ACHTERBAHN:

A Proposal for a Profile 2 Stream Cipher to ECRYPT's Call for Stream Cipher Primitives

Berndt M. Gammel, Rainer Göttfert and Oliver Kniffler

Infineon Technologies AG
St.-Martin-Str. 76
81541 Munich
Germany

berndt.gammel@infineon.com
rainer.goettfert@infineon.com
oliver.kniffler@infineon.com

30 May 2005

Abstract

We propose a new additive binary stream cipher called *Achterbahn*. The keystream generator (KSG) consists of eight primitive binary nonlinear feedback shift registers (NLFSR's). A binary N-stage feedback shift register is called primitive if it has a cycle of length $2^N - 1$ containing all binary nonzero N-tuples. Each shift register has a configurable linear feedforward output function. The output sequences of the shift registers are combined by a balanced 4th-order correlation immune Boolean combining function of eight variables and of algebraic degree three. Due to the modifiable shift register output functions, the KSG is able to produce an ensemble of 2^{64} (respectively of 2^{80}) cyclically inequivalent sequences. All sequences have periods larger than 2^{207} and linear complexities larger than 2^{85}. The size of the secret key is 80 bits. The feedback functions of the driving NLFSR's promote *fast* hardware implementations. In the high-speed implementation a throughput of more than 8 Gbps is reached.

Keywords. Nonlinear feedback shift registers, additive stream ciphers.

1 Introduction

The proposed stream cipher *Achterbahn* is a binary additive stream cipher. In a binary additive stream cipher, the plaintext is given as a string $m_0, m_1, \ldots$ of elements of the finite field $\mathbb{F}_2$. The keystream $z_0, z_1, \ldots$ is a binary pseudo-random sequence. The sender encrypts the plaintext message according to the rule $c_t = m_t + z_t$ for all $t \geq 0$. The ciphertext $c_0, c_1, \ldots$ is decrypted by the receiver by adding bitwise the keystream $z_0, z_1, \ldots$ to the received ciphertext sequence $c_0, c_1, \ldots$. Sender and receiver produce the keystream $z_0, z_1, \ldots$ via identical copies of the *key stream generator* (KSG).

The basic ingredients of the keystream generator are eight binary nonlinear feedback shift registers (NLFSR's) of lengths between 22 and 31, and a balanced 4th-order correlation immune Boolean combining function $R : \mathbb{F}_2^8 \to \mathbb{F}_2$. The NLFSR's are such that they can produce binary sequences of period $2^N - 1$, where N is the length of the shift register. Each shift register has a constant nonlinear feedback function, governing the internal state of the shift register, and an adjustable linear feedforward output function. The output functions of the eight NLFSR's deliver the input sequences for the Boolean combining function R which in turn outputs the running key.

The output functions of the underlying feedback shift registers (FSR's), and thus the output function of the KSG, are changed after each resynchronization step. Sequences produced under different configurations are cyclically inequivalent. There are 2^{64} (respectively 2^{80}) possibilities for the configuration of the KSG, so that the KSG is able to generate an ensemble $\mathcal{E}$ of 2^{64} (respectively of 2^{80}) translation distinct periodic sequences each of which has period larger than 2^{207} and linear complexity larger than 2^{85}. The key-loading algorithm to be described below has the following property: Let the key $K \in \mathbb{F}_2^{80}$ be fixed, then any two different initial values IV and IV' always result in two different output functions of the KSG. As a consequence, the produced keystream segments between any two resynchronization steps belong to distinct sequences of the ensemble $\mathcal{E}$. Thus any unintended re-use of key material is excluded.

The NLFSR's have been selected under the objective to enable fast hardware implementations of the KSG. In a straightforward implementation, the KSG emits one bit of keystream per clock cycle. In the high-speed implementation, the KSG generates one byte of keystream within each clock cycle. The eight bits forming this byte are the same eight bits that could have been generated within eight clock cycles using the straightforward implementation. It is important to note that the acceleration of encryption and decryption speed is achieved via a special implementation of the underlying NLFSR's and by duplicating the Boolean combining function but without introducing any new cryptographic components in the design (like multi-output Boolean functions).

2 Detailed description of the keystream generator

The overall structure of the keystream generator is depicted in the Figure 1. The core of the KSG consists of eight primitive (in the sense of Definition 1 in Appendix A) binary

NLFSR's labelled with capital letters A, B, C, ..., H. Each NLFSR is endowed with a linear feedforward logic described by filter polynomials $a(x)$, $b(x)$, $c(x)$, ..., $h(x)$. The linear feedforward logics supply the Boolean combining function R with inputs. The function R then outputs the keystream. At the outset—under the control of the secret key K and a public initial value IV—all eight NLFSR's are loaded and all linear feedforward functions are adjusted.

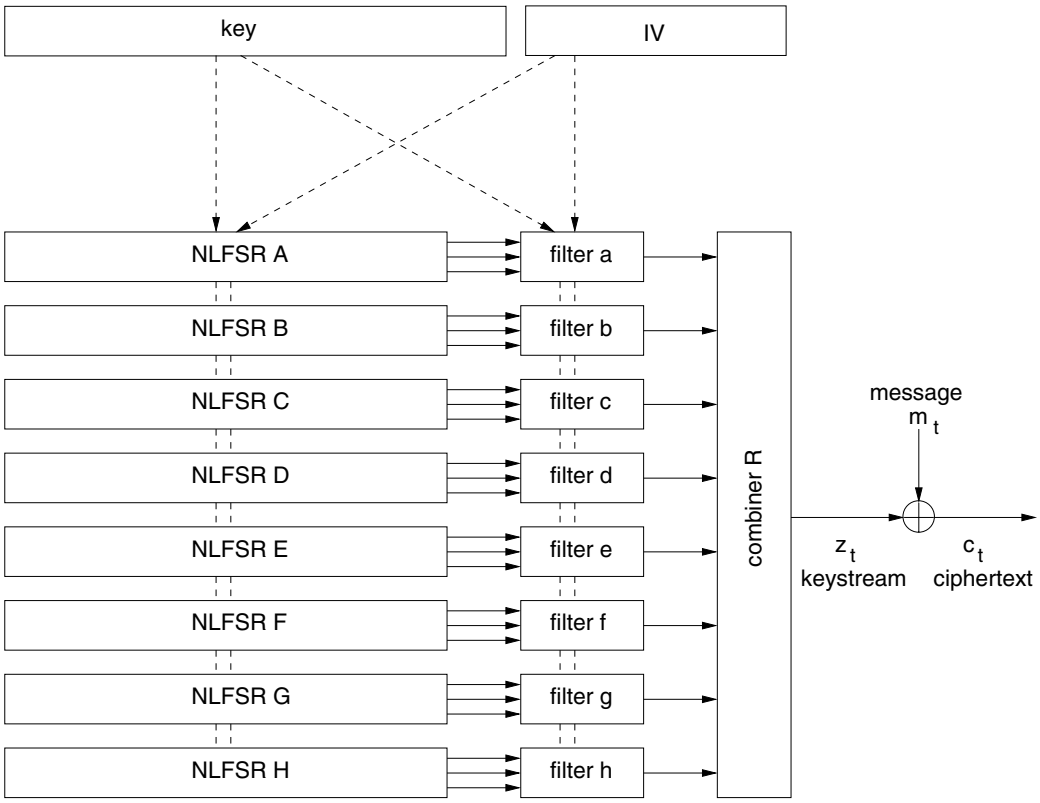


Figure 1: The keystream generator

2.1 The Boolean combining function

The Boolean combining function $R : \mathbb{F}_2^8 \to \mathbb{F}_2$ has algebraic degree 3 and nonlinearity 64. The algebraic normal form of R is given by

$$\begin{aligned} R(y_1, y_2, \ldots, y_8) = {} & y_1 + y_2 + y_3 + y_4 + y_5y_7 + y_6y_7 + y_6y_8 \\ & + y_5y_6y_7 + y_6y_7y_8. \end{aligned} \tag{1}$$

Using the logical OR-symbol $\vee$, defined by $a \vee b = a + b + ab$ for all $a, b \in \mathbb{F}_2$, the function R can be represented in the form

$$R(y_1, y_2, \ldots, y_8) = y_1 + y_2 + y_3 + y_4 + y_5y_7 \vee y_6y_7 \vee y_6y_8. \tag{2}$$

The Boolean function R is balanced and 4th-order correlation immune. The algebraic degree 3 is large enough to guarantee that the produced keystream $\zeta = (z_t)_{t=0}^{\infty}$ will have linear complexity larger than 2^{85}. The order 4 of correlation immunity is the maximum possible value for balanced, 8-variable, Boolean functions of algebraic degree 3 (see Siegenthaler [34]). There are other 8-variable Boolean functions of algebraic degree 3 having order of resiliency 4. The particular one presented in (1) and (2) was chosen because it has a simple realization in hardware. See Figure 2.

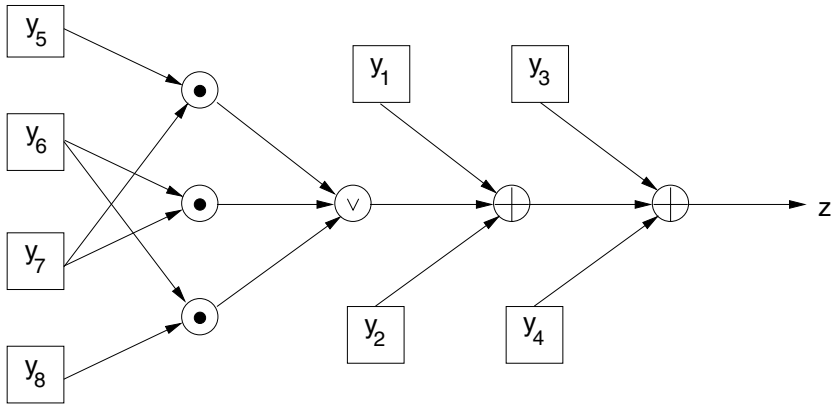


Figure 2: The Boolean Combining Function

In the high-speed implementation of the stream cipher, the KSG contains eight copies of the Boolean function R.

2.2 The feedback shift registers

The principal components of the KSG are eight binary primitive nonlinear feedback shift registers. Throughout the proposal these NLFSR's will be labelled by the capital letters A, B, C, $\ldots$, H. The lengths, periods, linear complexities, and nonlinearities of the eight NLFSR's are given in the following table. Periods and linear complexities of a binary primitive FSR are understood in the sense of Definition 3 in Appendix A.

NLFSR's				
label	length	period	linear complexity	nonlinearity
A	22	$2^{22}-1$	$L_A = 2^{22}-13$	30208
B	23	$2^{23}-1$	$L_B = 2^{23}-2$	245760
C	25	$2^{25}-1$	L_C	499712
D	26	$2^{26}-1$	L_D	233472
E	27	$2^{27}-1$	L_E	983040
F	28	$2^{28}-1$	L_F	237568
G	29	$2^{29}-1$	L_G	999424
H	31	$2^{31}-1$	L_H	999424

Of course, the listing of the periods is redundant, as a primitive binary FSR of length N has, by definition, period $2^N - 1$. Note that shift registers G and H have the same nonlinearities.

The computations of the linear complexities of the NLFSR's are currently (30 April 2005) in progress. For the NLFSR's A and B we found $L_A = 2^{22} - 13$ and $L_B = 2^{23} - 2$, respectively. One can with good reason expect that the linear complexities $L_C, \ldots, L_H$ of the remaining six feedback shift registers will also turn out to be close to the periods. This expectation is supported by experimental observations on a vast number of randomly selected primitive binary NLFSR's carried out by the authors over the last two years, as well as by theoretical investigations of Rueppel [31], Dai and Yang [8], and Meidl and Niederreiter [24], concerning the linear complexity of periodically repeated random strings.

At any rate until the computations of the linear complexities for the remaining NLFSR's will be completed, we shall be very conservative assuming that the linear complexities of the involved primitive NLFSR's are only greater than half the periods. Thus we shall assume in the sequel that $L_C \geq 2^{24}$, $L_D \geq 2^{25}$, $L_E \geq 2^{26}$, $L_F \geq 2^{27}$, $L_G \geq 2^{28}$, and $L_H \geq 2^{30}$. The feedback functions of the eight driving NLFSR's are presented in Appendix C.

There is another NLFSR contained in the KSG. This NLFSR is labelled by the capital letter V. The shift register V is nonsingular but not primitive. It has length 64 and nonlinearity 64700416. Feedback shift register V is used to determine the configuration of the output function of the KSG. The feedback function of shift register V is given by

$$\begin{aligned} V(x_0, x_1, \ldots, x_{63}) = {} & 1 + x_0 + x_3 + x_7 + x_{10} + x_{12} + x_{27} + x_{28} + x_{38} + x_{46} \\ & + x_{47} + x_8 x_{20} + x_{17} x_{23} + x_{24} x_{25} + x_{29} x_{31} + x_{33} x_{34} x_{37} \\ & + x_1 x_3 x_9 x_{10} + x_{39} x_{41} x_{51} x_{52}. \end{aligned}$$

Remark. It is likely that the requested IV-size for a PROFILE 2-stream cipher in ECRYPT's call for stream cipher primitives will be extended to 80 bits. See De Cannière, Lano and Preneel [9], and Hong and Sarkar [16]. In this case, the above NLFSR V of length 64 will be replaced by another NLFSR of length 80.

2.3 The linear feedforward functions

Each feedback shift register A, B, C, ..., H is endowed with a configurable linear feedforward output function. The linear feedforward output function can be described by the filter polynomial (see Appendix A). The binary filter polynomial $a(x)$ for NLFSR A has degree at most 6. All filter polynomials will have nonzero constant terms. Thus the polynomial $a \in \mathbb{F}_2[x]$ has the form

$$a(x) = a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + 1.$$

There are $2^6 = 64$ possibilities for the filter polynomial $a(x)$, corresponding to the six binary coefficients $a_1, a_2, \ldots, a_6$. For the NLFSR A, we depict the situation in Figure 3.

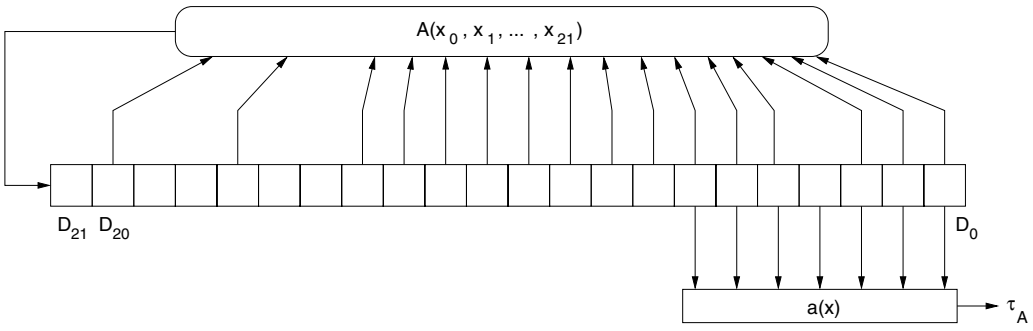


Figure 3: Linear feedforward function for NLFSR A

Enlarging the right part of Figure 3, we get:

If the coefficient $a_j = 1$, $1 \le j \le 6$, the corresponding wire in Figure 4 is connected and the content of cell D_j contributes to the output. If $a_j = 0$, the wire is disconnected and the content of D_j is ignored. Note that cell D_0 is always connected to the output line. If all six coefficients of $a(x)$ are zero, in other words, if the filter polynomial is equal to the constant polynomial $a(x) = 1$, the standard output sequence $\sigma_A = (s_n)_{n=0}^{\infty}$ is emitted. Using the shift operator T, defined by $T\sigma = (s_{n+1})_{n=0}^{\infty}$ for all $\sigma = (s_n)_{n=0}^{\infty} \in \mathbb{F}_q^{\infty}$, we can write the output sequence $\tau_A = (t_n)_{n=0}^{\infty}$ in the form $\tau_A = a(T)\sigma_A$.

