ACHTERBAHN-128/80

Berndt Gammel, Rainer Göttfert, Oliver Kniffler

gammel@matpack.de
rainer.goettfert@gmx.de
oliver.kniffler@arcor.de

30 June 2006

Contents

1 Introduction

Achterbahn is an additive binary (synchronous) stream cipher. In an additive binary stream cipher encryption and decryption are realized by adding bitwise a binary pseudorandom sequence, called the *keystream*, to the plaintext string or ciphertext string, respectively. The binary pseudorandom sequence is produced by a finite state automaton, called the *keystream generator*, whose initial internal state depends on the secret key K and a public initial value IV.

The keystream generator of ACHTERBAHN-80 deploys eleven primitive binary nonlinear feedback shift registers. These shift registers are the nonlinear counterparts of the popular linear feedback shift registers with primitive characteristic polynomials. The keystream generator of ACHTERBAHN-128 deploys thirteen primitive nonlinear feedback shift registers and contains the keystream generator of ACHTERBAHN-80 as a substructure. As a consequence of this, ACHTERBAHN-128 is downward compatible and can produce the same keystream as ACHTERBAHN-80 if so desired—hence the name ACHTERBAHN-128/80.

The expected design strengths of ACHTERBAHN-80 and ACHTERBAHN-128 correspond to the key sizes of 80 bits and 128 bits, respectively. These are the maximum key lengths. However, all key lengths between 40 bits and 80 bits and between 40 bits and 128 bits, respectively, can be used, provided that the key lengths are divisible by eight. ACHTERBAHN-80 accommodates the IV-lengths $0, 8, 16, \ldots, 72, 80$. ACHTERBAHN-128 accommodates the IV-lengths $0, 8, 16, \ldots, 120, 128$. Shorter key or IV-lengths lead to shorter key-loading and resynchronization times. For example, instead of using ACHTERBAHN-128 with an 128-bit secret key K and an 128-bit initial value IV, in some applications it might be sufficient to use a 96-bit key and a 64-bit initial value. This would reduce the resynchronization time by approximately 30% or, to be more specific, from 337 clock cycles to 241 clock cycles in the 1-bit implementation, and from 43 clock cycles to 31 clock cycles in the 8-bit implementation of the keystream generator.

For a fixed (K, IV) pair the maximum amount of keystream that can be used for encryption, the so-called *frame length*, is 2^{64} bits for both variants of Achterbahn.

If this amount of keystream is consumed we can use a different initial value IV with the same key K to produce another 2^{64} bits of keystream. The process can be repeated until all initial values are exhausted. Only then it becomes necessary to change the secret key K. In practice, of course, the amount of keystream between resynchronization steps is much smaller than 2^{64} bits, and short resynchronization times are more important than long frame lengths. We collect the parameters of both Achterbahn variants in Table 1.

	ACHTERBAHN-80	**ACHTERBAHN-128**
Maximum key size	80 bit	128 bit
Maximum IV size	80 bit	128 bit
Internal state	297 bit	351 bit
Frame length	2^{64} bit	2^{64} bit

Table 1: Parameters of Achterbahn

We state that there are no hidden weaknesses in the proposed stream cipher that are inserted by the designers, nor are we aware of any weaknesses in the design.

ACHTERBAHN-128/80 is our contribution to the second phase of eSTREAM, a stream cipher project of ECRYPT NoE. Achterbahn-128/80 is a further development of our first stream cipher proposal, that was submitted to eSTREAM on April 28, 2005, and was called "The Achterbahn Stream Cipher" with no assigned version number [9]. If we refer to the stream cipher of the initial proposal we shall now use the name "Achterbahn-1".[1]

[1]In the literature [19] one can also find the names "Achterbahn-v2" and "Achterbahn-v3". They refer to modified variants of Achterbahn-1 which were not released by the designers [10]. A paper [15] dealing with a not fully specified predecessor version of Achterbahn-80 [11] containing an incorrect cryptanalysis has recently been posted at http://www.ecrypt.eu.org/stream/papers.html.

2 Preliminaries

In this section we discuss some basic facts that will be used throughout the proposal.

2.1 Notions and notation

We use $\mathbb{F}_2$ to denote the binary finite field which consists of two elements, 0 and 1. $\mathbb{F}_2[x]$ denotes the ring of polynomials in the indeterminate x and with coefficients from $\mathbb{F}_2$. The elements of $\mathbb{F}_2[x]$ are called *binary polynomials*. A sequence of elements $s_0, s_1, \ldots$ of $\mathbb{F}_2$ is called a *binary sequence* and is represented in the form $\sigma = (s_n)_{n=0}^{\infty}$. If $\sigma = (s_n)_{n=0}^{\infty}$ and $\tau = (t_n)_{n=0}^{\infty}$ are two binary sequences, then their sum $\sigma + \tau$ and product $\sigma\tau$ are defined termwise: $\sigma + \tau = (s_n + t_n)_{n=0}^{\infty}$ and $\sigma\tau = (s_n t_n)_{n=0}^{\infty}$. The scalar product of a binary sequence $\sigma = (s_n)_{n=0}^{\infty}$ and an element $c \in \mathbb{F}_2$ is defined by $c\sigma = (cs_n)_{n=0}^{\infty}$.

