

# Building Stream Ciphers from FCSRs

Dirk Stegemann

Theoretical Computer Science  
University of Mannheim  
68131 Mannheim, Germany

## 1 Introduction

Stream ciphers are commonly used for fast encryption of arbitrarily long data streams. A common design principle is to use a keystream generator to produce a pseudorandom stream of running key bits  $\mathbf{z} = (z_t)_{t \geq 0}$ , which is added bitwise to the plaintext stream  $(p_t)_{t \geq 0}$  in order to obtain the ciphertext  $(c_t)_{t \geq 0}$  with  $c_t = p_t \oplus z_t$ . The receiver computes the same keystream  $\mathbf{z}$  and recovers the plaintext via  $p_t = c_t \oplus z_t$ . Many keystream generators operate as finite state machines (FSMs). Their initial state is derived from a secret key  $\mathcal{K}$  and a public initialization vector IV by a procedure named key/IV setup, and in each clock  $t$  the FSM outputs a piece of keystream and changes its state according to the state transition function. For the construction of the FSM, linear feedback shift registers (LFSRs) have been widely used in practical applications due to their pseudorandomness properties and their efficiency. However, in the light of recent cryptanalysis results on LFSR-based stream ciphers (especially correlation attacks and algebraic attacks) it seems appropriate to look for other constructions that provide similar pseudorandomness properties but are more resistant to known attacks. Feedback with carry shift registers (cf. for example [4]), which we address in this paper, have been identified as possible alternatives.

## 2 Preliminaries

We call a state of a finite state machine (e.g., an FCSR) *periodic* if, left to run, the machine will return to that same state after a finite number of steps. We call a sequence  $\mathbf{u} = (u_i)_{i \geq 0}$  *strictly periodic* (or simply *periodic*) with period  $T$  if  $u_{i+T} = u_i$  for all  $i \geq 0$ . We call a sequence  $\mathbf{u}$  *eventually periodic* if there exists a  $t \geq 0$  such that  $\mathbf{u}' = (u_i)_{i \geq t}$  is periodic.

By  $\text{wt}(d)$ , we denote the Hamming weight of a binary vector  $d$ . We will identify a vector  $(u_0, \dots, u_{k-1}) \in \{0, 1\}^k$  with the integer  $u = \sum_{i=0}^{k-1} u_i 2^i$ .

A 2-adic integer is a formal power series  $\alpha = \sum_{i=0}^{\infty} u_i 2^i$  with  $u_i \in \{0, 1\}$ . The collection of all such formal power series forms the ring of 2-adic numbers. This ring especially contains rational numbers  $p/q$ , where  $p$  and  $q$  are integers and  $q$  is odd. 2-adic numbers and eventually periodic sequences are linked by the following Theorem [1].

**Theorem 1** *There is a one-to-one correspondence between rational numbers  $\alpha = p/q$  (with odd  $q$ ) and eventually periodic binary sequences  $\mathbf{u}$  which associates to each such rational number  $\alpha$  the bit sequence  $\mathbf{u} = (u_0, u_1, \dots)$  of its 2-adic expansion. The sequence  $\mathbf{u}$  is strictly periodic if and only if  $\alpha \leq 0$  and  $|\alpha| \leq 1$ .*