The filter polynomials that will determine the output values of the eight shift registers A, B, C, ..., H are designated by $a(x)$, $b(x)$, $c(x)$, ..., $h(x)$, respectively. The degrees of the filter polynomials are restricted according to $\deg(a) \le 6$, $\deg(b) \le 7$, $\deg(c) \le 7$, $\deg(d) \le 8$, $\deg(e) \le 8$, $\deg(f) \le 9$, $\deg(g) \le 9$, and $\deg(h) \le 10$. Note that the sum of the maximum permissible degrees of all eight filter polynomials is 64. As a consequence, the KSG has 2^{64} different configurations for its output function.

Theorem 1. *If the NLFSR's A, B, C, ..., H are loaded with any nonzero initial state vectors, then for all filter polynomials a(x), b(x), c(x), ..., h(x), the produced output sequences τ_A, τ_B, τ_C, ..., τ_H have periods $2^N - 1$, where N is the length of the corresponding shift register.*

Proof. Let $m_A(x), m_B(x), m_C(x), \ldots, m_H(x)$ be the minimal polynomials associated with the NLFSR's A, B, C, ..., H, respectively. By Theorem 19 it suffices to check that each of the polynomials $m_A(x), m_B(x), m_C(x), \ldots, m_H(x)$ is divisible by a binary primitive polynomial of degree N, where N is the length of the corresponding shift register. Given a periodic binary sequence σ of period $P = 2^N - 1$, it can be checked whether or not a given binary polynomial f of degree N divides the minimal polynomial m_σ of σ without actually knowing the minimal polynomial m_σ. One merely has to check whether the polynomial $g(x) = (x^P - 1)/f(x)$ is still a characteristic polynomial of σ. The polynomial f divides m_σ precisely if g is not a characteristic polynomial of σ. See Corollary 1. In this manner we verified that each minimal polynomial $m_A(x), m_B(x), m_C(x), \ldots, m_H(x)$ is divisible by some primitive binary polynomial of the required degree. □

Theorem 2. *If the NLFSR's A, B, C, ..., H are loaded with any nonzero initial state vectors, then for all filter polynomials a(x), b(x), c(x), ..., h(x), the produced output sequences τ_A, τ_B, τ_C, ..., τ_H have linear complexities greater or equal to L'_A, L'_B, L'_C, ..., L'_H, where these numbers are related to the linear complexities L_A, L_B, L_C, ..., L_H of the shift registers A, B, C, ..., H as follows: $L'_A = L_A - 2$, $L'_B = L_B$, $L'_C = L_C - 5$, $L'_D = L_D - 2$, $L'_E = L_E - 6$, $L'_F = L_F - 9$, $L'_G = L_G$, $L'_H = L_H$.*

Proof. By Definition 3, the linear complexity of NLFSR A, say, is equal to the linear complexity of any nonzero standard output sequence σ_A. In other words, $L_A = L(\sigma_A)$. The filter-output sequence τ_A is related to σ_A by $\tau_A = a(T)\sigma_A$, where $a \in \mathbb{F}_2[x]$ is the applied filter polynomial. The assertion now follows from Theorems 13 and 18. □

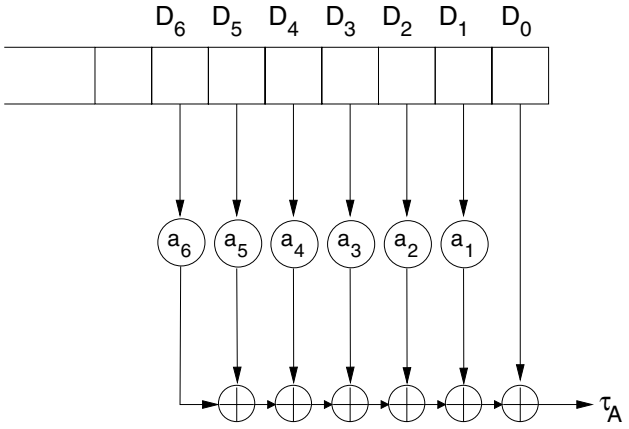


Figure 4: Linear feedforward function for NLFSR A: enlarged

Theorem 15 in Appendix A shows that the sequences $\tau_A, \tau_B, \tau_C, \ldots, \tau_H$ have good distribution properties.

Remark. If PROFILE 2-stream ciphers in ECRYPT's call for stream cipher primitives will be requested to handle 80 bit IV-lengths, the above mentioned restrictions on the degrees of the filter polynomials will be redefined in such a way that the sum of the maximum permissible degrees of all eight filter polynomials is 80. As a consequence, the KSG will possess 2^{80} different configurations and will be able to produce 2^{80} cyclically inequivalent periodic sequences.

2.4 Linear complexity and period of the keystream

Let τ_A, τ_B, τ_C, ..., τ_H be the output sequences of the eight driving NLFSR's. The sequences are combined by the Boolean combining function R to produce the keystream $\zeta = (z_t)_{t=0}^{\infty}$. The Boolean combining function is described in equation (1). The output sequences of the shift registers are assigned to the inputs of R according to the mapping:

Input variable	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8
NLFSR	A	C	D	E	B	G	H	F

Therefore,

$$\begin{aligned}\zeta = R(\tau_A, \tau_C, \tau_D, \tau_E, \tau_B, \tau_G, \tau_H, \tau_F) = \tau_A + \tau_C + \tau_D + \tau_E + \tau_B\tau_H \\ + \tau_G\tau_H + \tau_F\tau_G + \tau_B\tau_G\tau_H + \tau_F\tau_G\tau_H.\end{aligned} \tag{3}$$

The objective is to derive lower bounds for the period and linear complexity of ζ. We give some hints to the proof and will then state the results. Theorems 18 and 20 provide information on the minimal polynomials of τ_A, τ_B, τ_C, ..., τ_H. Theorem 21 yields the minimal polynomials of the product sequences appearing in (3). Theorem 12 now yields information on the minimal polynomial m_ζ of ζ. The linear complexity of ζ is assessed using Corollary 2, $L(\zeta) = \deg(m_\zeta)$, and Theorem 16. The period of ζ is determined using Corollary 2, $\mathrm{per}(\zeta) = \mathrm{ord}(m_\zeta)$, Theorem 17, and Lemma 1. We summarize the results.

Theorem 3. *The canonical factorization of the minimal polynomial of the keystream ζ consists only of irreducible binary polynomials of degrees* 2, 3, 5, 9, 11, 13, 22, 25, 26, 27, 58, 116, 203, 406, 713, 812, 899, 1798, 3596, 6293, 12586, 20677, *and* 25172. *All irreducible polynomials have multiplicity one.*

Theorem 4. *For all 2^{64} configurations of the output function of the KSG corresponding to all possible combinations of the filter polynomials $a(x)$, $b(x)$, $c(x)$, ..., $h(x)$, and*

for all initializations of the eight NLFSR's with nonzero initial state vectors, the linear complexity $L(\zeta)$ of the produced keystream ζ satisfies

$$\begin{aligned} L(\zeta) \geq\ & L'_A + L'_C + L'_D + L'_E + L'_B L'_H + L'_G L'_H \\ & + L'_F L'_G + L'_B L'_G L'_H + L'_F L'_G L'_H - 4, \end{aligned} \tag{4}$$

where the primed numbers are related to the linear complexities L_A, L_B, L_C, ..., L_H of the underlying NLFSR's by the equations $L'_A = L_A - 2 = 2^{22} - 15$, $L'_B = L_B = 2^{23} - 2$, $L'_C = L_C - 5$, $L'_D = L_D - 2$, $L'_E = L_E - 6$, $L'_F = L_F - 9$, $L'_G = L_G$, and $L'_H = L_H$. Under the assumption made in Section 2.2 we obtain

$$L(\zeta) \geq 2^{85}. \tag{5}$$

Theorem 5. *For all 2^{64} configurations of the output function of the KSG corresponding to all possible combinations of the filter polynomials $a(x)$, $b(x)$, $c(x)$, ..., $h(x)$, and for all initializations of the eight NLFSR's with nonzero initial state vectors, the produced keystream ζ has period*

$$\begin{aligned} \mathrm{per}(\zeta) = \frac{1}{9} & \left(2^{22}-1\right)\left(2^{23}-1\right)\left(2^{25}-1\right)\left(2^{26}-1\right) \\ & \cdot \left(2^{27}-1\right)\left(2^{28}-1\right)\left(2^{29}-1\right)\left(2^{31}-1\right). \end{aligned} \tag{6}$$

This implies that

$$\mathrm{per}(\zeta) \geq 2^{207}. \tag{7}$$

Remark. In the case that our stream cipher will have to be adapted to meet the 80 bit *IV*-size request, the KSG of the adapted stream cipher will then possess 2^{80} different configurations for its output function rather than 2^{64}. The formulas (4), (5), (6), and (7) will still hold. Only the relationships between the linear complexities L_A, L_B, L_C, ..., L_H, and L'_A, L'_B, L'_C, ..., L'_H may change slightly. For instance, relation $L'_E = L_E - 6$ must then be replaced by $L'_E = L_E - 9$.

3 The key-loading algorithm

The length of the secret key K is 80 bits. The bit length l of the initial value can be any number in $\{0, 8, 16, 24, 32, 40, 48, 56, 64\}$, where $l = 0$ means that no resynchronisations will be performed. We first concatenate the key K and the initial value IV to obtain the *interim key* $\mathbf{u}_r = (K, IV) = (k_0, k_1, \ldots, k_{79}, i_0, i_1, \ldots, i_{l-1}) = (u_0, u_1, \ldots, u_{r-1})$. The length r of the interim key is $r = 80 + l$, and can thus take on any value in $\{80, 88, 96, 104, 112, 120, 128, 136, 144\}$. The key-loading algorithm consists of several steps.

Step 1. (Load in parallel.) Load the first bits of the interim key $\mathbf{u}_r = (u_0, u_1, \ldots, u_{r-1})$ into the eight NLFSR's A, B, C, $\ldots$, H, and into the configuration register V. If the relevant shift register has length N, it receives the N bits $u_0, u_1, \ldots, u_{N-1}$. Cell D_j will contain the element u_j for $0 \leq j \leq N-1$. The loading of the shift register cells is performed in parallel. All 211 cells of the eight driving shift registers and the 64 cells of the configuration register are loaded simultaneously in order to avoid the leakage of side channel information.

Step 2. (Load in serially.) Feed-in into each FSR all bits of the interim key $\mathbf{u}_r = (u_0, u_1, \ldots, u_{r-1})$ that have not already been loaded into the register in Step 1. If the regarded shift register has length N, the bits $u_N, u_{N+1}, \ldots, u_{r-1}$ are fed into the register in serial. At each clock pulse, the feedback value and the current element u_j are added, and the obtained value is fed into cell D_{N-1} of the shift register.

Step 3. (Set the content of D_0 to 1.) In each of the eight FSR's A, B, C, $\ldots$, H, overwrite the content of cell D_0 with the bit 1. This operation makes sure that no driving NLFSR will be set to the all-zero state. For the configuration register V, the all-zero state is as good as all other states. Thus the element in cell D_0 of the register V is not overwritten.

Step 4. (Warm-up.) Each of the shift registers A, B, C, $\ldots$, H performs $N + 32$ shifts, where N is the length of the shift register. For instance, NLFSR A performs 54 warm-up shifts. The longest shift register, NLFSR H, performs 63 warm-up shifts. The given number of warm-up shifts for each shift register has the consequence that all eight driving NLFSR's achieve their final states simultaneously. The configuration register V performs 48 warm-up shifts.

The final state of the register V defines the configuration of the linear feedforward output function for each driving NLFSR. Suppose the cells $D_0, D_1, D_2, \ldots, D_{62}, D_{63}$ of the register V contain the bits $a_1, a_2, a_3, \ldots, h_9, h_{10}$, then the filter polynomials defining the configurations of the linear feedforward functions, are given by

$$
\begin{aligned}
a(x) &= a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + 1,\\
b(x) &= b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + 1,\\
c(x) &= c_7x^7 + c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + 1,\\
d(x) &= d_8x^8 + d_7x^7 + d_6x^6 + d_5x^5 + d_4x^4 + d_3x^3 + d_2x^2 + d_1x + 1,\\
e(x) &= e_8x^8 + e_7x^7 + e_6x^6 + e_5x^5 + e_4x^4 + e_3x^3 + e_2x^2 + e_1x + 1,\\
f(x) &= f_9x^9 + f_8x^8 + f_7x^7 + f_6x^6 + f_5x^5 + f_4x^4 + f_3x^3 + f_2x^2 + f_1x + 1,\\
g(x) &= g_9x^9 + g_8x^8 + g_7x^7 + g_6x^6 + g_5x^5 + g_4x^4 + g_3x^3 + g_2x^2 + g_1x + 1,\\
h(x) &= h_{10}x^{10} + h_9x^9 + h_8x^8 + h_7x^7 + h_6x^6 + h_5x^5 + h_4x^4 + h_3x^3 + h_2x^2 + h_1x + 1.
\end{aligned}
$$

Using Lemma 1 of Rueppel, Lai, and Woollven [20], and the fact that all NLFSR's are nonsingular, the following two theorems can be proved.

Theorem 6. *Consider any one of the eight driving NLFSR's of the KSG. Let N be the length of the shift register. Then there are exactly 2^{N-1} different states into which the shift register can be loaded as the result of the described key-loading algorithm. If all input vectors $\mathbf{u}_r = (u_0, u_1, \ldots, u_{r-1})$ are equally likely, then each possible state of the shift register is attained with the same probability.*

Theorem 7. *By applying the described key-loading algorithm, the NLFSR V can be loaded into any of its 2^{64} different states. If all input vectors $\mathbf{u}_r = (u_0, u_1, \ldots, u_{r-1})$ are equally likely, then each possible state of the shift register is attained with the same probability. In other words, any of the 2^{64} translation distinct sequences that the KSG is able to produce has the same chance to be selected.*

Remark. The described key-loading algorithm can easily be adapted to handle 80 bit IV-lengths.

4 Design rationale

Various attacks are known on stream ciphers based on linear feedback shift registers (LFSR's). We mention fast correlation attacks of Meier and Staffelbach [25], algebraic attacks of Courtois and Meier [7], and fault analysis attacks of Hoch and Shamir [15]. In a recent talk, Canteaut [6] gave a state of the art overview on algebraic attacks on LFSR based stream ciphers. One reason for choosing nonlinear feedback shift registers as building blocks in our stream cipher was to avoid such attacks.

Here are some remarks concerning the deployed NLFSR's. See Appendix C. Note that the NLFSR's are relatively sparse, considering that the algebraic normal form of the feedback function of a nonsingular binary N-stage NLFSR can contain 2^{N-1} nonzero terms. However, sparser primitive binary NLFSR's do exist. Note that each feedback function in Appendix C contains not only quadratic terms but also at least one cubic term and one term of order 4. The higher order terms increase the complexity of an algebraic attack.