We use the symbol $\mathbb{F}_2^{\infty}$ to denote the set of all binary sequences. Endowed with the above defined addition and scalar multiplication $\mathbb{F}_2^{\infty}$ becomes a vector space over the field $\mathbb{F}_2$. We shall use the symbol $\mathbb{F}_2^{\infty}$ also to denote this vector space. A useful linear operator on $\mathbb{F}_2^{\infty}$ is the *shift operator* T, defined by $T\sigma = (s_{n+1})_{n=0}^{\infty}$ for all $\sigma \in \mathbb{F}_2^{\infty}$.

A sequence $\sigma = (s_n)_{n=0}^{\infty}$ is called *periodic* if there exists a positive integer p such that $s_{n+p} = s_n$ for all $n \geq 0$. The least positive integer with this property is called the *least period* of σ and we then write $p = \mathrm{per}(\sigma)$.

If g is a binary polynomial then $g(T)$ defines a linear operator on the vector space $\mathbb{F}_2^{\infty}$. If $\sigma \in \mathbb{F}_2^{\infty}$ is periodic, then there are infinitely many polynomials $g \in \mathbb{F}_2[x]$ such that $g(T)\sigma$ is the zero sequence $\mathbf{0} = (0, 0, \ldots)$. Each of these polynomials is called a *characteristic polynomial* of σ. For instance, $g(x) = x^{\mathrm{per}(\sigma)} - 1$ is a characteristic polynomial of σ. The binary characteristic polynomial of least degree is called the *minimal polynomial* of σ and denoted by m_σ. The minimal polynomial m_σ is uniquely determined by the sequence σ and has the property that it divides every characteristic polynomial of σ (see Appendix A.) The degree of m_σ is called the *linear complexity* of σ, denoted by $L(\sigma)$. The minimal polynomial of the zero sequence is the constant polynomial $m(x) = 1$ so that the linear complexity of the zero sequence is 0.

Let $\sigma = (s_n)_{n=0}^{\infty}$ be a nonzero periodic sequence in $\mathbb{F}_2^{\infty}$ with minimal polynomial

$$m_\sigma(x) = x^L + a_{L-1}x^{L-1} + \cdots + a_1 x + a_0.$$

Then $m_\sigma(T)\sigma = \mathbf{0}$ is just a condensed way of saying that

$$s_{n+L} = a_{L-1}s_{n+L-1} + \cdots + a_1 s_{n+1} + a_0 s_n \qquad \text{for all } n \geq 0.$$

That is, the terms of the sequence σ satisfy a *linear recurrence relation* of order L. The sequence σ is uniquely determined by the linear recurrence relation and the L *initial values* $s_0, s_1, \ldots, s_{L-1}$.

2.2 Feedback shift registers

Mathematically speaking, a binary N-stage feedback shift register (FSR) is a mapping Λ from $\mathbb{F}_2^N$ into $\mathbb{F}_2^N$ of the form

$$\Lambda : (x_0, x_1, \ldots, x_{N-1}) \mapsto (x_1, x_2, \ldots, x_{N-1}, A(x_0, x_1, \ldots, x_{N-1})). \tag{1}$$

The function $A : \mathbb{F}_2^N \to \mathbb{F}_2$ is called the *feedback function*. The shift register is called a *linear feedback shift register* (LFSR) if Λ is a linear transformation from the vector space $\mathbb{F}_2^N$ into itself. Otherwise the shift register is called a *nonlinear feedback shift register*. The shift register is called *nonsingular* if Λ is a bijection. All feedback shift registers used in the design of the proposed stream cipher are nonsingular and nonlinear.

Consider a binary sequence $\sigma = (s_n)_{n=0}^{\infty}$ whose first N terms $s_0, s_1, \ldots, s_{N-1}$ are given and whose remaining terms are uniquely determined by the recurrence relation

$$s_{n+N} = A(s_n, s_{n+1}, \ldots, s_{n+N-1}) \quad \text{for all } n \geq 0. \tag{2}$$

Then we call σ an output sequence of the FSR in (1). The binary N-tuple $\mathbf{s}_0 = (s_0, s_1, \ldots, s_{N-1})$ is referred to as the *initial state vector* of σ or the *initial state* of the FSR.

The recurrence relation (2) can be implemented in hardware as a special electronic switching circuit consisting of N memory cells (flip-flops) which we shall denote by $D_0, D_1, \ldots, D_{N-1}$. Each cell can store one binary element, 0 or 1. The shift register is regulated by an external clock. Assume that at the outset D_j contains the element s_j, $0 \leq j \leq N-1$. At each clock pulse, the content of D_j is shifted one position to the left into the next cell D_{j-1}, $1 \leq j \leq N-1$. The content of the left-most cell D_0 is emitted (or discarded) while the right-most cell D_{N-1} receives a new term computed by the feedback function $A(x_0, x_1, \ldots, x_{N-1})$ from the N previous terms. If at any clock pulse the content of D_0 is emitted, then the sequence $\sigma = (s_n)_{n=0}^{\infty}$ is produced.