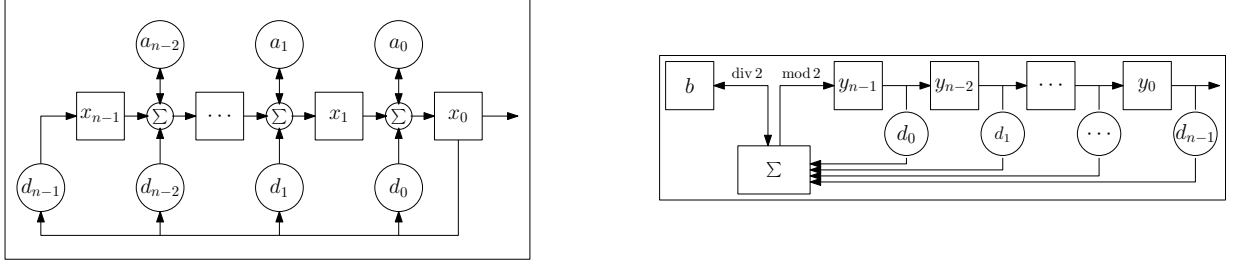


Figure 1: FCSR in Galois and Fibonacci architecture

### 3 Feedback With Carry Shift Registers (FCSRs)

#### 3.1 Galois and Fibonacci Architectures

An FCSR of length  $n$  in Fibonacci architecture contains  $n$  binary register cells  $(y_0, \dots, y_{n-1})$  with fixed binary feedback taps  $(d_0, \dots, d_{n-1})$  and an additional  $l$ -bit memory  $b$ , see Fig. 1. From an initial configuration  $(y, b)$ , the FCSR outputs in each clock  $t$  the value  $y_0$ , computes the sum  $\sigma = b + \sum_{i=0}^{n-1} y_i d_{n-i-1}$  over the integers and updates the register and memory according to  $b = \sigma \text{ div } 2$  and  $y = (\sigma \text{ mod } 2, y_{n-1}, \dots, y_1)$ . If the Fibonacci FCSR is in a periodic state, we have  $0 \leq b < \text{wt}(d)$ , i.e., we will need at most  $\lfloor \log_2(\text{wt}(d) - 1) \rfloor + 1$  memory bits to store  $b$  [4].

An FCSR of length  $n$  in Galois architecture contains  $n$  binary register cells  $x_i$  with fixed binary feedback taps  $(d_0, \dots, d_{n-1})$  and  $n - 1$  memory cells  $a_0, \dots, a_{n-2}$ , see Fig. 1. Starting from an initial configuration  $(x, a)$ , the Galois FCSR outputs in each clock the value  $x_0$ , computes the sums  $\sigma_i = x_{i+1} + a_i d_i + x_0 d_i$  for  $0 \leq i < n$  (with  $x_n = 0$  and  $a_{n-1} = 0$ ) and updates  $x_i$  to  $\sigma_i \text{ mod } 2$  and  $a_i$  to  $\sigma_i \text{ div } 2$  for all  $0 \leq i < n - 1$ . We will assume that memory cells are only present at those positions with feedback taps, i.e.,  $a_i = 0$  if  $d_i = 0$  for all  $0 \leq i < n - 1$ .

#### 3.2 Properties of FCSR-Sequences

Identify a Galois state  $(x, a)$  with the integer  $p = x + 2a$ , a Fibonacci state  $(y, b)$  with the integer

$$p = b2^n - \sum_{k=0}^{n-1} \sum_{i=0}^k q_i y_{k-i} 2^k,$$

and define for both architectures the connection integer  $q$  as  $q = 1 - 2d$ . Then the output of the FCSR is the 2-adic expansion of  $\alpha = \frac{p}{q}$  (cf. [1], Proposition 1, and [4], Theorem 2.4). Theorem 1 implies that the output will be strictly periodic if and only if  $0 \leq p \leq |q|$ .

**Proposition 2** *For an FCSR with connection integer  $q$  and a periodic initial state corresponding to  $p \in \{0, \dots, |q| - 1\}$ , the sequence of integer representations of the states  $(p_t)_{t \geq 0}$  is given by  $p_t = 2^{-t} p \text{ mod } q$  and the  $t$ -th output bit can be computed as  $z_t = p_t \text{ mod } 2 = (2^{-t} p \text{ mod } q) \text{ mod } 2$ .*

If  $0 < p < |q|$ ,  $q$  odd, and  $p$  and  $q$  are coprime, Prop. 2 implies that the period of the sequence  $(p_t)_{t \geq 0}$  is the order of 2 modulo  $q$  (cf. Theorem 2 in [1]). The period reaches its maximum  $|q| - 1$  if  $q$  is a prime for which 2 is a primitive root. We call such FCSRs *maximum-length FCSRs* and the

sequences they produce *l-sequences*. The properties of 2-adic numbers imply that an *l*-sequence consists of two half-periods, where the second half is the binary complement of the first [1].

For a binary sequence  $u$ , we denote by its *2-adic complexity* the length of the shortest FCSR that produces the sequence. The 2-adic complexity along with an actual FCSR that produces the given sequence can be efficiently computed using FCSR-synthesis algorithms (cf. [6] for an overview). Clearly, the 2-adic complexity is upper-bounded by the length of the sequence, and we can expect the 2-adic complexity of a random sequence to be close to this bound, whereas its value will be small for an FCSR-sequence. This bias can be used to efficiently distinguish FCSR-sequences from truly random sequences.

However, no other statistical tests except for computing the 2-adic complexity are known to reveal biases from the behaviour that is expected from truly random sequences. Particularly, the expected autocorrelation of *l*-sequences is zero [10], and their linear complexity is high [8]. Experiments indicate that FCSR-sequences pass the NIST statistical test suite [7].

### 3.3 Particular Properties of the Galois Architecture

Implementors will often prefer the Galois architecture to the Fibonacci architecture since the size of the memory is intrinsically limited and the memory bits can be updated in parallel.

For a Galois FCSR, we have  $0 \leq p \leq |q|$ . The output corresponds to the all-zero sequence if  $p = 0$  and to the all-one sequence if  $p = |q|$  [1]. We call Galois states  $(x, a)$  with  $p \in \{0, |q|\}$  *invariant* states.