As mentioned earlier, the Boolean combining function R is 4th-order correlation immune. Therefore, an attacker must select at least five NLFSR's in order to run a classical correlation attack. The sum of the lengths of the smallest five NLFSR's is 123. Taking into account that—due to Step 3 in the key-loading algorithm—for each shift register half of all possible initial states can be ruled out, there remain 2^{118} initial settings for the five NLFSR's. For each setting a bit string of some length must be computed and compared with the actual keystream. Obviously the attack is more complex than exhaustive key search.

One might argue that we provided an unnecessary high security margin. Certainly, a similar stream cipher based on seven NLFSR's of comparable lengths with a suitable 7-variable, 3rd-order correlation immune Boolean combining function would still push the above correlation attack beyond the complexity of exhaustive key search. One reason

that we chose eight NLFSR's is that the additional NLFSR will add to the statistical quality of the produced keystream. At any rate high complexity distinguishing attacks become a growing threat to today's stream cipher systems, so that the extra shift register seems to be a good investment. We subjected the output sequence of the KSG to 164 statistical tests [21] which were all passed. The most stringent 60 tests evaluated approximately 2^{43} bits.

With regard to the potential leakage of side channel information the key-loading algorithm is the most sensitive part in a stream cipher. The goal of Step 1 in the above described key-loading algorithm is to prevent the leakage of side channel information during key-setup. Another advantage of the parallel loading in Step 1 is the reduction of the resynchronization time.

The algebraic degree of the Boolean combining function is 3. The algebraic degree 3 was chosen to guarantee that the linear complexity of the keystream is larger than $2^{\text{key length}}$.

We hereby state that we are not aware of any hidden weaknesses of the proposed stream cipher. Furthermore, the stream cipher does not seem to have any weak keys.

5 The reduced keystream generator

In a reduced form of the KSG, the ability of changing the configuration of the output function of the KSG after each resynchronisation step has been dropped. No linear feedforward logics are implemented in the reduced KSG. Instead the standard output function for each shift register is used. In the standard output function at each clock pulse, the content of cell D_0 is forwarded to the Boolean combining function R. In order to distinguish it from the reduced KSG, we call the KSG discussed so far the *full-fledged KSG*. Note that the standard output function of the reduced KSG corresponds to the special configuration of the linear feedforward functions in the full-fledged KSG in which all eight filter polynomials are equal to 1. That is, $a(x) = 1$, $b(x) = 1$, $c(x) = 1, \ldots, h(x) = 1$.

While the full-fledged KSG can produce 2^{64} (respectively 2^{80}) cyclically inequivalent sequences of period larger than 2^{207}, the reduced KSG can produce only one sequence of period larger than 2^{207}. In the full-fledged KSG, the secrecy lies both in the particular sequence that has been chosen from the ensemble and in the initial phase (the starting point) of that sequence. In the reduced KSG, the secrecy lies solely in the unknown initial phase. In chapter 7 of his dissertation [17], Jansen introduces the concepts of *phase uncertainty profile* and *sequence uncertainty profile*. The latter concept can only be applied to stream ciphers which are capable to produce an ensemble of different sequences. The conclusions that Jansen draws after investigating the behavior of the mentioned uncertainty profiles were the main reason for us to advocate a stream cipher based on the full-fledged KSG.

However, the stream cipher variant based on the reduced KSG should also be regarded as an option. In the corresponding reference implementations, we found that

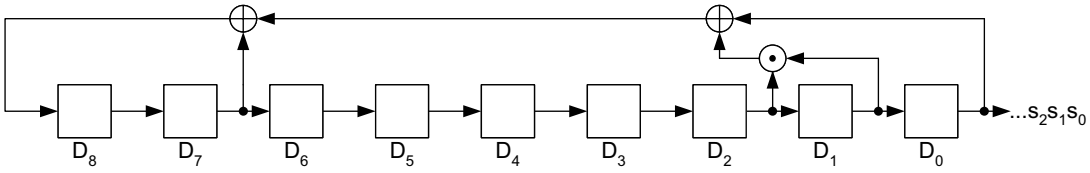
the design size of the full-fledged KSG corresponds to 2988 gate equivalents, whereas the design size of the reduced KSG amounts to 2173 gate equivalents. Thus the reduced KSG is in particular of interest if gate count and power consumption become determining factors. Despite the superiority of the full-fledged KSG over the reduced KSG from an information theoretical point of view, it might very well be the case that the stream cipher using the reduced KSG will also be secure against known attacks, insofar that the computational complexity of any such attack might be larger than exhaustive key search.

6 Parallel implementation

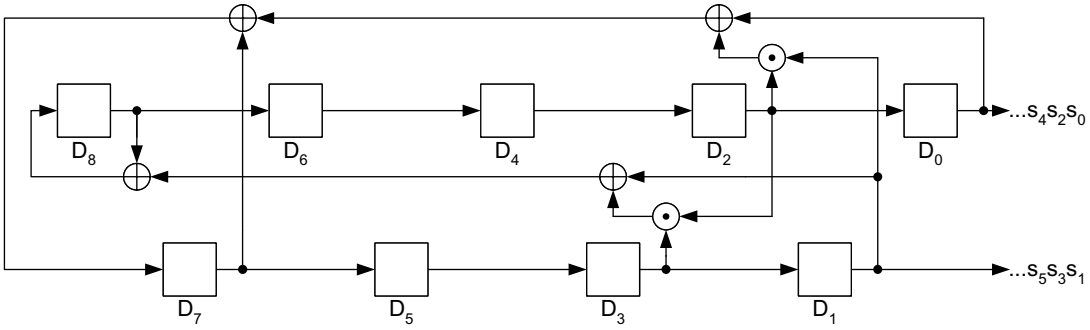
The speed of keystream generation and thus the encryption rate of the stream cipher can be increased by choosing accelerated hardware implementations for the eight driving NLFSR's. Instead of shifting the content of one flip-flop to the next flip-flop in line, in an accelerated implementation of the FSR, a few intermediate flip-flops are skipped. In order that the accelerated shift register can accommodate all flip-flops with the correct input at each clock pulse, the feedback logic must be duplicated. The number of flip-flops remains the same. The concept of increasing the performance of a FSR, by taking larger step sizes, is best explained on an example. Consider the 9-stage NLFSR defined by the feedback function

$$F(x_0, x_1, \ldots, x_8) = x_0 + x_7 + x_1x_2.$$

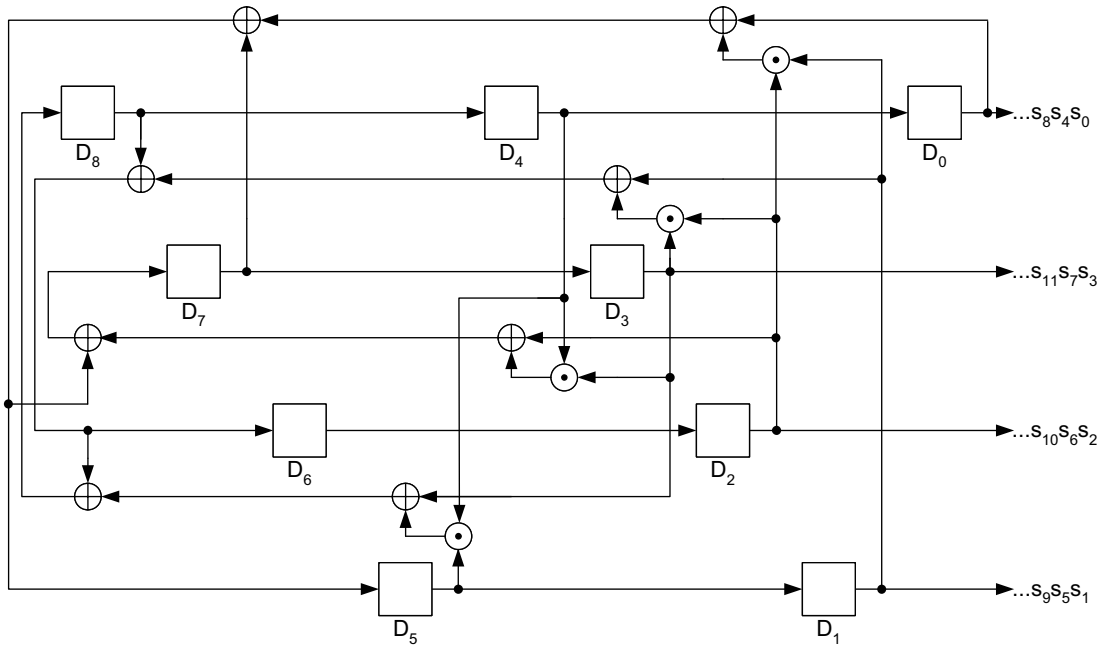
The regular implementation of this FSR is depicted in the following figure:



If we choose step size $k = 2$, the corresponding *fast* implementation looks like this:



The next figure shows the implementation of the FSR for step size $k = 4$.



One leading point in the selection process of the deployed NLFSR's was their ability to facilitate fast hardware implementations up to a factor of eight. This is the reason that in all feedback functions the last seven variables appear only linearly.

In the high-speed implemention of the KSG not only the encryption rate is increased. The key-loading algorithm is also accelerated. See the table in Section 7.

7 Hardware considerations

In Table 1, the Achterbahn stream cipher is compared with the stream ciphers E0, A5/1, and RC4 and with three different implementations of the Advanced Encryption Standard (AES). The design size (given in gate equivalents, GE), throughput, frequency, efficiency, maximum throughput, maximum frequency, crytical path, and resynchronisation times are listed. Four different implementations of the stream cipher Achterbahn are discussed. The straightforward implementation, where one bit of keystream is generated per clock cycle, is abbreviated as *serial*. The terms *2-bit-parallel*, *4-bit-parallel*, and *8-bit-parallel* refer to the accelerated implementations, in which the underlying FSR's operate with step sizes $k = 2$, 4, and 8, respectively. The numbers in the parenthesis refer to the reduced KSG. For more background on the mentioned figures of merit, we refer to Appendix B.

	Achterbahn				**AES**			**E0**	**A5/1**	**RC4**
	1	2	4	8	minimal	small	high speed			
reference					[10]	[32, 26]		[18]	[2]	[19]
key/IV size [*bit*]	80/0,8,...,64				128/128			128/74	64/22	8,...,128[e]
state size [*bit*]	275 (211 for reduced version)[a]				128			132	64	2064
word size [*bit*]	1	2	4	8	128			1	1	8
notes	serial	2 bit parallel	4 bit parallel	8 bit parallel	1 S-box 1016 cyc/enc	4 S-boxes 54 cyc/enc	20 S-boxes 11 cyc/enc			3 cyc/byte
resync. time [*cyc*]	$112 + l(IV)$	$56 + \frac{l(IV)}{2}$	$28 + \frac{l(IV)}{4}$	$14 + \frac{l(IV)}{8}$	1016	54	11	239	188	768[f]
design size [*GE*]	2988 (2173)	3427 (2412)	4633 (3113)	7547 (4778)	4563[b]	6966[c]	26105[d]	1902	932	12952
throughput $\langle \mathcal{N} \rangle$ [*bit/cyc*]	1	2	4	8	0.13	2.37	11.64	1	1	2.66
efficiency $\mathcal{E}$ [*bit/cyc/kGE*]	0.33 (0.46)	0.58 (0.83)	0.86 (1.28)	1.06 (1.67)	0.028	0.34	0.45	0.53	1.07	0.21
critical path[g] [*#gate delays*]	8	8	8	8	nn	50[h]	35[k]	2	3	nn
max. frequency	>1 GHz[m]	>1 GHz[m]	>1 GHz[m]	>1 GHz[m]	100 kHz	95 MHz[n]	130 MHz[p]	>3 GHz	>2 GHz	nn
max. throughput	>1 Gbps[m]	>2 Gbps[m]	>4 Gbps[m]	>8 Gbps[m]	12.5 kbps	220 Mbps[n]	1.5 Gbps[p]	>3 Gbps	>3 Gbps	nn

nn: not known, cyc: clock cycle, kGE: 1000 gate equivalents, $l(IV)$: length of IV,
[a]all values for Achterbahn given in parenthesis refer to the reduced keystream generator, see Sec. 5,
[b]we added to the reported size of 3595 GE the 128 bit state with 768 GE and 200 GE for DPA protection (masking, see [35]),
[c]we added to the reported size of 5398 GE the 128 bit state with 768 GE and 800 GE for DPA protection (masking, see [35]),
[d]we added to the reported size of 21337 GE the 128 bit state with 768 GE and 4000 GE for DPA protection (masking, see [35]),
[e]the 8-bit word implementation of RC4 allows key+IV sizes from 8 to 2048 bits using various resynchronization schemes,
[f]following the recommendations of dropping the first 512 bytes [13] will increase the setup time to 2304 cycles,
[g]the critical path is calculated assuming the use of multi-input gates,
[h]we estimate a critical path of approx. 35 gate delays and add 15 gate delays for DPA protection (masking, see [35]),
[k]we estimate a critical path of approx. 20 gate delays and add 15 gate delays for DPA protection (masking, see [35]),
[m]for a 0.13μ CMOS process (with a pipelined implementation the throughput can be further increased by a factor of 2 to 4),
[n]estimated reduction of the reported throughput of 311Mbps at 131MHz (for a 0.11μm CMOS process) due to DPA counter measures,
[p]estimated reduction of the reported throughput of 2.6Gbps at 224MHz (for a 0.11μm CMOS process) due to DPA counter measures.

Table 1: Comparison of figures for the hardware implementation of several stream ciphers.

8 Conclusion

We proposed a new synchronous stream cipher called *Achterbahn*. The core of the keystream generator consists of eight primitive binary nonlinear feedback shift registers. Each shift register is endowed with a configurable linear feedforward output function. The produced output sequences are combined by a balanced, 8-variable, 4th order correlation immune Boolean combining function of algebraic degree 3. In the key-loading algorithm, the initial states of the nonlinear feedback shift registers are determined and, for each shift register, the configuration of its output function is defined. Due to the modifications of the output functions, the keystream generator is able to produce an ensemble of 2^{64} translation distinct binary sequences. Each sequence has period larger than 2^{207} and linear complexity larger than 2^{85}. The key-loading algorithm receives as inputs the 80 bit secret key and a public initial value of up to 64 bits.

Based on parallel implementations of the underlying feedback shift registers, the encryption speed of the stream cipher can be scaled by any positive integer factor less than the length of the shortest feedback shift register. The feedback functions of the applied shift registers specifically promote parallel implementations up to a factor of eight. If the factor eight is chosen, the keystream generator will produce one byte of keystream per clock cycle.

The potential to encrypt the eight lines (acht Bahnen) of an 8-bit bus in real time was one reason for choosing the name Achterbahn. Another cause for the name is the fact that eight feedback shift registers are the driving force in the keystream generator. Furthermore, the number eight is reflected in the size of the secret key. There is yet another more subtle motivation for the name. Achterbahn is the german word for roller coaster. If people ride a state of the art roller coaster, some become addicted to it while others get sick. We hope that the same will happen to the cryptographer who studies our stream cipher and to the cryptanalyst who aims to break it.