Example 1. The feedback function

$$A(x_0, x_1, \ldots, x_4) = x_0 + x_1 + x_3 + x_1x_3$$

defines a binary 5-stage NLFSR. The shift register is shown in Figure 1.

If we start the shift register in the initial state $\mathbf{s}_0 = (0, 0, 0, 0, 1)$ and use D_0 as the output cell, then the shift register will generate a binary sequence σ of least period 31 given by

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

The sequence $\sigma = (s_n)_{n=0}^{\infty}$ is uniquely determined by the nonlinear recurrence relation

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

and the initial values $s_0 = s_1 = s_2 = s_3 = 0$, and $s_4 = 1$. If we compute the linear complexity of σ (e.g., with the Berlekamp-Massey algorithm), we find that $L(\sigma) = 30$, which means that σ could also be generated by a linear feedback shift register of length 30. □

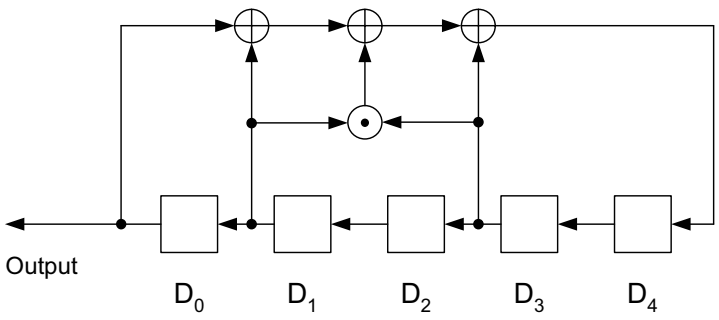


Figure 1: A binary nonlinear feedback shift register

2.3 Primitive feedback shift registers

Clearly the least period of an output sequence of a binary N-stage FSR cannot exceed 2^N. Nonsingular shift registers with output sequences of least period $2^N - 1$ are important for a variety of reasons: (i) they exist; (ii) their output sequences have good statistical properties; (iii) the output sequences have minimal polynomials with a simple algebraic structure.

Definition 1. A nonsingular binary N-stage feedback shift register is called *primitive* if for any nonzero binary initial state vector $\mathbf{s}_0 = (s_0, s_1, \ldots, s_{N-1})$, the corresponding output sequence $\sigma = (s_n)_{n=0}^{\infty}$ has least period $2^N - 1$.

If $A : \mathbb{F}_2^N \to \mathbb{F}_2$ is the feedback function of a primitive N-stage FSR, then we clearly must have $A(\mathbf{0}) = 0$. That is, a primitive FSR fixes the all-zero state. In other words, the zero sequence is an output sequence of any primitive FSR. Notice that the NLFSR considered in Example 1 is primitive.

Consider a binary N-stage LFSR with feedback function

$$F(x_0, x_1, \ldots, x_{N-1}) = a_{N-1}x_{N-1} + \cdots + a_1x_1 + a_0x_0.$$

The polynomial

$$f(x) = x^N + F(1, x, x^2, \ldots, x^{N-1}) = x^N + a_{N-1}x^{N-1} + \cdots + a_1x + a_0$$

is called the *characteristic polynomial* of the LFSR, and $f(x)$ is usually used to describe the LFSR (rather than the feedback function F). The LFSR is a primitive feedback shift register— in the sense of Definition 1— if and only if $f(x)$ is a primitive polynomial in $\mathbb{F}_2[x]$. This observation is the reason why we coined the name *primitive FSR* for the shift registers in Definition 1.

Two periodic sequences that are shifted versions of each other (such sequences are called *cyclically equivalent*) have the same minimal polynomial and, therefore, also the same least period and linear complexity. Any two nonzero output sequences of some primitive FSR are obviously cyclically equivalent so that the next definition makes sense.

Definition 2. Let A be a binary N-stage primitive FSR. The *minimal polynomial* of A, the *period* of A, and the *linear complexity* of A are defined to be the minimal polynomial, the least period, and the linear complexity of any nonzero output sequence of A, respectively.

We mention three basic facts concerning binary primitive FSR's.

Fact 1. *The number B_N of binary N-stage primitive FSR's is given by*

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

This was shown by Flye Sainte-Marie [4] in 1894 but the result went unnoticed for a long time until it was rediscovered by De Bruijn [2], [3].[2]

Although the number B_N is very large the portion of primitive FSR's among all binary nonsingular N-stage FSR's is only $1/2^N$. One must also take into account that most primitive FSR's are not suitable for a low-cost hardware implementation as their feedback functions are too complex.

Fact 2. *Let $\sigma = (s_n)_{n=0}^{\infty}$ be a nonzero output sequence of a binary primitive N-stage FSR. Let $1 \le k \le N$ and $\mathbf{b} = (b_1, \ldots, b_k) \in \mathbb{F}_2^k$. Let $Z(\mathbf{b})$ be the number of n in $\{0, 1, \ldots, 2^N - 2\}$ such that $(s_n, s_{n+1}, \ldots, s_{n+k-1}) = \mathbf{b}$. Then*

$$Z(\mathbf{b}) = \begin{cases} 2^{N-k} - 1 & \text{for } \boldsymbol{b} = \boldsymbol{0}, \\ 2^{N-k} & \text{for } \boldsymbol{b} \neq \boldsymbol{0}. \end{cases}$$

Proof. Since the binary sequence σ has least period $2^N - 1$ and is generated by an N-stage FSR which fixes the all-zero state, every nonzero binary N-tuple occurs precisely once in a full portion of the period of σ. From this, the assertion follows at once. □

Fact 3. *The minimal polynomial of a binary primitive N-stage FSR is the product of distinct irreducible binary polynomials whose degrees divide N and are greater than* 1.