The sequence  $(x_{i,t})_{t \geq 0}$  of values taken by the main register cell  $i$  is again an FCSR-sequence, more precisely the 2-adic expansion of  $p_i/q$  with  $p_i = F_i(x, a) \cdot q + M_i \cdot p$ ,  $F_i(x, a) = \sum_{j=i}^{n-1} (x_j + 2a_j)2^{j-i}$ , and with constants  $M_i = 2 \sum_{j=i}^{n-1} d_j 2^{j-i}$  [2]. This expression can be further simplified for periodic initial states as follows [3].

**Proposition 3** *For a maximum-length Galois FCSR with connection integer  $q$ , a periodic initial state  $(x(0), a(0))$ , and  $p_0 = x(0) + 2a(0)$ , the sequence  $(x_{i,t})_{t \geq 0}$  of values taken by a fixed main register cell  $i$  corresponds to  $(p_{t+s_i} \bmod 2)_{t \geq 0}$  with  $s_i = -\log_2(M_i) \bmod q$  and  $M_i = 2 \sum_{j=i}^{n-1} d_j 2^{j-i}$ .*

Proposition 3 implies that the sequence  $(x_{i,t})_{t \geq 0}$  corresponds to the sequence produced by the whole FCSR (which is in turn equal to the sequence  $(x_{0,t})_{t \geq 0}$ ) shifted by  $s_i$  positions. Note that the phase shifts  $s_i$  are independent of the initial state  $p$  and depend on  $i$  and  $q$  only.

Note that although equivalent states produce the same output, the sequence of states  $(x(t), a(t))_{t \geq 0}$  obtained by running the FCSR from equivalent starting states may be different.

### 3.4 Mappings between Galois and Fibonacci States

There is an onto function  $E : \{(x, a)\} - \{(1, \dots, 1; a_0, \dots, a_{n-2})\} \rightarrow \mathbb{Z}_{|q|}$  that assigns to a Galois state  $(x, a)$  the number  $E(x, a) = x + 2a \bmod |q|$  [4]. Moreover, there exists a one to one mapping  $S$  from  $\mathbb{Z}_{|q|}$  onto the set  $L$  of strictly periodic states of the Fibonacci FCSR with connection integer  $q$  except for the state  $(1, \dots, 1; \text{wt}(q+1) - 1)$ . More precisely, the bijective mapping  $S$  assigns to a  $p \in \mathbb{Z}_{|q|}$  the initial Fibonacci main register state  $y_i = ((2^{-i}p \bmod q) \bmod 2)$  for  $0 \leq i \leq n-1$  and the initial memory state

$$b = \frac{1}{2^n} \left( p + \sum_{k=0}^{n-1} \sum_{i=0}^k q_i y_{k-i} 2^k \right) .$$

Conversely, for a given periodic Fibonacci state  $(y, b)$  the corresponding integer  $p$  will satisfy  $0 \leq p < |q|$ .

Obviously, the mapping  $E$  from the Galois states to  $\mathbb{Z}_{|q|}$  is not one to one, i.e., generally more than one state is mapped to the same  $p \in \mathbb{Z}_{|q|}$ . However, we can compute for a given  $p \in \mathbb{Z}_{|q|}$  the uniquely determined corresponding periodic state  $(x, a)$  [3].

**Proposition 4** *For all  $p \in \mathbb{Z}_{|q|}$ , the only strictly periodic state  $(x, a)$  with  $x+2a = p$  of a maximum-length Galois FCSR with connection integer  $q$  is given by  $x_i = M_i \cdot p \bmod q \bmod 2$  and  $a = (p-x)/2$  with  $M_i$  defined as in Prop. 3.*

For an initial state of a Galois FCSR with connection integer  $q$ , we can thus compute a (periodic) initial state of a Fibonacci FCSR with connection integer  $q$  (and vice versa) such that the two registers will produce the same output (cf. [4], Corollary 3.7). Moreover, we can map between periodic Fibonacci states and periodic Galois states using Prop. 4.

**Corollary 5** *Let  $R_g$  and  $R_f$  denote Galois resp. Fibonacci FCSRs with the same connection integer  $q$ . Then there is a bijective mapping  $T$  between the periodic states of  $R_g$  and  $R_f$ .*

## 4 From FCSRs to Stream Ciphers

We have observed in Sect. 3.2 that FCSR-sequences and especially  $l$ -sequences have many desirable pseudorandomness properties but low 2-adic complexity, which prevents them from being used directly as keystream. In the context of LFSRs, which suffer from a similar weakness (their low, efficiently computable linear complexity), popular approaches are combination and filter generators. A combination generator consists of a small number of feedback shift registers and a Boolean function  $f$  that combines the output sequences of the internal registers in order to produce the output keystream. A filter generator contains only one feedback shift register and a Boolean filter function  $g$  that produces the output keystream from the current content of certain register cells.