Appendix A

A Mathematical Background

Let $\mathbb{F}_q$ be the finite field of order q. The set of all sequences of elements of $\mathbb{F}_q$ is denoted by $\mathbb{F}_q^\infty$. If we define for $\sigma = (s_n)_{n=0}^\infty \in \mathbb{F}_q^\infty$ and $\tau = (t_n)_{n=0}^\infty \in \mathbb{F}_q^\infty$ and for $c \in \mathbb{F}_q$ the *sum* $\sigma + \tau = (s_n + t_n)_{n=0}^\infty$ and the *scalar product* $c\sigma = (cs_n)_{n=0}^\infty$, then $\mathbf{F}_q^\infty$ becomes a vector space over $\mathbb{F}_q$. An important linear operator on the vector space $\mathbf{F}_q^\infty$ is the shift operator T, defined by $T\sigma = (s_{n+1})_{n=0}^\infty$ for all sequences $\sigma = (s_n)_{n=0}^\infty \in \mathbb{F}_q^\infty$.

A sequence $\sigma = (s_n)_{n=0}^\infty$ in $\mathbb{F}_q^\infty$ is called *ultimately periodic* if there are integers $n_0 \geq 0$ and $p \geq 1$ such that $s_{n+p} = s_n$ for all $n \geq n_0$. The smallest such integers n_0 and p are called the *preperiod* and the *period* of σ, respectively. We then write $\mathrm{per}(\sigma) = p$. If $s_{n+p} = s_n$ for all $n \geq 0$, then the sequence is called *periodic*. Note that the expression *ultimately periodic* allows the possibility that the sequence is actually periodic.

Any ultimately periodic sequence σ of $\mathbb{F}_q^\infty$ possesses a unique polynomial $m_\sigma \in \mathbb{F}_q[x]$, called the *minimal polynomial* of σ. There are various approaches to the minimal polynomial, one uses ideal theory. If $g \in \mathbb{F}_q[x]$ is a polynomial over $\mathbb{F}_q$, then $g(T)$ defines a linear operator on the vector space $\mathbb{F}_q^\infty$. For instance, let $g(x) = x^3 + x + 1$. Then $g(T)\sigma = T^3\sigma + T\sigma + \sigma = (s_{n+3} + s_{n+1} + s_n)_{n=0}^\infty$ for all $\sigma = (s_n)_{n=0}^\infty$ of $\mathbb{F}_q^\infty$. We say that a polynomial $g \in \mathbb{F}_q[x]$ *annulates* σ, if $g(T)\sigma$ is the zero sequence $\mathbf{0} = (0, 0, \dots)$. For instance, if $\sigma = (s_n)_{n=0}^\infty \in \mathbb{F}_q^\infty$ is ultimately periodic such that $s_{n+p} = s_n$ for all $n \geq n_0$, then $g(x) = x^{n_0+p} - x^{n_0} \in \mathbb{F}_q[x]$ annulates σ. Thus, for every ultimately periodic sequence $\sigma \in \mathbb{F}_q^\infty$,

$$J_\sigma = \{g \in \mathbb{F}_q[x] : g(T)\sigma = \mathbf{0}\}$$

is a nonzero ideal in the principal ideal domain $\mathbb{F}_q[x]$. The minimal polynomial m_σ of σ is the unique monic polynomial over $\mathbb{F}_q$ generating J_σ, that is,

$$J_\sigma = (m_\sigma) = \{hm_\sigma : h \in \mathbb{F}_q[x]\}.$$

Theorem 8. *Let σ be an ultimately periodic sequence in $\mathbb{F}_q^\infty$. Any polynomial $g \in \mathbb{F}_q[x]$ that annulates σ is called a* characteristic polynomial *of σ. A polynomial $g \in \mathbb{F}_q[x]$ is a characteristic polynomial of σ if and only if m_σ divides g.*

Proof. The assertion follows directly from the fact that the minimal polynomial of σ generates the ideal J_σ and from the definition of the characteristic polynomial of σ. □

Corollary 1. *Let σ be a periodic sequence in $\mathbb{F}_q^\infty$ with minimal polynomial $m_\sigma \in \mathbb{F}_q[x]$. Let $c \in \mathbb{F}_q[x]$ be a characteristic polynomial of σ without multiple roots, and let $f \in \mathbb{F}_q[x]$ be an irreducible factor of c. Then f divides m_σ if and only if the polynomial $g = c/f$ is not a characteristic polynomial of σ.*

Proof. Since the minimal polynomial divides any characteristic polynomial we have $c = bm_\sigma$ for some $b \in \mathbb{F}_q[x]$. Clearly, $g = c/f$ is a multiple of m_σ if and only if f divides b. Thus g is not a multiple of m_σ if and only if f divides m_σ. □

The minimal polynomial m_σ of an ultimately periodic sequence $\sigma \in \mathbb{F}_q^\infty$ contains a lot of information about σ.

1. The multiplicity of the element 0 as a root of m_σ coincides with the preperiod of σ. In particular, σ is periodic if and only if $m_\sigma(0) \neq 0$.
2. The polynomial m_σ is the characteristic polynomial of the shortest linear feedback shift register that can generate σ when appropriately initialized.
3. By definition, the linear complexity of σ is equal to the degree of m_σ.
4. The order of the polynomial m_σ coincides with the period of σ.

We restate the last property as a theorem.

Theorem 9. *Let σ be an ultimately periodic sequence in $\mathbb{F}_q^\infty$ with minimal polynomial $m_\sigma \in \mathbb{F}_q[x]$. Then the period of σ is equal to the order of the minimal polynomial of σ, denoted by* $\operatorname{ord}(m_\sigma)$.

Proof. See Lidl and Niederreiter [22, Theorem 8.44]. □

The order of a polynomial f is sometimes also called the *period* of f or the *exponent* of f. We quote another theorem from [22] concerning the order of a polynomial.

Theorem 10. *Let $g_1, \ldots, g_k$ be pairwise relatively prime nonzero polynomials over $\mathbb{F}_q$, and let $f = g_1 \cdots g_k$. Then*

$$\operatorname{ord}(f) = \operatorname{lcm}\left(\operatorname{ord}(g_1), \ldots, \operatorname{ord}(g_k)\right).$$

Proof. See Lidl and Niederreiter [22, Theorem 3.9]. □

Another interesting approach to the minimal polynomial of an ultimately periodic sequence $\sigma \in \mathbb{F}_q^\infty$ makes use of generating functions. Following Niederreiter [28], we assign to an arbitrary sequence $\sigma = (s_n)_{n=0}^\infty$ of elements of $\mathbb{F}_q$ the generating function

$$G_\sigma(x) = s_0 x^{-1} + s_1 x^{-2} + s_2 x^{-3} + \cdots,$$

regarded as an element of the field $\mathbb{F}_q((x^{-1}))$ of formal Laurent series in the indeterminate x^{-1}. The field $\mathbb{F}_q((x^{-1}))$ contains the field $\mathbb{F}_q(x)$ of rational functions as a subfield. A sequence $\sigma \in \mathbb{F}_q^\infty$ is ultimately periodic if and only if the associated generating function G_σ belongs to the subfield $\mathbb{F}_q(x)$.

Theorem 11. *Let $m \in \mathbb{F}_q[x]$ be a monic polynomial, and let $\sigma = (s_n)_{n=0}^{\infty}$ be a sequence of elements of $\mathbb{F}_q$. Then σ is ultimately periodic and m is the minimal polynomial of σ if and only if*

$$\sum_{n=0}^{\infty} s_n x^{-n-1} = \frac{h(x)}{m(x)}$$

with a polynomial $h \in \mathbb{F}_q[x]$ with $\deg(h) < \deg(m)$ and $\gcd(h, m) = 1$.

Proof. See Niederreiter [28], [22, p. 218]. □

Theorem 12. *For each $j = 1, \ldots, k$, let σ_j be an ultimately periodic sequence in $\mathbb{F}_q^{\infty}$ with minimal polynomial $m_j \in \mathbb{F}_q[x]$. If the polynomials $m_1, \ldots, m_k$ are pairwise relatively prime, then the minimal polynomial of the sum $\sigma = \sigma_1 + \cdots + \sigma_k$ is equal to the product $m_1 \cdots m_k$.*

Conversely, let σ be an ultimately periodic sequence in $\mathbb{F}_q^{\infty}$ whose minimal polynomial $m \in \mathbb{F}_q[x]$ is the product of pairwise relatively prime monic polynomials $m_1, \ldots, m_k \in \mathbb{F}_q[x]$. Then, for each $j = 1, \ldots, k$, there exists a uniquely determined ultimately periodic sequence σ_j with minimal polynomial $m_j \in \mathbb{F}_q[x]$ such that $\sigma = \sigma_1 + \cdots + \sigma_k$.

Proof. A proof of the first part of the theorem can be found on page 426 in [22]. To prove the second part, let $h/m \in \mathbb{F}_q(x)$ be the generating function of σ in the sense of Theorem 11. Let

$$\frac{h}{m} = \frac{h_1}{m_1} + \cdots + \frac{h_k}{m_k} \tag{8}$$

be the partial fraction decomposition of h/m. By Theorem 11, $\deg(h) < \deg(m)$ and $\gcd(h, m) = 1$. This implies $\deg(h_j) < \deg(m_j)$ and $\gcd(h_j, m_j) = 1$ for $1 \leq j \leq k$. The rational functions h_j/m_j correspond to uniquely determined ultimately periodic sequences $\sigma_j \in \mathbb{F}_q^{\infty}$ with minimal polynomials m_j according to Theorem 11. Equation (8) implies that $\sigma = \sigma_1 + \cdots + \sigma_k$. □

Let σ be an ultimately periodic sequence of $\mathbb{F}_q^{\infty}$, and let f be a nonzero polynomial over $\mathbb{F}_q$. We call the sequence $\tau = f(T)\sigma$ a *linearly filtered* sequence derived from σ. The polynomial f is called the *filter polynomial*. Note that the sequence τ is a linear combination of shifted versions of σ.

Theorem 13. *Let σ be an ultimately periodic sequence of elements of $\mathbb{F}_q$ with minimal polynomial $m_\sigma \in \mathbb{F}_q[x]$, and let f be a nonzero polynomial over $\mathbb{F}_q$. Then the sequence $\tau = f(T)\sigma$ is again ultimately periodic and has minimal polynomial*

$$m_\tau = \frac{m_\sigma}{\gcd(m_\sigma, f)}.$$

Proof. See Niederreiter [27], or Blackburn [3], or Göttfert [14, Chap. 2]. □

If $\sigma \in \mathbb{F}_q^\infty$ is periodic and $f \in \mathbb{F}_q[x]$ is arbitrary, then $\tau = f(T)\sigma$ is also periodic. This is trivial if f is the zero polynomial. Otherwise, recall that σ is periodic if and only if $m_\sigma(0) \neq 0$. Since m_τ, the minimal polynomial of τ, divides m_σ, by Theorem 13, we have $m_\tau(0) \neq 0$, so that τ is periodic.

If $\sigma = (s_n)_{n=0}^\infty$ is a periodic sequence in $\mathbb{F}_q^\infty$ with $\mathrm{per}(\sigma) = p$, then the least positive integer N such that the N-tuples $\mathbf{s}_n = (s_n, s_{n+1}, \ldots, s_{n+N-1})$, $0 \leq n \leq p-1$, are distinct is called the *span* or the *maximum order complexity* of σ. Equivalently, the periodic sequence $\sigma \in \mathbb{F}_q^\infty$ has maximum order complexity N (or span N) if σ could be generated by some feedback shift register over $\mathbb{F}_q$ of length N but by no shorter feedback shift register.

An N-stage feedback shift register (FSR) over $\mathbb{F}_q$ is uniquely determined by its feedback function $F : \mathbb{F}_q^N \to \mathbb{F}_q$. The FSR is called *nonsingular* if the mapping

$$\Phi : (y_{N-1}, \ldots, y_1, y_0) \in \mathbb{F}_q^N \mapsto (F(y_0, y_1, \ldots, y_{N-1}), y_{N-1}, \ldots, y_2, y_1) \in \mathbb{F}_q^N$$

is a bijection. If the feedback function F of an N-stage FSR is linear, it is called a *linear feedback shift register* (LFSR). Otherwise it is referred to as a *nonlinear feedback shift register* (NLFSR).

Definition 1. *Let $F : \mathbb{F}_q^N \to \mathbb{F}_q$ be the feedback function of an N-stage feedback shift register. The FSR is called* primitive *if for any nonzero initial state vector of $\mathbb{F}_q^N$ the corresponding standard output sequence of the FSR has period $q^N - 1$, and if $F(\mathbf{0}) = 0$, where $\mathbf{0}$ is the zero vector of $\mathbb{F}_q^N$.*

Note that a primitive feedback shift register over $\mathbb{F}_q$ is necessarily nonsingular.

If the feedback function $F : \mathbb{F}_q^N \to \mathbb{F}_q$ of an N-stage FSR is linear, that is, if

$$F(x_0, x_1, \ldots, x_{N-1}) = a_0 x_0 + a_1 x_1 + \cdots + a_{N-1} x_{N-1}$$

with $a_j \in \mathbb{F}_q$ for $0 \leq j \leq N-1$, then the N-degree polynomial $c \in \mathbb{F}_q[x]$ given by

$$c(x) = x^N + F(1, x, x^2, \ldots, x^{N-1}) = x^N + a_{N-1}x^{N-1} + \cdots + a_1 x + a_0$$

is called the *characteristic polynomial* of the LFSR. It is well known (see e.g. Lidl and Niederreiter [22, Chap. 8]) that an N-stage LFSR whose characteristic polynomial is a primitive polynomial over $\mathbb{F}_q$, will generate a periodic sequence of period $q^N - 1$ for any nonzero initial state vector. Thus an LFSR with a primitive characteristic polynomial is a primitive FSR in the sense of Definition 1.

Example 1. Consider the binary 5-stage NLFSR with feedback function

$$F(x_0, x_1, x_2, x_3, x_4) = x_0 + x_1 + x_3 + x_1 x_3.$$

The feedback shift register is shown in Figure 5.

The standard output sequence corresponding to the initial state vector $\mathbf{s}_0 = (0, 0, 0, 0, 1)$ for the given feedback shift register is

$$\sigma = (0\,0\,0\,0\,1\,0\,1\,0\,1\,1\,1\,0\,1\,0\,0\,1\,1\,0\,1\,1\,0\,0\,1\,0\,0\,0\,1\,1\,1\,1\,1)^{\infty}.$$

The sequence σ has period $\mathrm{per}(\sigma) = 31$ and linear complexity $L(\sigma) = 30$. Note that the sequence $\sigma = (s_n)_{n=0}^{\infty}$ can be defined by the nonlinear recursion

$$s_{n+5} = s_{n+3}s_{n+1} + s_{n+3} + s_{n+1} + s_n \qquad \text{for all } n \geq 0, \tag{9}$$

together with the initial values $s_0 = s_1 = s_2 = s_3 = 0$, and $s_4 = 1$. One can say that the above feedback shift register provides the hardware implementation of recursion (9). □

There are $\varphi(2^N - 1)/N$ primitive binary polynomials of degree N. Therefore, there are $\varphi(2^N - 1)/N$ binary primitive N-stage LFSR's. The total number of binary primitive N-stage FSR's, linear or nonlinear, is given by

$$B_N = 2^{2^{N-1}-N}.$$

This is the number of translation distinct (see Definition 2 below) binary deBruijn sequences [5], and there is a one-to-one correspondence between binary deBruijn sequences and sequences produced by binary primitive FSR's. The number of nonsingular binary N-stage FSR's is given by

$$C_N = 2^{2^{N-1}}.$$

See Walker [36]. Comparing numbers B_N and C_N, we find $B_N/C_N = 1/2^N$. Thus, on the average one out of 2^N nonsingular binary FSR's is primitive.