Fact 3 is a corollary to the following lemma.

Lemma 1. *Let N be a positive integer, and let $\sigma = (s_n)_{n=0}^{\infty}$ be a binary periodic sequence with least period $2^N - 1$. The canonical factorization of the minimal polynomial of σ over $\mathbb{F}_2$ consists of distinct irreducible binary polynomials whose degrees all divide N. In particular, the minimal polynomial of σ contains no repeated factors. If σ is a nonzero output sequence of a binary primitive N-stage FSR, then the minimal polynomial of σ does not contain the factor $x - 1$.*

[2]Both, C. Flye Sainte-Marie and N. G. de Bruijn proved that B_N is equal to the number of complete cycles in a De Bruijn graph containing 2^{N-1} vertices (and 2^N edges)—or, equivalently, that B_N is the number of cyclically inequivalent binary De Bruijn sequences of span N. There is, however, an obvious one-to-one correspondence between binary De Bruijn sequences of span N and the nonzero output sequences of a binary primitive N-stage FSR. This correspondence then implies that the number of binary primitive N-stage FSR's coincides with B_N.

Proof. Set $p = 2^N - 1$. The binary polynomial $f(x) = x^p - 1$ is a characteristic polynomial of σ. The minimal polynomial $m_\sigma(x)$ of σ divides $f(x)$. Hence, $m_\sigma(x)$ divides $x^{2^N} - x$, which is the product of all irreducible binary polynomials whose degrees divide N (see [21, Theorem 3.20]). If $\sigma = (s_n)_{n=0}^{\infty}$ is any nonzero output sequence of a primitive binary N-stage FSR, then the N-tuples $(s_n, s_{n+1}, \ldots, s_{n+N-1})$, $0 \le n \le p-1$, run through all nonzero vectors of $\mathbb{F}_2^N$. The element 1 occurs exactly 2^{N-1} times among the first coordinates of these N-tuples. Hence, $s_0 + s_1 + \cdots + s_{p-1} = 0$. Since σ is periodic with $\mathrm{per}(\sigma) = p$, we conclude that

$$s_n + s_{n+1} + \cdots + s_{n+p-1} = 0 \quad \text{for all } n \ge 0,$$

which means that

$$f(x) = x^{p-1} + x^{p-2} + \cdots + x + 1$$

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

The actual computation of the minimal polynomial of a primitive binary FSR is increasingly difficult as the length N of the shift register gets larger. Using the Berlekamp-Massey algorithm [21, p. 439] or the formula of Laksov [20], it can be done up to shift register lengths $N = 25$.

Even if we cannot compute the minimal polynomial of a primitive FSR, we can determine for any given irreducible polynomial $f \in \mathbb{F}_2[x]$ whether or not f is a factor of the minimal polynomial. This can be done up to shift register lengths $N = 45$ using the following algorithm. The algorithm is proved in Appendix B.

For a binary sequence $\sigma = (s_n)_{n=0}^{\infty}$ and a positive integer d we use $\mathbf{s}_n^{(d)}$ to denote the d-tuple $(s_n, s_{n+1}, \ldots, s_{n+d-1})$.

ALGORITHM A:

Input: A binary primitive N-stage FSR (with unknown minimal polynomial $m(x)$). An irreducible polynomial $f \in \mathbb{F}_2[x]$ of degree $d \ge 2$ such that d divides N.

1. Compute the polynomial

$$q(x) = \frac{x^p - 1}{f(x)} = x^{p-d} + c_1 x^{p-d-1} + c_2 x^{p-d-2} + \cdots + c_{p-d-1} x + c_{p-d}\,,$$

   where $p = 2^N - 1$ and $f^*(x) = x^d f(1/x)$.

2. Choose an arbitrary nonzero initial state vector $\mathbf{s}_0 = (s_0, s_1, \ldots, s_{N-1}) \in \mathbb{F}_2^N$ and produce one full period $(s_0, s_1, \ldots, s_{p-1})$ of the FSR.

3. Compute the row vector $\mathbf{v} \in \mathbb{F}_2^d$ according to

$$\mathbf{v} = \mathbf{s}_0^{(d)} + c_1 \mathbf{s}_1^{(d)} + c_2 \mathbf{s}_2^{(d)} + \cdots + c_{p-d-1} \mathbf{s}_{p-d-1}^{(d)} + c_{p-d} \mathbf{s}_{p-d}^{(d)}\,.$$

Then, $f(x)$ divides $m(x)$ if and only if $\mathbf{v}$ is not the zero vector in $\mathbb{F}_2^d$.

