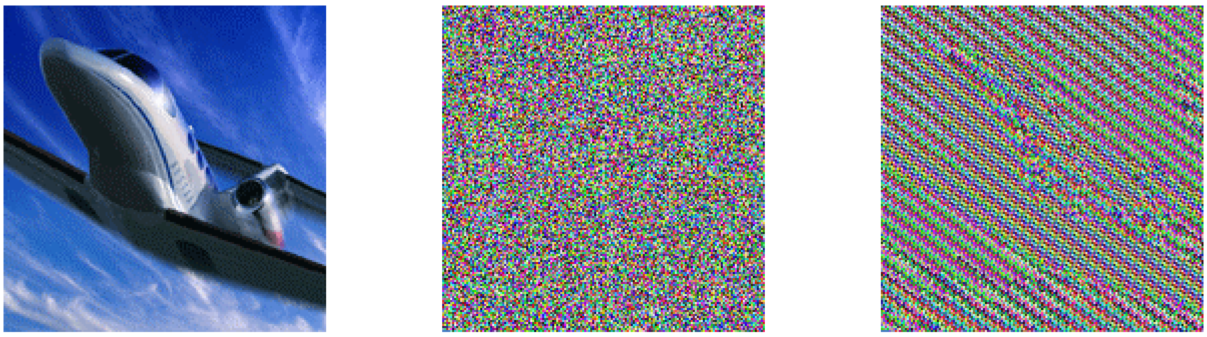

Original image
Encrypted with SSPG-8_1 image
Encrypted with SSPG-8_2 image

Fig. 4: Original and Encrypted with SSPG–8s images

CONCLUSIONS

The statistical analysis of all images encrypted by above described software implementation of SSPG–8 leads to the following conclusions:

1. The generated pseudo random sequences are **uniform, balanced** and have **large period**.

2. The images are **well encrypted**, i.e. the image outlines can't be identified even if the length of used p-adic FCSR is small (see Fig. 4).

3. The values of three basic colour components R (Red), G (Green) and B (Blue) are **uniformly distributed** on the image size (see encrypted with SSPG–8_1 image on Fig. 4).

4. The proposed software implementation of SSPG–8 is very **fast** because the generator forms 8 binary digits (1 byte) in one clock step.

REFERENCES

[1] M. Goresky, A. Klapper, "Fibonacci and Galois Representations of Feedback-With-Carry Shift Registers", IEEE Trans. Inform. Theory, vol. 48, pp. 2826−2836, November 2002.

[2] A. Klapper, M. Goresky, "Feedback Shift Registers, 2-adic Span, and Combiners With Memory", Journal of Cryptology, Volume 10, Number 2, 1997, pp. 111-147, http://www.math.ias.edu/~goresky/pdf/2adic.jour.pdf

[3] W. Meier, O. Staffelbach, "The Self-Shrinking Generator", Proceedings of Advances in Cryptology, EuroCrypt '94, Springer-Verlag, pp. 205-214, 1998.

[4] P. van Oorshot, A. Menezes, S. Vanstone, "Handbook of Applied Cryptography", CRC Press, 1997.

[5] Schneier, "Applied Cryptography", John Wiley & Sons, New York, 1996.

[6] Zh. N. Tasheva, "An Algorithm for Fast Software Encryption", accepted for International Conference on Computer Systems and Technologies - CompSysTech' 2005, 16-17 June 2005, Technical University, Varna, Bulgaria, will be published.

[7] Zh. N. Tasheva, B. Y. Bedzhev, B. P. Stoyanov, "Self-Shrinking p-adic Cryptographic Generator", accepted for XL International Scientific Conference on Information, Communication and Energy Systems and Technologies ICEST 2005, June 29 – July 1, 2005, Nish, Serbia and Montenegro, will be published.