Definition 2. *Let σ and τ be periodic sequences in $\mathbb{F}_q^{\infty}$ with $\mathrm{per}(\sigma) = \mathrm{per}(\tau) = p$. Then we call σ and τ* cyclically equivalent *if there is an integer j with $1 \leq j \leq p$ such that $\tau = T^j\sigma$. If no such integer j exists, we call σ and τ* cyclically inequivalent *or* translation distinct.

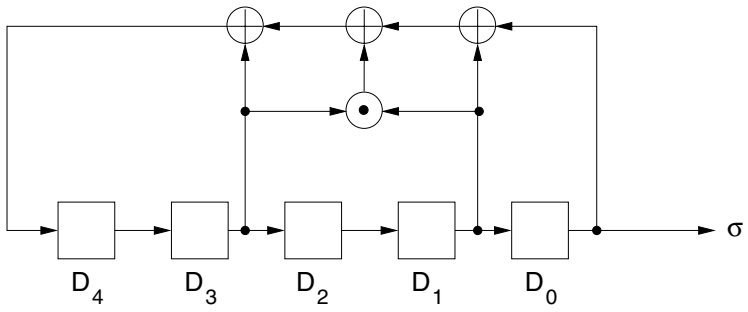


Figure 5: A primitive 5-stage NLFSR

Theorem 14. *Cyclically equivalent sequences of $\mathbb{F}_q^\infty$ have the same minimal polynomial.*

Proof. Let σ and τ be two cyclically equivalent sequences in $\mathbb{F}_q^\infty$ of period $p \geq 1$. Then there exists an integer j with $1 \leq j \leq p$ and $\tau = T^j\sigma$. By Theorem 13,

$$m_\tau(x) = \frac{m_\sigma(x)}{\gcd(m_\sigma(x), x^j)}.$$

Since σ is periodic, $m_\sigma(0) \neq 0$ so that $m_\sigma(x)$ and x^j are relatively prime. □

Any two standard output sequences of a primitive N-stage FSR over $\mathbb{F}_q$ corresponding to nonzero initial state vectors are cyclically equivalent. They are both shifted versions of the same periodic sequence of period $q^N - 1$. Therefore, the following definition makes sense.

Definition 3. *Let $F : \mathbb{F}_q^N \to \mathbb{F}_q$ be the feedback function of a primitive N-stage FSR over $\mathbb{F}_q$. Then we define the minimal polynomial, the period, and the linear complexity of the FSR to be the minimal polynomial, the period, and the linear complexity, respectively, of any nonzero standard output sequence of the FSR.*

Let $\sigma = (s_n)_{n=0}^\infty$ be any nonzero standard output sequence of a primitive N-stage feedback shift register. Equivalently, let σ be a periodic sequence in $\mathbb{F}_q^\infty$ of period $p = q^N - 1$ and of span N that does not contain N consecutive zero terms. We investigate the distribution properties of the linearly filtered sequence $\tau = f(T)\sigma$ in the case that the nonzero filter polynomial $f \in \mathbb{F}_q[x]$ has degree less than N. We will show that up to a slight aberration for the zero element, the elements of $\mathbb{F}_q$ are equidistributed in τ. Moreover, all possible strings of elements of $\mathbb{F}_q$ up to a certain length which depends on the length N of the primitive FSR and the degree of the filter polynomial, appear equally often within a full portion of the period of τ—again, up to a slight aberration for the all-zero string.

If $f(x) = x^e g(x)$ with $e \geq 0$ and $g \in \mathbb{F}_q[x]$ with $g(0) \neq 0$, then the sequence $f(T)\sigma$ is a shifted version of the sequence $g(T)\sigma$. Therefore, w.l.o.g. we can restrict our attention to filter polynomials f which are not divisible by x.

Theorem 15. *Let $\sigma = (s_n)_{n=0}^\infty$ be any nonzero output sequence of an N-stage primitive FSR over $\mathbb{F}_q$. Let $f \in \mathbb{F}_q[x]$ with $f(0) \neq 0$ and $0 \leq \deg(f) = k \leq N - 1$. Let $\tau = (t_n)_{n=0}^\infty = f(T)\sigma$. For $1 \leq m \leq N - k$ and $\mathbf{b} = (b_1, \ldots, b_m) \in \mathbb{F}_q^m$ let $Z(\mathbf{b})$ be the number of $n \in \{0, 1, \ldots, r-1\}$ for which $(t_n, t_{n+1}, \ldots, t_{n+m-1}) = \mathbf{b}$. Then*

$$Z(\mathbf{b}) = \begin{cases} q^{N-m} - 1 & \text{for } \boldsymbol{b} = \boldsymbol{0}, \\ q^{N-m} & \text{for } \boldsymbol{b} \neq \boldsymbol{0}. \end{cases}$$

Proof. By assumption, $f(x) = a_0 + a_1 x + \cdots + a_k x^k$ with $a_0 a_k \neq 0$. Thus

$$t_n = a_0 s_n + a_1 s_{n+1} + \cdots + a_k s_{n+k} \quad \text{for } n = 0, 1, \ldots.$$

Let $\mathbf{b} = (b_1, \ldots, b_m) \in \mathbb{F}_q^m$ be fix. Consider the system of m linear equations in N unknowns $x_0, x_1, \ldots, x_{N-1}$, given by

$$\sum_{j=0}^{k} a_j x_{j+h} = b_{h+1}, \qquad h = 0, 1, \ldots, m-1. \tag{10}$$

Let A be the matrix of coefficients of the corresponding homogeneous system of linear equations, so that

$$A = \begin{pmatrix} a_0 & a_1 & \ldots & a_k & 0 & 0 & 0 & \ldots & 0 \\ 0 & a_0 & a_1 & \ldots & a_k & 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \ddots & & \ddots & & & \vdots \\ 0 & 0 & \ldots & a_0 & a_1 & \ldots & a_k & \ldots & 0 \end{pmatrix}.$$

Then A is an $m \times N$ matrix over $\mathbb{F}_q$ of rank m, since $a_0 \neq 0$. If $\mathbf{b} \neq \mathbf{0}$ then the augmented matrix $A' = (A, \mathbf{b}^{\mathrm{t}})$, which is the $m \times (N+1)$ matrix whose first N columns are the columns of the matrix A and whose last column is the transpose of $\mathbf{b}$, has also rank m. Hence the system of linear equations in (10) has q^{N-m} distinct solution vectors $(x_0, \ldots, x_{N-1}) \in \mathbb{F}_q^N$.

If $\mathbf{b} = \mathbf{0}$, then the zero vector of $\mathbb{F}_q^N$ is one of the q^{N-m} solution vectors of the system (10). As n runs through $0, 1, \ldots, r-1$, all nonzero N-tuples occur among $\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_{r-1} \in \mathbb{F}_q^N$, so that $Z(\mathbf{0}) = q^{N-m} - 1$. If $\mathbf{b} \neq \mathbf{0}$, then all q^{N-m} solution vectors of (10) are nonzero and thus occur among $\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_{r-1}$, so that $Z(\mathbf{b}) = q^{N-m}$. □

Example 2. Consider the 5-stage NLFSR of Example 1. If we apply to the stages of the shift register a linear feedforward function, then the shift register will produce a new output sequence $\tau = (t_n)_{n=0}^{\infty}$. In contrast to the standard output sequence $\sigma = (s_n)_{n=0}^{\infty}$ which is obtained by emitting the content of cell D_0 at any clock pulse, the terms of sequence τ are obtained by outputting the contents of several cells and adding together the outputs. As an illustration, see Figure 6.

We have

$$t_n = s_n + s_{n+1} + s_{n+3} \qquad \text{for all } n \geq 0. \tag{11}$$

Using the shift operator T, equation (11) can be written as

$$\tau = \sigma + T\sigma + T^3\sigma = (1 + T + T^3)\sigma = f(T)\sigma,$$

and the filter polynomial is given by $f(x) = x^3 + x + 1$. □

While the application of the linear operator $f(T)$ to any nonzero standard output sequence σ of a primitive N-stage FSR over $\mathbb{F}_q$ preserves the distribution properties of σ largely, provided that $\deg(f)$ is small compared to N, the linearly filtered sequence $\tau = f(T)\sigma$ has in general a maximum order complexity twice as high as σ. For more information on linear filtering see [12].

We recall some results from Selmer [33, Chap. 4], and Zierler and Mills [37]. Let $f, g, \ldots, h \in \mathbb{F}_q[x]$ be nonconstant polynomials without multiple roots in their respective splitting fields over $\mathbb{F}_q$ and with nonzero constant terms. Then $f \vee g \vee \cdots \vee h$ is defined to be the monic polynomial whose roots are the distinct products $\alpha\beta\cdots\gamma$, where α is a root of f, β a root of g, and γ a root of h. The polynomial $f \vee g \vee \cdots \vee h$ is again a polynomial over the ground field $\mathbb{F}_q$. This follows from the fact that all conjugates (over $\mathbb{F}_q$) of a root of $f \vee g \vee \cdots \vee h$ are roots of $f \vee g \vee \cdots \vee h$. The importance of the polynomial $f \vee g \vee \cdots \vee h$ stems from the fact that if $\sigma = (s_n)_{n=0}^{\infty}$, $\tau = (t_n)_{n=0}^{\infty}$, $\ldots$, $\upsilon = (u_n)_{n=0}^{\infty}$ are periodic sequences of elements of $\mathbb{F}_q$ with characteristic polynomials $f, g, \ldots, h$, respectively, then $f \vee g \vee \cdots \vee h$ is a characteristic polynomial of the product sequence $\sigma\tau\cdots\upsilon = (s_n t_n \cdots u_n)_{n=0}^{\infty}$.

Theorem 16. *Let $f, g, \ldots, h \in \mathbb{F}_q[x]$ be polynomials over $\mathbb{F}_q$ without multiple roots and with nonzero constant terms. The polynomial $f \vee g \vee \cdots \vee h \in \mathbb{F}_q[x]$ is irreducible if and only if the polynomials $f, g, \ldots, h$ are all irreducible and of pairwise relatively prime degrees. In this case*

$$\deg(f \vee g \vee \cdots \vee h) = \deg(f)\deg(g)\cdots\deg(h). \tag{12}$$

If $\sigma, \tau, \ldots, \upsilon$ are periodic sequences in $\mathbb{F}_q$ with irreducible minimal polynomials $f, g, \ldots, h \in \mathbb{F}_q[x]$ of pairwise relatively prime degrees and with $f(0)g(0)\cdots h(0) \neq 0$, then $f \vee g \vee \cdots \vee h$ is the minimal polynomial of the product sequence $\sigma\tau\cdots\upsilon$.

Proof. See Selmer [33, Chap. 4]. □

Example 3. Consider the binary irreducible polynomials $f(x) = x^2 + x + 1$ and $g(x) = x^3 + x + 1$. Over the splitting field $\mathbb{F}_{64}$ of $fg \in \mathbb{F}_2[x]$ we can write

$$f(x) = x^2 + x + 1 = (x - \alpha)(x - \alpha^2) = x^2 + (\alpha + \alpha^2)x + \alpha^3,$$
$$g(x) = x^3 + x + 1 = (x - \beta)(x - \beta^2)(x - \beta^4).$$

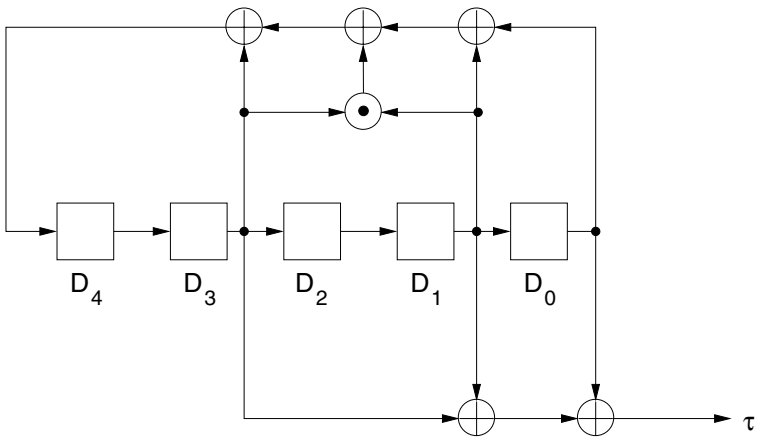


Figure 6: The 5-stage NLFSR with linear feedforward output function

In particular, for the root $\alpha \in \mathbb{F}_4$ of f we have

$$\alpha + \alpha^2 = 1 \qquad \text{and} \qquad \alpha^3 = 1.$$

Using these identities, we obtain

$$\begin{aligned}(f \vee g)(x) &= (x-\alpha\beta)(x-\alpha\beta^2)(x-\alpha\beta^4)(x-\alpha^2\beta)(x-\alpha^2\beta^2)(x-\alpha^2\beta^4) \\ &= \frac{1}{\alpha^3}\left(\frac{x}{\alpha}-\beta\right)\left(\frac{x}{\alpha}-\beta^2\right)\left(\frac{x}{\alpha}-\beta^4\right)\frac{1}{\alpha^6}\left(\frac{x}{\alpha^2}-\beta\right)\left(\frac{x}{\alpha^2}-\beta^2\right)\left(\frac{x}{\alpha^2}-\beta^4\right) \\ &= g\left(\frac{x}{\alpha}\right)g\left(\frac{x}{\alpha^2}\right) = \left(x^3+\alpha^2x+1\right)\left(x^3+\alpha x+1\right) \\ &= x^6+x^4+x^2+x+1.\end{aligned}$$

Let $\sigma = (s_n)_{n=0}^{\infty}$ be the periodic binary sequence defined by the linear recursion $s_{n+2} = s_{n+1}+s_n$ for all $n \geq 0$ and the initial values $s_0 = 0$ and $s_1 = 1$. Similarly, let $\tau = (t_n)_{n=0}^{\infty}$ be the periodic binary sequence defined by $t_{n+3} = t_{n+1} + t_n$ for all $n \geq 0$ and $t_0 = 1$ and $t_1 = t_2 = 0$. Then σ has minimal polynomial f and period $\operatorname{ord}(f) = 3$ whereas the sequence τ has minimal polynomial g and period $\operatorname{ord}(g) = 7$.

$$\begin{aligned}\sigma &= 0\,1\,1\,0\,1\,1\,0\,1\,1\,0\,1\,1\,0\,1\,1\,0\,1\,1\,0\,1\,1\,\ldots \\ \tau &= 1\,0\,0\,1\,0\,1\,1\,1\,0\,0\,1\,0\,1\,1\,1\,0\,0\,1\,0\,1\,1\,\ldots \\ \sigma\tau &= 0\,0\,0\,0\,0\,1\,0\,1\,0\,0\,1\,0\,0\,1\,1\,0\,0\,1\,0\,1\,1\,\ldots\end{aligned}$$

One readily checks that the product sequence $\sigma\tau = (w_n)_{n=0}^{\infty}$ satisfies the 6th-order linear recursion

$$w_{n+6} = w_{n+4} + w_{n+2} + w_{n+1} + w_n \qquad \text{for all } \; n \geq 0.$$

Note that the first five terms of $(w_n)_{n=0}^{\infty}$ are zero, so that the sequence $\sigma\tau$ cannot satisfy any shorter linear recursion. Thus $(f \vee g)(x) = x^6 + x^4 + x^2 + x + 1$ is the minimal polynomial of $\sigma\tau$. □

The following Lemma will be needed in the sequel. We skip its rather simple proof here.