Example 2. The feedback function $F(x_0, x_1, x_2, x_3) = x_0 + x_1 + x_2 + x_1x_3$ defines a binary primitive FSR of length $N = 4$. An output sequence of the shift register is

$$\sigma = (0\,0\,0\,1\,0\,1\,0\,0\,1\,1\,1\,1\,0\,1\,1)^{\infty}. \tag{3}$$

We want to know whether $f(x) = x^4 + x + 1$ is a factor of the minimal polynomial of the FSR (respectively of σ). We use long division to compute the quotient $q(x)$ of $x^{15} - 1$ and $f^*(x) = x^4 + x^3 + 1$. We obtain

$$q(x) = x^{11} + x^{10} + x^9 + x^8 + x^6 + x^4 + x^3 + 1.$$

Next we compute $\mathbf{v} \in \mathbb{F}_2^4$:

$$\mathbf{v} = \mathbf{s}_0 + \mathbf{s}_1 + \mathbf{s}_2 + \mathbf{s}_3 + \mathbf{s}_5 + \mathbf{s}_7 + \mathbf{s}_8 + \mathbf{s}_{11} = (0, 1, 1, 0).$$

Since $\mathbf{v} \neq \mathbf{0}$, the polynomial $f(x) = x^4 + x + 1$ is a divisor of the FSR's minimal polynomial. □

Example 3. We want to check whether $f(x) = x^2 + x + 1$ is a divisor of the minimal polynomial of the shift register considered in the preceding example. Here we have $f^*(x) = f(x)$, so that

$$q(x) = \frac{x^{15} - 1}{x^2 + x + 1} = x^{13} + x^{12} + x^{10} + x^9 + x^7 + x^6 + x^4 + x^3 + x + 1,$$

and

$$\mathbf{v} = \mathbf{s}_0^{(2)} + \mathbf{s}_1^{(2)} + \mathbf{s}_3^{(2)} + \mathbf{s}_4^{(2)} + \mathbf{s}_6^{(2)} + \mathbf{s}_7^{(2)} + \mathbf{s}_9^{(2)} + \mathbf{s}_{10}^{(2)} + \mathbf{s}_{12}^{(2)} + \mathbf{s}_{13}^{(2)} = (0, 0).$$

Thus, $f(x) = x^2 + x + 1$ is not a factor of the minimal polynomial of the FSR. If we compute the minimal polynomial $m(x)$ of the sequence σ in (3), using the Berlekamp-Massey algorithm or Proposition 3A in Appendix A, we obtain

$$m(x) = x^{12} + x^9 + x^6 + x^3 + 1 = (x^4 + x + 1)(x^4 + x^3 + 1)(x^4 + x^3 + x^2 + x + 1),$$

which is in accordance with the above results. □

A repeated application of Algorithm A can be used to assess the linear complexity of a binary primitive N-stage FSR.

ALGORITHM B:

Input: A binary primitive N-stage FSR.

1. Choose at random a polynomial f from the pool of all binary irreducible polynomials whose degrees divide N and are greater than 1.
2. Apply Algorithm A to determine whether or not f divides the minimal polynomial of the given FSR.
3. Repeat Steps 1 and 2 several times.

If in all the repetitions it turns out that the chosen polynomial f is indeed a factor of the minimal polynomial, then the linear complexity of the primitive N-stage FSR is greater than 2^{N-1} with high probability.

2.4 Hardware considerations

In this section we present some basic facts about hardware costs. Table 2 below contains a subset of logical gates taken from a standard cell library for 130 nm CMOS technology. The hardware costs are given units of *gate equivalents*. One gate equivalent (GE) is the area necessary to implement a 2-input NAND-gate on silicon.

Logical operation	**Binary function**	**Hardware cost**
NAND(a, b)	$ab + 1$	1.00 GE
NOR(a, b)	$1 + a + b + ab$	1.00 GE
AND(a, b)	ab	1.25 GE
OR(a, b)	$a + b + ab$	1.25 GE
XOR(a, b)	$a + b$	2.25 GE
NAND(a, b, c)	$abc + 1$	1.25 GE
NOR(a, b, c)	$1 + a + b + c + ab + ac + bc + abc$	1.50 GE
AND(a, b, c)	abc	1.50 GE
OR(a, b, c)	$a + b + c + ab + ac + bc + abc$	1.75 GE
XOR(a, b, c)	$a + b + c$	4.00 GE
MAJ(a, b, c)	$ab + ac + bc$	2.25 GE
MUX$(a, b; c)$	$a + ac + bc$	2.50 GE

Table 2: Hardware costs of logical operations

Reconsider the feedback shift register of Example 1. If we implement the feedback function

$$A(x_0, x_1, \ldots, x_4) = x_0 + x_1 + x_3 + x_1x_3 \tag{4}$$

as depicted in Figure 1, we need three 2-input XOR-gates (3×2.25 GE) and one 2-input AND-gate (1.25 GE). The implementation costs of the feedback function are then 8 GE. A better way to implement the feedback function would be to use one 2-input OR-gate (1.25 GE) plus one 2-input XOR-gate (2.25 GE). The implementation costs are then reduced to 3.5 GE. In fact, $\mathrm{OR}(a, b) = a \vee b = a + b + ab$ for $a, b \in \mathbb{F}_2$, so that (4) is equivalent to

$$A(x_0, x_1, \ldots, x_4) = x_0 + (x_1 \vee x_3).$$

The second implementation is preferable also because it has a lower *logical depth*. The logical depth of the first implementation is three while the logical depth of the second implementation is only two.