It seems natural to apply the ideas of combination and filter generators to the FCSR-context. This has been done in the case of the F-FCSR stream cipher family [1], which is to the best of our knowledge the first FCSR-based stream cipher. F-FCSR is a filter generator based on a single Galois FCSR and a filter function  $g$  that simply computes the binary XOR of its inputs. The initialization procedure of F-FCSR ensures that the initial state of the generator is periodic. Hence, by Prop. 3, the keystream generation procedure of F-FCSR is equivalent to taking the bitwise XOR-sum of different parts of the same FCSR-sequence. This design is motivated by the conjecture that linear and 2-adic operations are unrelated and that the correlation between two distant parts of the same FCSR-sequence is low. At present, we are not aware of any cryptanalytical results that contradict this assumption.<sup>1</sup>

Proposition 3 further implies that a Galois-based filter generator like F-FCSR can be equivalently represented as a combination generator based on Galois FCSRs that contains as many FCSRs as the filter function  $g$  has inputs, where the FCSR producing the sequence  $(x_{i,t})_{t \geq 0}$  is initialized with  $p_{s_i}$ . Furthermore, one or more Galois registers in the combination generator may be replaced by Fibonacci registers producing the same output according to Corollary 5. We can also build an equivalent filter generator based on a Fibonacci FCSR with the following result [3].

---

<sup>1</sup>Generic attack strategies have been reported in [1] and [9] but are less efficient than exhaustive search for well-chosen cipher parameters.

**Proposition 6** *The value  $x_i$  of the  $i$ -th cell in the main register of the Galois FCSR can be computed from the (strictly) periodic state  $(y, b)$  of the corresponding Fibonacci FCSR by*

$$x_i = M_i \left( b2^n - \sum_{k=0}^{n-1} \sum_{j=0}^k d_{j-1} y_{k-j} 2^k \right) \pmod{q} \pmod{2} .$$

Note that these representations are all equivalent in the sense that they produce the same keystream sequence from a given (possibly transformed) starting state, but one representation may be more suitable for further cryptanalysis than the other.

Finally, we want to point out that the security of the keystream generation (based on a secret initial state) does not imply the security of the whole cipher, since the initial state of the FSM is not entirely secret but partly depends on the publicly known initialization vector. In fact, many stream ciphers (particularly an early version of F-FCSR [5]) have been broken by exploiting weaknesses in the key/IV setup.

## References

- [1] F. Arnault and T. P. Berger. Design and properties of a new pseudorandom generator based on a filtered FCSR automaton. *IEEE Trans. Comp.*, 54(11):1374–1383, 2005.
- [2] F. Arnault, T. P. Berger, and M. Minier. Some results on FCSR automata with applications to the security of FCSR-based pseudorandom generators. *IEEE Trans. Inform. Theory*, 54(2):836–840, 2008.
- [3] S. Fischer, W. Meier, and D. Stegemann. Equivalent representations of the F-FCSR keystream generator. In Christophe de Cannière and Orr Dunkelman, editors, *The State of the Art of Stream Ciphers (SASC 2008) – Workshop Record*, pages 87–96, 2008.
- [4] M. Goresky and A. Klapper. Fibonacci and Galois representations of feedback-with-carry shift registers. *IEEE Trans. Inform. Theory*, 48(11):2826–2836, 2002.
- [5] É. Jaulmes and F. Muller. Cryptanalysis of the F-FCSR Stream Cipher Family. In B. Preneel and S. Tavares, editors, *Proc. of Selected Areas in Cryptography (SAC 2005)*, volume 3897 of *LNCS*, pages 20–35. Springer, 2005.
- [6] A. Klapper. A survey of feedback with carry shift registers. In T. Helleseht et al., editors, *Proc. of SETA 2004*, volume 3486 of *LNCS*, pages 56–71. Springer, 2004.
- [7] NIST. A statistical test suite for the validation of random number generators and pseudo random number generators for cryptographic applications. <http://csrc.nist.gov/rng/>, 2004.
- [8] C. Seo, S. Lee, Y. Sung, K. Han, and S. Kim. A lower bound on the linear span of an FCSR. *IEEE Trans. Inform. Theory*, 46(2):691–693, March 2000.
- [9] D. Stegemann. Extended BDD-based cryptanalysis of keystream generators. In C. Adams, A. Miri, and M. Wiener, editors, *Proc. of Selected Areas in Cryptography (SAC 2007)*, volume 4876 of *LNCS*, pages 17–35. Springer, 2007.
- [10] H. Xu and W.-F. Qi. Autocorrelations of maximum-length FCSR sequences. *SIAM J. Discrete Math.*, 20(3):568–577, 2006.