Lemma 1. *Let a and b be positive integers. Then*

$$\gcd(2^a - 1, 2^b - 1) = 2^{\gcd(a,b)} - 1. \tag{13}$$

In particular, $2^a - 1$ and $2^b - 1$ are relatively prime if and only if a and b are.

From now on we restrict ourselves to the binary case $q = 2$.

Theorem 17. *Let $f, g, \ldots, h \in \mathbb{F}_2[x]$ be irreducible binary polynomials without multiple roots, of pairwise relatively prime degrees, and with nonzero constant terms. Then*

$$\operatorname{ord}(f \vee g \vee \cdots \vee h) = \operatorname{ord}(f)\operatorname{ord}(g)\cdots\operatorname{ord}(h). \tag{14}$$

Proof. It suffices to prove the assertion for two polynomials $f, g \in \mathbb{F}_2[x]$. Let $\deg(f) = a$, and let $\alpha \in \mathbb{F}_{2^a}$ be a root of f. Since f is irreducible, $\operatorname{ord}(f)$ coincides with the order of α as an element of the group $\mathbb{F}_{2^a}^*$, the multiplicative group formed by all nonzero elements of $\mathbb{F}_{2^a}$. The order of any element of $\mathbb{F}_{2^a}^*$ divides the order of the group $\mathbb{F}_{2^a}^*$ which is $2^a - 1$. Let $\deg(g) = b$, and let $\beta \in \mathbb{F}_{2^b}$ be a root of g. Then, by the same argument, we conclude that the order of β in $\mathbb{F}_{2^b}^*$ is equal to $\operatorname{ord}(g)$ and both numbers divide $2^b - 1$. By hypothesis, the greatest common divisor $\gcd(a, b)$ of a and b is 1, so that, by Lemma 1, $\gcd(2^a - 1, 2^b - 1) = 1$. It follows that α and β are elements of relatively prime orders in the group $\mathbb{F}_{2^{ab}}^*$. By Theorem 16, the polynomial $f \vee g$ is irreducible over $\mathbb{F}_2$. Thus the order of the polynomial $f \vee g$ is equal to the order of $\gamma = \alpha\beta$ in $\mathbb{F}_{2^{ab}}^*$. It is well known (see e.g. McEliece [23, p. 38]) that the order of the product of two elements in a commutative group is the product of the orders of the two elements if these orders are relatively prime. Hence $\operatorname{ord}(f \vee g) = \operatorname{ord}(\alpha\beta) = \operatorname{ord}(\alpha)\operatorname{ord}(\beta) = \operatorname{ord}(f)\operatorname{ord}(g)$. □

For the binary polynomials $f(x) = x^2 + x + 1$, $g(x) = x^3 + x + 1$, and $h(x) = (f \vee g)(x) = x^6 + x^4 + x^2 + x + 1$ appearing in Example 1, we find $\operatorname{ord}(f) = 3$, $\operatorname{ord}(g) = 7$, and $\operatorname{ord}(h) = 21$. Thus $21 = \operatorname{ord}(f \vee g) = \operatorname{ord}(f)\operatorname{ord}(g) = 3 \cdot 7$, in agreement with equation (14). The imposed restriction in Theorem 17 to the binary field $\mathbb{F}_2$ is necessary. Consider, for instance, the two polynomials $f(x) = x + 1 = \mathbb{F}_3[x]$ and $g(x) = x^2 + 1 \in \mathbb{F}_3[x]$ over the finite field of order 3. Then $\operatorname{ord}(f) = 2$, $\operatorname{ord}(g) = 4$, and $f \vee g = g$, and equation (14) does not hold in this case.

Theorem 18. *Let N be a positive integer, and let $\sigma = (s_n)_{n=0}^{\infty}$ be a binary periodic sequence with period $p = 2^N - 1$. The canonical factorization of the minimal polynomial $m_\sigma \in \mathbb{F}_2[x]$ of σ over $\mathbb{F}_2$ consists of distinct irreducible polynomials whose degrees all divide N. In particular, m_σ contains no repeated factors.*

Proof. Since σ has period p, the polynomial $c(x) = x^p - 1 \in \mathbb{F}_2[x]$ is a characteristic polynomial of σ. By Theorem 8, $m_\sigma(x)$ divides $c(x)$. Consequently, $m_\sigma(x)$ divides $x^{2^N} - x$, which is the product of all irreducible binary polynomials whose degrees divide N (see [22, Theorem 3.20]). □

Theorem 19. *Let N be a positive integer, and let $\sigma = (s_n)_{n=0}^{\infty}$ be a binary periodic sequence with period $P = 2^N - 1$. Let $f \in \mathbb{F}_2[x]$ be a nonzero polynomial with $\deg(f) < N$. If the canonical factorization of the minimal polynomial of σ over $\mathbb{F}_2$ contains at least one primitive binary polynomial of degree N, then the linearly filtered sequence $\tau = f(T)\sigma$ has period $\operatorname{per}(\tau) = 2^N - 1$.*

Proof. Let $m_\sigma = g_1 \cdots g_k$ be the canonical factorization of the minimal polynomial of σ in $\mathbb{F}_2[x]$. Let g_1 be a primitive binary polynomial of degree N. Then, $\operatorname{ord}(g_1) = 2^N - 1$ and $\operatorname{ord}(g_j)$ divides $2^N - 1$ for $1 \le j \le N$. An application of Theorem 13 yields that $m_\tau = m_\sigma / \gcd(m_\sigma, f)$ is divisible by the primitive polynomial g_1. Let—after a possible rearrangement of factors—the canonical factorization of the minimal polynomial of τ be given by $m_\tau = g_1 \cdots g_h$, where $h \le k$. By Theorems 9 and 10,

$$\operatorname{per}(\tau) = \operatorname{ord}(m_\tau) = \operatorname{lcm}(\operatorname{ord}(g_1), \ldots, \operatorname{ord}(g_k)) = 2^N - 1.$$

□

Theorem 20. *Let $N \geq 1$, and let $\sigma = (s_n)_{n=0}^{\infty}$ be a binary periodic sequence of period $p = 2^N - 1$ and of span N. If the zero vector $\mathbf{0} \in \mathbb{F}_2^N$ does not occur among the N-tuples $\mathbf{s}_n = (s_n, s_{n+1}, \ldots, s_{n+N-1})$, $0 \leq n \leq p-1$, then $x - 1$ does not divide the minimal polynomial m_σ of σ.*

Proof. The N-tuples $\mathbf{s}_n$, $0 \leq n \leq p-1$, run through all nonzero vectors of $\mathbb{F}_2^N$. Therefore, the element 1 occurs exactly 2^{N-1} times among the first coordinates of these N-tuples. Thus

$$s_0 + s_1 + \cdots + s_{p-1} = 0.$$

Since σ is periodic with period p, we get

$$s_n + s_{n+1} + \ldots + s_{n+p-1} = 0 \qquad \text{for all } n \geq 0,$$

which means that

$$c(x) = x^{p-1} + x^{p-2} + \cdots + x + 1 \in \mathbb{F}_2[x]$$

is a characteristic polynomial of σ. Since $c(1) \neq 0$, the polynomial $c(x)$ is not divisible by $x - 1$, nor is the minimal polynomial $m_\sigma(x)$ which is a divisor of $c(x)$. □

Theorem 21. *Let $S, T, \ldots, U$ be pairwise relatively prime integers greater than 1. Let $\sigma = (s_n)_{n=0}^{\infty}$, $\tau = (t_n)_{n=0}^{\infty}$, $\ldots$, $\upsilon = (u_n)_{n=0}^{\infty}$ be binary periodic sequences of periods $\mathrm{per}(\sigma) = 2^S - 1$, $\mathrm{per}(\tau) = 2^T - 1$, $\ldots$, $\mathrm{per}(\upsilon) = 2^U - 1$, respectively. Assume that the canonical factorizations over $\mathbb{F}_2$ of the minimal polynomials of $\sigma, \tau, \ldots, \upsilon$ are*

$$m_\sigma = \prod_{i=1}^{s} f_i, \qquad m_\tau = \prod_{j=1}^{t} g_j, \qquad \ldots, \qquad m_\upsilon = \prod_{k=1}^{u} h_k. \tag{15}$$

Then the minimal polynomial of the product sequence $\sigma\tau\cdots\upsilon = (s_n t_n \cdots u_n)_{n=0}^{\infty}$ is given by

$$m_{\sigma\tau\cdots\upsilon} = \prod_{i=1}^{s}\prod_{j=1}^{t}\cdots\prod_{k=1}^{u}(f_i \vee g_j \vee \cdots \vee h_k). \tag{16}$$

In fact, (16) *represents the canonical factorization of the minimal polynomial of $\sigma\tau\cdots\upsilon$ over $\mathbb{F}_2$.*

Proof. It suffices to carry out the details of the proof for the product of two such sequences σ and τ. The general statement then follows by induction. Consider the canonical factorization of the minimal polynomials m_σ and m_τ in (15). By Theorem 18, the irreducible polynomials $f_1, \ldots, f_s \in \mathbb{F}_2[x]$ are distinct and $\deg(f_i)$ divides S for $1 \leq i \leq s$. Similarly, the irreducible polynomials $g_1, \ldots, g_t$ are distinct and $\deg(g_j)$ divides T for $1 \leq j \leq t$. Since the sequences σ and τ are periodic, their minimal polynomials m_σ and m_τ are not divisible by x. Thus, the first-degree irreducible polynomial $p(x) = x$ does not occur among the polynomials $f_1, \ldots, f_s$ and $g_1, \ldots, g_t$.

By Theorem 12, the sequences σ and τ possess unique representations

$$\sigma = \sum_{i=1}^{s} \sigma_i \quad \text{and} \quad \tau = \sum_{j=1}^{t} \tau_j,$$

where σ_i is a binary periodic sequence with minimal polynomial f_i for $1 \leq i \leq s$, and τ_j is a binary periodic sequence with minimal polynomial g_j for $1 \leq j \leq t$. It follows that

$$\sigma\tau = \sum_{i=1}^{s} \sum_{j=1}^{t} \sigma_i \tau_j.$$

By hypothesis, $\gcd(S, T) = 1$. It follows that for each $i \in \{1, \ldots, s\}$ and $j \in \{1, \ldots, t\}$, the corresponding irreducible polynomials f_i and g_j have relatively prime degrees. Invoking Theorem 16, we conclude that for each $i \in \{1, \ldots, s\}$ and $j \in \{1, \ldots, t\}$, the sequence $\sigma_i \tau_j$ has the irreducible minimal polynomial $f_i \vee g_j \in \mathbb{F}_2[x]$.

As will be shown below, the irreducible polynomials $f_i \vee g_j$, $1 \leq i \leq s$, $1 \leq j \leq t$, are distinct. Another application of Theorem 12 therefore shows that the minimal polynomial of $\sigma\tau$ has the form

$$m_{\sigma\tau} = \prod_{i=1}^{s} \prod_{j=1}^{t} (f_i \vee g_j). \tag{17}$$

It remains to show that the polynomials $f_i \vee g_j$, $1 \leq i \leq s$, $1 \leq j \leq t$, are distinct. To see this, let f_i and f_i' be any two factors from the canonical factorization of m_σ, and let g_j and g_j' be any two factors from the canonical factorization of m_τ. Assume to the contrary that the two irreducible polynomials $f_i \vee g_j$ and $f_i' \vee g_j'$ are equal. Note that two irreducible polynomials over the finite field $\mathbb{F}_q$ are equal if and only if they have a common root (in some extension field of $\mathbb{F}_q$). Let γ be a common root of $f_i \vee g_j$ and $f_i' \vee g_j'$. Then we can write γ in the form

$$\gamma = \alpha\beta = \alpha'\beta', \tag{18}$$

where α, β, α', and β' are roots of the polynomials f_i, g_j, f_i', and g_j', respectively. Since α is a root of the irreducible polynomial f_i, we have $\alpha \in \mathbb{F}_{2^{\deg(f_i)}}$, which is a subfield of $\mathbb{F}_{2^S}$, as $\deg(f_i)$ divides S. Similarly, we conclude that $\alpha' \in \mathbb{F}_{2^S}$ and $\beta, \beta' \in \mathbb{F}_{2^T}$. From (18) we obtain

$$\frac{\alpha}{\alpha'} = \frac{\beta'}{\beta}. \tag{19}$$

Clearly, $\alpha/\alpha' \in \mathbb{F}_{2^S}$ and $\beta'/\beta \in \mathbb{F}_{2^T}$. Since S and T are relatively prime, we have $\mathbb{F}_{2^S} \cap \mathbb{F}_{2^T} = \mathbb{F}_2$, so that both sides of (19) must be equal to 1. Hence $\alpha = \alpha'$ and $\beta = \beta'$. This, however, implies $f_i = f_i'$ and $g_j = g_j'$. □

Corollary 2. *Let* $\sigma = (s_n)_{n=0}^{\infty}$, $\tau = (t_n)_{n=0}^{\infty}$, $\ldots$, $\upsilon = (u_n)_{n=0}^{\infty}$ *be binary periodic sequences of periods* $\mathrm{per}(\sigma) = 2^S - 1$, $\mathrm{per}(\tau) = 2^T - 1$, $\ldots$, $\mathrm{per}(\upsilon) = 2^U - 1$, *and linear complexities* $L(\sigma), L(\tau), \ldots, L(\upsilon)$ *respectively. If the integers* $S, T, \ldots, U$ *be pairwise relatively prime and greater than* 1, *then the product sequence* $\sigma\tau\cdots\upsilon = (s_n t_n \cdots u_n)_{n=0}^{\infty}$ *has linear complexity*

$$L(\sigma\tau\cdots\upsilon) = L(\sigma)L(\tau)\cdots L(\upsilon), \tag{20}$$

and period

$$\mathrm{per}(\sigma\tau\cdots\upsilon) = \left(2^S - 1\right)\left(2^T - 1\right)\cdots\left(2^U - 1\right). \tag{21}$$

Proof. Let the minimal polynomials of $\sigma, \tau, \ldots, \upsilon$ be given by the expressions in (15). Then, by Theorem 21 and equation (12), we obtain

$$\begin{aligned}
L(\sigma\tau\cdots\upsilon) = \deg(m_{\sigma\tau\cdots\upsilon}) &= \sum_{i=1}^{s}\sum_{j=1}^{t}\cdots\sum_{k=1}^{u}\deg(f_i \vee g_j \vee \cdots \vee h_k) \\
&= \sum_{i=1}^{s}\sum_{j=1}^{t}\cdots\sum_{k=1}^{u}\deg(f_i)\deg(g_j)\cdots\deg(h_k)) \\
&= \left(\sum_{i=1}^{s}\deg(f_i)\right)\left(\sum_{j=1}^{t}\deg(g_j)\right)\cdots\left(\sum_{k=1}^{u}\deg(h_k)\right) \\
&= L(\sigma)L(\tau)\cdots L(\upsilon).
\end{aligned}$$