When implementing a FSR on hardware a considerable amount of area will be used up for the implementation of the memory cells. We can distinguish three types of flip-flops. The simplest and least expensive flip-flop (4.75 GE) is a flip-flop without

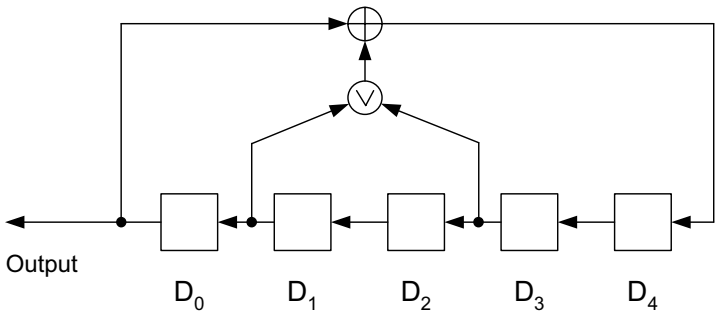


Figure 2: A more efficient implementation of the FSR of Example 1

reset functionality. The more expensive *scan flip-flop* (6.75 GE) is a flip-flop with an integrated multiplexor. The first two flip-flops in Table 3 have one data input and one data output while a scan flip-flop has two data inputs and one data output.

Memory unit	**Hardware costs**
Flip-flop	4.75 GE
Reset flip-flop	5.75 GE
Scan flip-flop	6.75 GE

Table 3: Hardware costs of memory units

The flip-flops used in the keystream generator of Achterbahn do not need to have reset functionality. There is no need to reset a flip-flop at any time during key-loading or resynchronization. In the first phase of the key-loading algorithm, each flip-flop of any single FSR is loaded with a key bit. If the shift register has length N, then it receives the first N bits of the secret key. We can introduce the key bits into the flip-flops $D_0, D_1, \ldots, D_{N-1}$ in two different ways: Bit after bit, or several key bits at once. The second method provides a higher resistance to simple power analysis attacks but it also causes higher hardware costs—since we then have to implement scan flip-flops or additional multiplexors.

3 Specification

In this chapter, we describe the keystream generator of ACHTERBAHN-128/80 and specify its main ingredients: The Boolean combining function(s) and the primitive feedback shift registers. We shall mention some cryptographic properties of the combining and feedback functions. Finally, we shall describe the key-loading algorithm.

3.1 The keystream generator

The keystream generator of ACHTERBAHN-128 consists of thirteen binary primitive nonlinear feedback shift registers of lengths between 21 and 33 and a Boolean combining function $F : \mathbb{F}_2^{13} \to \mathbb{F}_2$. The function F combines the output sequences of the thirteen feedback shift registers to produce the keystream $\zeta = (z_0, z_1, \dots)$. Throughout this proposal we shall use the capital letters A_j, $j = 0, 1, \dots, 12$, to designate the primitive FSR's and, in a slight abuse of notation, also to designate the feedback functions of the shift registers. The length of the shift register A_j is denoted by N_j. We have

$$N_j = 21 + j \quad \text{for } j = 0, 1, \dots, 12.$$

Any nonzero output sequence of the shift register A_j will be denoted by σ_j, $0 \le j \le 12$. The minimal polynomial, the period, and the linear complexity of the shift register A_j (in the sense of Definition 2) will be denoted by m_j, p_j, and L_j, respectively. We have

$$p_j = \mathrm{per}(\sigma_j) = 2^{N_j} - 1 \quad \text{for } j = 0, 1, \dots, 12.$$

Let the initial state of the shift register A_j prior to encryption be given by the row vector

$$\mathbf{r}_0 = (r_0, r_1, \dots, r_{N_j-1}) \in \mathbb{F}_2^{N_j}.$$

The row vector $\mathbf{r}_0$ is derived from the secret key K and the initial value IV using the key-loading algorithm to be described in Section 3.5. The key-loading algorithm ensures that $\mathbf{r}_0$ will not be the zero vector no matter which (K, IV) pair is used for initialization.

The standard output sequence of the shift register A_j is defined to be the binary sequence $\rho = (r_n)_{n=0}^{\infty}$, defined by

$$r_{n+N_j} = A_j(r_n, r_{n+1}, \dots, r_{n+N_j-1}) \quad \text{for } n = 0, 1, \dots .$$

We shall, however, not use the sequence ρ as an input sequence to the Boolean combining function but rather a shifted version of ρ. We shall use the sequence $\sigma_j = T^{N_j-16}\rho$, where T is the shift operator on $\mathbb{F}_2^{\infty}$. In other words, we use the sequence $\sigma_j = (s_n^{(j)})_{n=0}^{\infty}$ with $s_n^{(j)} = r_{n+N_j-16}$ for $n \ge 0$ as the jth input sequence to the Boolean combining function and we shall call this sequence the *output sequence* of the shift register A_j.

Let for $0 \le j \le 12$, $\sigma_j = (s_n^{(j)})_{n=0}^{\infty}$ be the output sequence of the shift register A_j. Then the keystream $\zeta = (z_n)_{n=0}^{\infty}$ of ACHTERBAHN-128 is defined by

$$z_n = F(s_n^{(0)}, s_n^{(1)}, \dots, s_n^{(12)}) \quad \text{for } n = 0, 1, \dots . \tag{5}$$

Instead of (5) we shall use in the sequel the more compact notation

$$\zeta = F(\sigma_0, \sigma_1, \ldots, \sigma_{12}).$$

The keystream generator of ACHTERBAHN-80 consists of the eleven shift registers $A_1, \ldots, A_{11}$, that are also used in the keystream generator of ACHTERBAHN-128, and has a Boolean combining function $G : \mathbb{F}_2^{11} \to \mathbb{F}_2$ which is a subfunction of of the combining function of ACHTERBAHN-128:

$$G(x_1, \ldots, x_{11}) = F(0, x_1, \ldots, x_{11}, 0).$$

In other words, the keystream generator of ACHTERBAHN-128 contains the keystream generator of ACHTERBAHN-80 as a substructure. The design is reminiscent to a set of Russian dolls.

ACHTERBAHN-128 is downward compatible to ACHTERBAHN-80: If we load the first and the last shift register (i.e, the shift registers A_0 and A_{12}) into the all-zero state, and if we load the remaining shift registers with the same key and initial value as ACHTERBAHN-80, then ACHTERBAHN-128 will produce the same keystream as ACHTERBAHN-80.

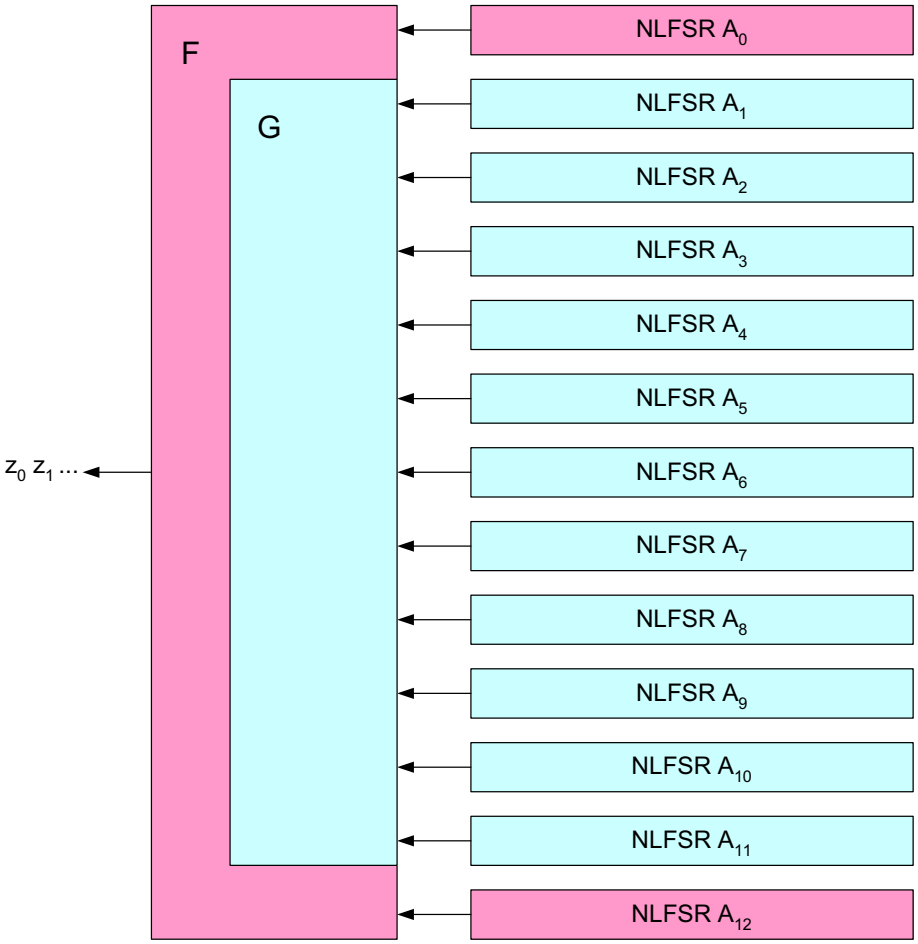


Figure 3: The keystream generator of ACHTERBAHN-128/80

What we have described so far is the 1-bit implementation of Achterbahn. In this implementation, 1 bit of keystream is generated per clock cycle. A requirement imposed on the shift registers $A_0, A_1, \ldots, A_{12}$ was that they should facilitate parallel implementations of the keystream generator. In Section 6.1, we shall describe 2-bit, 4-bit, and 8-bit implementations of Achterbahn. In an 8-bit implementation, 1 byte of keystream is generated per clock cycle. It is then possible to supply in real time all eight (=acht) lines (=Bahnen) of a bus with keystream bits. This feature was the reason for choosing the name ACHTERBAHN. The plain translation of the German word *Achterbahn* is roller coaster.