This proves equation (20). For the proof of (21), recall that over an arbitrary finite field $\mathbb{F}_q$, the period of a periodic sequence of field elements is equal to the order of the sequence's minimal polynomial (Theorem 9). Using Theorems 21, 10, and 17, we get

$$\begin{aligned}
&\mathrm{per}(\sigma\tau\cdots\upsilon) = \mathrm{ord}(m_{\sigma\tau\cdots\upsilon}) \\
&\quad = \mathrm{lcm}\{\mathrm{ord}(f_i \vee g_j \vee \cdots \vee h_k) : 1 \le i \le s,\ 1 \le j \le t, \ldots, 1 \le k \le u\} \\
&\quad = \mathrm{lcm}\{\mathrm{ord}(f_i)\,\mathrm{ord}(g_j)\cdots\mathrm{ord}(h_k) : 1 \le i \le s,\ 1 \le j \le t, \ldots, 1 \le k \le u\} \\
&\quad = \mathrm{lcm}\{\mathrm{ord}(f_i) : 1 \le i \le s\}\,\mathrm{lcm}\{\mathrm{ord}(g_j) : 1 \le j \le t\}\cdots\mathrm{lcm}\{\mathrm{ord}(h_k) : 1 \le k \le u\} \\
&\quad = \mathrm{ord}(m_\sigma)\,\mathrm{ord}(m_\tau)\cdots\mathrm{ord}(m_\upsilon) \\
&\quad = \mathrm{per}(\sigma)\,\mathrm{per}(\tau)\cdots\mathrm{per}(\upsilon) \\
&\quad = \left(2^S - 1\right)\left(2^T - 1\right)\cdots\left(2^U - 1\right).
\end{aligned}$$

To justify the fourth equality in the above argument we note that $\mathrm{ord}(f_i)$ divides $2^S - 1$, $\mathrm{ord}(g_j)$ divides $2^T - 1$, and $\mathrm{ord}(h_k)$ divides $2^U - 1$ for all $1 \le i \le s$, $1 \le j \le t$, and $1 \le k \le u$, and that the numbers $2^S - 1$, $2^T - 1$, $\ldots$, $2^U - 1$ are pairwise relatively prime according to Lemma 1. □

Example 4. Consider the 4-stage NLFSR with feedback function

$$F(x_0, x_1, x_2, x_3) = x_0 + x_1 + x_2 + x_1x_2,$$

and the 5-stage NLFSR defined by the feedback function

$$G(x_0, x_1, x_2, x_3, x_4) = x_0 + x_1 + x_3 + x_1x_3.$$

Using any nonzero initial state vector of $\mathbb{F}_2^4$, the first feedback shift register will produce a periodic binary sequence σ of period $\mathrm{per}(\sigma) = 15$ and linear complexity $L(\sigma) = 14$. For instance, if we use the initial state vector $(0, 0, 0, 1)$, we get

$$\sigma = (0\,0\,0\,1\,0\,1\,1\,0\,1\,0\,0\,1\,1\,1\,1)^\infty$$

The minimal polynomial of σ is

$$m_\sigma(x) = x^{14} + x^{13} + \cdots + x + 1 = f_1(x)f_2(x)f_3(x)f_4(x),$$

where $f_1(x) = x^2+x+1$, $f_2(x) = x^4+x^3+x^2+x+1$, $f_3(x) = x^4+x+1$, $f_4 = x^4+x^3+1$. Similarly, if we initialize the second shift register with the nonzero vector $(0, 0, 0, 0, 1)$, it generates the periodic sequence

$$\tau = (0\,0\,0\,0\,1\,0\,1\,0\,1\,1\,1\,0\,1\,0\,0\,1\,1\,0\,1\,1\,0\,0\,1\,0\,0\,0\,1\,1\,1\,1\,1)^\infty$$

of period $\mathrm{per}(\tau) = 31$ and linear complexity $L(\tau) = 30$. The corresponding minimal polynomial is

$$m_\tau(x) = x^{30} + x^{29} + \cdots + x + 1 = g_1(x)g_2(x)g_3(x)g_4(x)g_5(x)g_6(x),$$

where $g_1, \ldots, g_6$ are the six irreducible (and primitive) polynomials in $\mathbb{F}_2[x]$ of degree 5. The product sequence $\sigma\tau$ has period $\mathrm{per}(\sigma\tau) = 15 \cdot 31 = 465$ and linear complexity $L(\sigma\tau) = 14 \cdot 30 = 420$. The canonical factorization of the minimal polynomial of $\sigma\tau$ consists of $4 \cdot 6 = 24$ irreducible binary polynomials. Of these polynomials six have degree 10 and order 93, six have degree 20 and order 155, and twelve have degree 20 and order 465.

The irreducible factors:

	g_1	g_2	g_3	g_4	g_5	g_6
f_1	$f_1 \vee g_1$	$f_1 \vee g_2$	$f_1 \vee g_3$	$f_1 \vee g_4$	$f_1 \vee g_5$	$f_1 \vee g_6$
f_2	$f_2 \vee g_1$	$f_2 \vee g_2$	$f_2 \vee g_3$	$f_2 \vee g_4$	$f_2 \vee g_5$	$f_2 \vee g_6$
f_3	$f_3 \vee g_1$	$f_3 \vee g_2$	$f_3 \vee g_3$	$f_3 \vee g_4$	$f_3 \vee g_5$	$f_3 \vee g_6$
f_4	$f_4 \vee g_1$	$f_4 \vee g_2$	$f_4 \vee g_3$	$f_4 \vee g_4$	$f_4 \vee g_5$	$f_4 \vee g_6$

The degrees of the irreducible factors:

	5	5	5	5	5	5
2	10	10	10	10	10	10
4	20	20	20	20	20	20
4	20	20	20	20	20	20
4	20	20	20	20	20	20

The orders of the irreducible factors:

	31	31	31	31	31	31
3	93	93	93	93	93	93
5	155	155	155	155	155	155
15	465	465	465	465	465	465
15	465	465	465	465	465	465

Note that there are exactly $\varphi(93)/10 = 6$ irreducible binary polynomials of degree 10 and order 93, $\varphi(155)/20 = 6$ irreducible binary polynomials of degree 20 and order 155, and $\varphi(465)/20 = 12$ irreducible binary polynomials of degree 20 and order 465 (compare [22, Theorem 3.5]). All these polynomials appear in the canonical factorization of $m_{\sigma\tau}$. This is a consequence of the fact that the sequences σ and τ have maximum linear complexities $L(\sigma) = 2^4 - 2 = 14$ and $L(\tau) = 2^5 - 2 = 30$, respectively. □

Appendix B

B Hardware considerations and an overview on figures of merits for hardware implementations

In this section we will compare several implementation variants of the proposed Achterbahn stream cipher with other stream ciphers from different fields of applications. A5/1, E0 and RC4 are well known stream ciphers, because they are used in standards. A5/1 is specified for GSM applications, E0 is used in Bluetooth wireless communication, and RC4 in the IEEE 802.11b WLAN standard. We also consider the AES (Rijndael) block cipher standard which can be operated in the output feedback mode (OFB) as a keystream generator. The comparison is based on several figures of merit, which are defined and briefly discussed in the following sections.

When evaluating the implementation properties of a certain algorithm in hardware it is important to define the figures of merit precisely. In general the ultimate figure will be a performance/cost ratio. The cost function cannot be described easily, because it depends on several factors which must be weighted differently in different applications. The most important common factors, however, are the *size* of the implementation, the *power consumption*, the *throughput*, the *implementation efficiency*, and the capability of the algorithm to trade off one factor against another one. The latter property is often termed *scalability*.

B.1 Area and power

The size of the implementation of an algorithm depends strongly on the minimum feature size of the technology, which is the dimension of the smallest feature actually constructed in the manufacturing process. It also depends on the specific circuit design style, such as CMOS or DCVSL [30], and the number of available metal layers for wire routing. Hence, it is necessary to resort to an approximate, technology and circuit style independent measure. A commonly used measure for the size of a design is the *number of NAND gate equivalents* (GE). This is the area of the circuit implementation divided by the area of the smallest NAND gate in the used standard CMOS cell library. Tab. 1 shows the sizes of some gates in units of GE for a contemporary standard cell library. All CMOS standard cell libraries contain gates with more than two inputs, which generally reduces area, power consumption, and gate propagation delay of a circuit. Examples are AND and OR gates with three or four inputs or XOR gates with three inputs. Obviously, a 4-input NAND gate is smaller than the equivalent circuit built from three 2-input AND gates. Thus the gate equivalent count of a design will always reflect the optimized mixture of available multi-input cells, but not the count of binary operations in the algorithm.

The power consumption of a CMOS design is also related to the gate equivalent

gate	size [GE]	gate	size [GE]
2-input NAND	1	2-bit MUX	2.50
2-input AND	1.25	2-input XOR	2.25
3-input AND	1.50	3-input XOR	4.00
4-input AND	1.75	register bit	6.00

Table 1: Typical sizes of some gates in units of NAND gate equivalents (GE).

count. However, the dynamic power consumption of the implementations of two different algorithms, which have approximately the same gate count, can differ strongly. Power consumption estimations for an algorithm require a detailed analysis of the dynamic switching activity of the gates.

B.2 Throughput

The throughput of a stream cipher is conveniently defined as the average number of output ciphertext bits per second, which is in a synchronous design equivalent to the average number $\langle \mathcal{N} \rangle$ of output bits per clock cycles times the clock frequency f,

$$\mathcal{P} = \langle \mathcal{N} \rangle f. \tag{22}$$

Hence, the throughput can be enhanced either by increasing the clock frequency, or by increasing the average number of output bits per clock cycle. The maximum clock frequency a circuit can be operated at is determined by several factors, such as the feature size of the available technology and the operating voltage. Here we can hope, that for the next years Moore's Law will contribute to speed up existing algorithms. However, there is an important factor which is under full control of the algorithm designer. The *number of gate propagation delays* in the longest combinational path of the design, the so-called *critical path*, will ultimately limit the maximally reachable clock frequency. The gate propagation delay is the time required for switching the output of a gate after an input signal has changed. Generally, the sum of the gate propagation delays of the gates in the critical path must be smaller than the cycle time $(1/f)$. Hence, an algorithm allowing for an implementation with a smaller number of gate propagation delays in the critical path can be operated at higher frequencies. Various pipelining techniques can be used to cut down the critical path by some amount. However, this usually leads to a rapidly increasing number of gates in the design. A more complex algorithm will lead to a steeper rise of the size of the design. For the application of a stream cipher in a hardware design it is important that the cipher itself does not contain the critical path of the design. Hence the number of gate propagation delays in the critical path of the cipher (without already implementing pipelining) is an important technology independent figure of merit which determines the maximally reachable throughput.

Another significant factor determining the effective throughput in practice is the overhead time for setting up the encryption in the communication protocol. In the majority of applications the communication is packet-oriented: the message text is split into small packets, which are separately encrypted and transmitted. Typical packet lengths are, e.g., 224 bits in GSM applications, 512 bits for most of the TCP/IP packets in the internet, or up to 2745 bits in the Bluetooth wireless communication standard. To achieve a resynchronization after a transmission error the packets are marked with a frame number and other public information (like time stamps). To prevent the reuse of key material this block of public information, called the initialization value vector, is combined with the secret key. The period of time, starting with the processing of the initialization vector, until the first output of cipher text, is called the *resynchronization time*. Hence, the throughput is reduced by a factor which depends on the resynchronization time and the size of the packets. Consequently, an important figure of merit is a small resynchronization time.

B.3 Implementation efficiency

It is well known that different algorithms can be more or less well suited for a hardware implementation. In order to express, how efficiently a stream cipher design uses the gates to achieve a certain throughput, we introduce the *implementation efficiency* $\mathcal{E}$ of a stream cipher. Normalizing the average number $\mathcal{N}$ of generated key stream bits per cycle by the number of gate equivalents $\mathcal{G}$ of the implementation, we define

$$\mathcal{E} = \frac{\mathcal{N}}{\mathcal{G}} \left[\frac{bit}{cyc \cdot kGE} \right].$$

For convenience the number of gate equivalents is given in units of $1000\,\mathrm{GE} = 1\,\mathrm{kGE}$. This figure reflects how many kGE are necessary to generate one bit of keystream per cycle on average.

B.4 Scalability

To cover a broad range of possible applications a stream cipher algorithm should be suitable for a very small implementation with small throughput requirements, as well as for high throughput applications, where a larger area and power consumption can be tolerated. Examples for the first kind are mobile and smartcard applications. Future pervasive computing applications, such as RFID tags or sensor networks, will pose even more restrictive area and power constraints on the implementation of cryptographic primitives. Hence, the *minimal implementation size* of an algorithm is certainly an important figure of merit. Examples for applications with an intermediate bandwidth are video signals with serial bit rates between 143.18 Mbps (NTSC standard) and 1.458 Gbps (high definition video standard SMPTE 292M). On the high-end

scale there are Gigabit ATM networks and I/O interconnections for distributed computing with bandwidths between 1 Gbps and 30 Gbps (e.g. InfiniBand). In these fields of applications the *maximum throughput* is the important figure of merit.

B.5 Discussion

We implemented the Achterbahn stream cipher in VHDL and synthesized the design for a 0.13μ CMOS standard cell library. The design is configurable for bit-serial, 2-bit, 4-bit, or 8-bit parallelization. Additionally to the full-fledged KSG, which is the suggested KSG in our stream cipher proposal, we will also consider the reduced version of the KSG described in Section 5. Naturally, in the design of the stream cipher we strived for for minimum area without introducing pipelining in order to increase the maximum frequency. In the first four columns of Tab. 7 the figures of merit for the eight different implementation versions of Achterbahn are reported. The figures in parenthesis refer to the the reduced version of the KSG.

The bit-serial implementation of the Achterbahn stream cipher has a comparably small *minimal implementation size*[1]. The size of the design is approximately 3000 gate equivalents (GE), and the resynchronization time is given by 112 cycles plus the length l of the initialization vector. This implementation is suitable for securing the communication in pervasive computing applications, like RFID tags, for contactless or contact-based smartcard applications, or for wireless communications with moderate throughput requirements. It is also appropriate for securing serial data links in multi-chip solutions, for masking on-chip signals in security devices, or as a pseudorandom generator. The small number of gate propagation delays in the critical path allows very high target frequencies. In a 0.13μm CMOS technology a frequency of more than 1 GHz can be achieved.

As described in previous sections, the Achterbahn stream cipher consists of rather simple components: feedback shift registers, linear feedforward output functions, and a tiny (from a hardware point of view) Boolean combining function. In Section 6, we have seen due to the simplicity of the components and due to possible fast (parallel) implementations of the underlying nonlinear feedback shift registers, the throughput can be scaled in a straightforward manner. Paradoxically, the bit sequential nature of a feedback shift register neither prohibits an efficient parallelization nor a pipelined implementation. The implementation with 2-fold parallelization, that is with step size $k = 2$, is only 15% larger, whereas the throughput is increased by a factor of 2, and the resynchronization time is also reduced by a factor of 2. The implementation with 8-fold parallelization, corresponding to the step size $k = 8$, is 2.5 times larger while the resynchronization time is 8 times smaller (i.e. 22 clock cycles for an initialization vector of maximum length $l = 64$). It is important to note that the number of gate delays in the critical path is not increased by the parallelization. Hence the 8-bit parallel design can be operated with the same maximum frequency as the bit

[1]The smaller A5/1 cannot be considered as secure enough for most applications

serial design—consequently the maximum throughput is 8 times greater. In a 0.13μm CMOS technology, a throughput of more than 8 Gbps can be achieved. If even higher thoughputs are required one or two pipeline stages in the feedback functions of the NLFSR's, the linear feedforward functions, and the Boolean combining function can be introduced. The resulting reduction of the number of gate delays in the critical path will increase the maximum throughput to values between 15 Gbps and 30 Gbps. It is also possible to push the degree of parallelization (the step size) beyond $k = 8$, say up to $k = 16$ to further increase the throughput. So far, we have not explored the implication for the size of the design. Furthermore, the step size k is not limited to an even number. All small integer values are possible allowing for a fine-tuning of the throughput.

We implemented Achterbahn on also an FPGA (of type Stratix-I). In the full-fledged 8-bit parallel version the design was operated at a frequency of 240 MHz.

We now compare Achterbahn with three implementations of the AES with 128 bit key length, for which figures are publicly available. We assume that the AES is operated in the OFB mode. Hence, it has a state of 128 bit which is updated by repeated encryption operations, starting with a 128 bit initial value IV. The resynchronization time for this configuration is then given by the time required for one encryption operation. For the area comparison with the genuine stream ciphers we have to add to the reported areas the gates necessary to implement the state. According to Tab. 1 the implementation of the state corresponds to 768 GE. The considered AES implementations are not protected against side channel attacks, such as differential power analysis (DPA). The well known masking approaches [1, 4] for the nonlinear operations of the S-Boxes lead to additional hardware costs of roughly 200 GE for each S-box [35]. Furthermore, the masking leads to a significant increase of the propagation delay of the critical path. In reference [35] fifteen additional gate delay times for a specific implementation are reported. To have a basis for the comparison with the Achterbahn stream cipher, which already has a resynchronization mechanism, which is presumably robust against side-channel attacks, the figures for the AES implementations are corrected by the corresponding overhead areas and delay time penalties. In the footnotes of Tab. 1 the original figures are reported. Masking of the AES is also necessary during the keystream generation, because the key is inserted in each encryption of the state. It is believed that stream ciphers are in general robust against DPA during keystream generation. The other three reference stream cipher implementations, E0, A5/1, and RC4, do not contain specific DPA counter measures. Although the key sizes and the sizes of the initial values, are different for the considered stream ciphers, the comparison gives an indication about strengths and scalability of the different designs. Some designs (E0, A5/1, RC4) have already been attacked successfully.

The proposed Achterbahn stream cipher has a comparably high implementation efficiency. The efficiency grows with the degree of parallelization up to the step size $k = 8$. The efficiency of the 8-bit parallel version is approximately two times greater than the efficiency of the high speed AES implementation. At a frequency of approx-

imately 190 MHz the 8-bit parallel Achterbahn implementation reaches the maximum throughput of 1.5 Gbps of the high speed AES implementation. However, the frequency of the Achterbahn design can still be increased by more than a factor of 5. The introduction of pipeline stages and parallelization beyond the step size $k = 8$ are options to further increase the throughput.

Appendix C

C The feedback functions

In this Appendix, the feedback functions of the eight primitive binary nonlinear feedback shift registers are given which constitute the core of the keystream generator. See Definition 1 in Appendix A for the definition of a primitive feedback shift register. Note that in each feedback function the last seven variables appear only linearly. This is a decisive advantage for high-speed implementations of the shift registers.

$$\begin{aligned}
A(x_0, x_1, \ldots, x_{21}) &= x_0 + x_5 + x_6 + x_7 + x_{10} + x_{11} + x_{12} + x_{13} + x_{17} + x_{20} \\
&\quad + x_2x_7 + x_4x_{14} + x_8x_9 + x_{10}x_{11} + x_1x_4x_{11} + x_1x_4x_{13}x_{14};
\end{aligned}$$

$$\begin{aligned}
B(x_0, x_1, \ldots, x_{22}) &= x_0 + x_6 + x_7 + x_9 + x_{11} + x_{12} + x_{14} + x_{15} + x_{17} + x_{19} + x_{21} \\
&\quad + x_1x_4 + x_2x_7 + x_5x_9 + x_6x_{10} + x_2x_4x_8 + x_1x_3x_5x_{10} \\
&\quad + x_4x_{11}x_{12}x_{13};
\end{aligned}$$

$$\begin{aligned}
C(x_0, x_1, \ldots, x_{24}) &= x_0 + x_1 + x_3 + x_5 + x_6 + x_7 + x_9 + x_{12} + x_{14} + x_{15} + x_{17} \\
&\quad + x_{18} + x_{22} + x_1x_6 + x_4x_{13} + x_8x_{16} + x_{12}x_{15} + x_5x_{11}x_{14} \\
&\quad + x_1x_4x_{11}x_{15} + x_2x_5x_8x_{10};
\end{aligned}$$

$$\begin{aligned}
D(x_0, x_1, \ldots, x_{25}) &= x_0 + x_1 + x_4 + x_5 + x_7 + x_8 + x_9 + x_{13} + x_{14} + x_{16} + x_{20} \\
&\quad + x_{24} + x_1x_6 + x_4x_7 + x_{12}x_{16} + x_{15}x_{17} + x_4x_{15}x_{17} + x_7x_9x_{10} \\
&\quad + x_1x_3x_{14}x_{16} + x_8x_{11}x_{12}x_{17};
\end{aligned}$$

$$\begin{aligned}
E(x_0, x_1, \ldots, x_{26}) &= x_0 + x_1 + x_2 + x_6 + x_8 + x_9 + x_{10} + x_{13} + x_{14} + x_{16} + x_{19} \\
&\quad + x_{21} + x_{23} + x_1x_8 + x_3x_{12} + x_{11}x_{17} + x_{15}x_{18} + x_5x_6x_{15} \\
&\quad + x_3x_5x_{16}x_{17} + x_7x_{12}x_{14}x_{15};
\end{aligned}$$

$$\begin{aligned}
F(x_0, x_1, \ldots, x_{27}) &= x_0 + x_1 + x_2 + x_7 + x_{15} + x_{17} + x_{19} + x_{20} + x_{22} + x_{27} \\
&\quad + x_9x_{17} + x_{10}x_{18} + x_{11}x_{14} + x_{12}x_{13} + x_5x_{14}x_{19} + x_6x_{10}x_{12} \\
&\quad + x_6x_9x_{17}x_{18} + x_{10}x_{12}x_{19}x_{20};
\end{aligned}$$

$$\begin{aligned}
G(x_0, x_1, \ldots, x_{28}) &= x_0 + x_2 + x_3 + x_5 + x_6 + x_9 + x_{14} + x_{15} + x_{16} + x_{18} \\
&\quad + x_{21} + x_{27} + x_5x_7 + x_6x_{20} + x_{10}x_{14} + x_{13}x_{18} + x_8x_{19}x_{21} \\
&\quad + x_{11}x_{16}x_{18} + x_1x_5x_{15}x_{21} + x_2x_7x_{17}x_{20};
\end{aligned}$$

$$\begin{aligned}
H(x_0, x_1, \ldots, x_{30}) &= x_0 + x_3 + x_5 + x_7 + x_{10} + x_{16} + x_{17} + x_{18} + x_{19} + x_{20} \\
&\quad + x_{21} + x_{24} + x_{30} + x_5x_{15} + x_{11}x_{18} + x_{16}x_{22} + x_{17}x_{21} \\
&\quad + x_1x_2x_{19} + x_1x_{12}x_{14}x_{17} + x_2x_5x_{13}x_{20}.
\end{aligned}$$

References

[1] M.-L. Akkar and C. Giraud: An Implementation of DES and AES, Secure against Some Attacks, *CHES 2001*, (Ç. K. Koç, D. Naccache, and C. Paar, eds.), Lecture Notes in Computer Science, vol. 2162, pp. 309–318, Springer-Verlag, 2001.

[2] L. Batina, J. Lano, N. Mentens, S. B. Örs, B. Preneel, I. Verbauwhede: Energy, Performance, Area versus Security Trade-offs for Stream Ciphers, *Workshop Records of SASCS – The State of the Art of Stream Ciphers* (Brugge, Belgium, 2004), pp. 302–310. Available at http://www.isg.rhul.ac.uk/research/projects/excrypt/stvl/sasc-record.zip

[3] S. R. Blackburn: A generalization of the discrete Fourier transform: determining the minimal polynomial of a periodic sequence, *IEEE Trans. Inform. Theory* **40**, 1702–1704 (1994).

[4] J. Blömer, J. G. Merchan, and V. Krummel: Provably Secure Masking of AES, *Selected Areas in Cryptography – SAC 2004*, Lecture Notes in Computer Science, vol. 3357, pp. 69–83, Springer-Verlag, 2004.

[5] N. G. deBruijn: A combinatorial problem, *Indag. Math.* **8**, 461–467 (1946).

[6] A. Canteaut: Open problems related to algebraic attacks on stream ciphers (invited talk), *Proc. of The International Workshop on Coding and Cryptography WCC'2005* (Bergen, Norway, 2005), P. Charpin and Ø. Ytrehus, eds., pp. 1-10.

[7] N. Courtois and W. Meier: Algebraic attacks on stream ciphers with linear feedback, *Advances in Cryptology – EUROCRYPT 2003* (E. Biham, ed.), Lecture Notes in Computer Science, vol. 2656, pp. 345–359, Springer-Verlag, 2003.

[8] Z.-D. Dai and J.-H. Yang: Linear complexity of periodically repeated random sequences, *Advances in Cryptology – EUROCRYPT '91* (D. W. Davies, ed.), Lecture Notes in Computer Science, vol. 547, pp. 168–175, Springer-Verlag, 1991.

[9] C. De Cannière, J. Lano, and B. Preneel: Comments on the rediscovery of time memory data tradeoffs. Available at http://www.ecrypt.eu.org/stream/TMD.pdf

[10] M. Feldhofer, S. Dominikus, and J. Wolkersdorfer: Strong Authentication for RFID Systems Using the AES Algorithm, *CHES 2004*, Lecture Notes in Computer Science, vol. 3156, pp. 357-370, Springer-Verlag, 2004.

[11] B. M. Gammel and R. Göttfert: Combining certain nonlinear feedback shift registers, *Workshop Record of SASC – The State of the Art of Stream Ciphers* (Brugge, Belgium, 2004), pp. 234–248. Available at http://www.isg.rhul.ac.uk/research/projects/excrypt/stvl/sasc-record.zip

[12] B. M. Gammel and R. Göttfert: Linear filtering of nonlinear shift register sequences, *Proc. of The International Workshop on Coding and Cryptography WCC'2005* (Bergen, Norway, 2005), P. Charpin and Ø. Ytrehus, eds., pp. 117-126.

[13] C. Gehrmann, M. Näslud: ECRYPT: Yearly Report on Algorithms and Keysizes (2004), 1 March 2004, Revision 1.0. Available at http://www.ecrypt.eu.org/documents.html

[14] R. Göttfert: *Produkte von Schieberegisterfolgen*, Ph.D. Thesis, Univ. of Vienna, 1993.

[15] J. H. Hoch and A. Shamir: Fault analysis of stream ciphers, *CHES 2004* (M. Joye and J.-J. Quisquater, eds.) Lecture Notes in Computer Science, vol. 3156, pp. 240–253, Springer-Verlag, 2004.

[16] J. Hong and P. Sarkar: Rediscovery of time memory tradeoffs, Cryptology ePrint Archive, Report 2005/090, 2005.

[17] C. J. A. Jansen: *Investigations On Nonlinear Streamcipher Systems: Construction and Evaluation Methods*, Ph.D. Thesis, Technical Univ. of Delft, Delft, 1989.

[18] P. Kitsos, N. Sklavos, K. Papadomanolakis, O. Koufopavlou: Hardware Implementation of Bluetooth Security, *IEEE Pervasive Computing*, **2**(1), 21-29 (2003).

[19] P. Kitsos, G. Kostopoulos, N. Sklavos, O. Koufopavlou: Hardware Implementation of the RC4 Stream Cipher, *Proc. of the 46th IEEE Midwest Symposium on Circuits and Systems*, pp. 27-30, Cairo, Egypt, 2003.

[20] A fast cryptographic checksum algorithm based on stream ciphers, *Advances in Cryptology – AUSCRYPT '92* (J. Seberry and Y. Zheng, eds.), Lecture Notes in Computer Science, vol. 718, pp. 339–348, Springer-Verlag, 1993.

[21] P. L'Ecuyer and R. Simard: Test U01—A Software Library in ANSI C for Empirical Testing of Random Number Generators, January 14, 2004, Version 0.6.0. Available at http://www.iro.umontreal.ca/~ simardr/TestU01.zip

[22] R. Lidl and H. Niederreiter: *Finite Fields*, Encyclopedia of Mathematics and Its Applications, vol. 20, Addison-Wesley, Reading, Mass., 1983. (Now Cambridge Univ. Press.)

[23] R. J. McEliece: *Finite Fields for Computer Scientists and Engineers*, Kluwer, Boston, 1987.

[24] W. Meidl and H. Niederreiter: On the expected value of the linear complexity and the k-error linear complexity of periodic sequences, *IEEE Trans. Inform. Theory* **48**, 2817–2825 (2002).

[25] W. Meier and O. Staffelbach: Fast correlation attacks on certain stream ciphers, *J. Cryptology* **1**, 159–176 (1989).

[26] S. Morioka and A. Satoh: An Optimized S-Box Circuit Architecture for Low Power AES Design, *CHES 2002*, Lecture Notes in Computer Science, vol. 2523, pp.172-186, Springer-Verlag, 2002.

[27] H. Niederreiter: Distribution properties of feedback shift register sequences, *Problems Control Inform. Theory* **15**, 19–34 (1986).

[28] H. Niederreiter: Cryptology—The mathematical theory of data security, *Prospects of Mathematical Science* (T. Mitsui, K. Nagasaka, and T. Kano, eds.), pp. 189–209, World Sci. Pub., Singapore, 1988.

[29] H. Niederreiter: *Random Number Generation and Quasi-Monte Carlo Methods*, CBMS-NFS regional conference series in applied mathematics, vol. 63, SIAM, 1992.

[30] J. M. Rabaey: *Digital Integrated Circuits*, Prentice Hall, 1996.

[31] R. A. Rueppel: Linear complexity and random sequences, *Advances in Cryptology – EUROCRYPT '85* (F. Pichler, ed.), Lecture Notes in Computer Science, vol. 219, pp. 167–188, Springer-Verlag, 1985.

[32] A. Satoh, S. Morioka, K. Takano, and S. Munetoh: A Compact Rijndael Hardware Architecture with S-Box Optimization, *Advances in Cryptology – ASIACRYPT 2001*, (C. Boyd, ed.), Lecture Notes in Computer Science, vol. 2248, pp. 239-254, Springer-Verlag, 2001.

[33] E. S. Selmer: *Linear Recurrence Relations over Finite Fields*, Department of Mathematics, Univ. of Bergen, 1966.

[34] T. Siegenthaler: Correlation-immunity of nonlinear combining functions for cryptographic applications, *IEEE Trans. Inform. Theory* **IT-30**, 776-780 (1984).

[35] P. Voigtländer: Entwicklung einer Hardwarearchitektur für einen AES-Coprozessor, Ph.D.Thesis, Hochschule für Technik, Wirtschaft und Kultur Leipzig, Leipzig, Germany, 2003.

[36] E. A. Walker: Non-linear recursive sequences, *Can. J. Math.* **11**, 370–378 (1959).

[37] N. Zierler and W. H. Mills: Products of linear recurring sequences, *J. Algebra* **27**, 147-157 (1973).