3.2 The combining function of ACHTERBAHN-128

The algebraic normal form of the Boolean combining function of ACHTERBAHN-128 is given by

$$\begin{aligned}
F(x_0, x_1, \ldots, x_{12}) = {} & x_0 + x_1 + x_2 + x_3 + x_4 + x_5 + x_7 + x_9 + x_{11} + x_{12} + x_0x_5 \\
& + x_2x_{10} + x_2x_{11} + x_4x_8 + x_4x_{12} + x_5x_6 + x_6x_8 + x_6x_{10} + x_6x_{11} \\
& + x_6x_{12} + x_7x_8 + x_7x_{12} + x_8x_9 + x_8x_{10} + x_9x_{10} + x_9x_{11} + x_9x_{12} \\
& + x_{10}x_{12} + x_0x_5x_8 + x_0x_5x_{10} + x_0x_5x_{11} + x_0x_5x_{12} + x_1x_2x_8 \\
& + x_1x_2x_{12} + x_1x_4x_{10} + x_1x_4x_{11} + x_1x_8x_9 + x_1x_9x_{10} + x_1x_9x_{11} \\
& + x_1x_9x_{12} + x_2x_3x_8 + x_2x_3x_{12} + x_2x_4x_8 + x_2x_4x_{10} + x_2x_4x_{11} \\
& + x_2x_4x_{12} + x_2x_7x_8 + x_2x_7x_{12} + x_2x_8x_{10} + x_2x_8x_{11} + x_2x_9x_{10} \\
& + x_2x_9x_{11} + x_2x_{10}x_{12} + x_2x_{11}x_{12} + x_3x_4x_8 + x_3x_4x_{12} + x_3x_8x_9 \\
& + x_3x_9x_{12} + x_4x_7x_8 + x_4x_7x_{12} + x_4x_8x_9 + x_4x_9x_{12} + x_5x_6x_8 \\
& + x_5x_6x_{10} + x_5x_6x_{11} + x_5x_6x_{12} + x_6x_8x_{10} + x_6x_8x_{11} + x_6x_{10}x_{12} \\
& + x_6x_{11}x_{12} + x_7x_8x_9 + x_7x_9x_{12} + x_8x_9x_{10} + x_8x_9x_{11} + x_9x_{10}x_{12} \\
& + x_9x_{11}x_{12} + x_0x_5x_8x_{10} + x_0x_5x_8x_{11} + x_0x_5x_{10}x_{12} + x_0x_5x_{11}x_{12} \\
& + x_1x_2x_3x_8 + x_1x_2x_3x_{12} + x_1x_2x_7x_8 + x_1x_2x_7x_{12} + x_1x_3x_5x_8 \\
& + x_1x_3x_5x_{12} + x_1x_3x_8x_9 + x_1x_3x_9x_{12} + x_1x_4x_8x_{10} + x_1x_4x_8x_{11} \\
& + x_1x_4x_{10}x_{12} + x_1x_4x_{11}x_{12} + x_1x_5x_7x_8 + x_1x_5x_7x_{12} + x_1x_7x_8x_9 \\
& + x_1x_7x_9x_{12} + x_1x_8x_9x_{10} + x_1x_8x_9x_{11} + x_1x_9x_{10}x_{12} + x_1x_9x_{11}x_{12} \\
& + x_2x_3x_4x_8 + x_2x_3x_4x_{12} + x_2x_3x_5x_8 + x_2x_3x_5x_{12} + x_2x_4x_7x_8 \\
& + x_2x_4x_7x_{12} + x_2x_4x_8x_{10} + x_2x_4x_8x_{11} + x_2x_4x_{10}x_{12} + x_2x_4x_{11}x_{12} \\
& + x_2x_5x_7x_8 + x_2x_5x_7x_{12} + x_2x_8x_9x_{10} + x_2x_8x_9x_{11} + x_2x_9x_{10}x_{12} \\
& + x_2x_9x_{11}x_{12} + x_3x_4x_8x_9 + x_3x_4x_9x_{12} + x_4x_7x_8x_9 + x_4x_7x_9x_{12} \\
& + x_5x_6x_8x_{10} + x_5x_6x_8x_{11} + x_5x_6x_{10}x_{12} + x_5x_6x_{11}x_{12}.
\end{aligned}$$

The combining function F has the following properties:

(i) F is balanced;
(ii) F has algebraic degree 4;
(iii) F is correlation immune of order 8;
(iv) F has nonlinearity 3584;
(v) F has algebraic immunity 4;

(vi) Each variable of F appears in at least one monomial of degree 4 such that the shift register lengths corresponding to the variables in that monomial are pairwise relatively prime.
(vii) F has an efficient hardware implementation: 68 GE.

In the construction of the Boolean function F we made use of outstanding results achieved by Tarannikov [33].

The above mentioned properties have the following bearing.

ad (i) A Boolean function f in n variables is called *balanced* if $f(\mathbf{a}) = 1$ for exactly 2^{n-1} vectors $\mathbf{a} \in \mathbb{F}_2^n$. Equivalently, if $X_1, \ldots, X_n$ are independent symmetrically distributed binary random variables, then the random variable $Z = f(X_1, \ldots, X_n)$ satisfies

$$\Pr(Z = 1) = \frac{1}{2}.$$

ad (ii) According to Siegenthaler [32] the algebraic degree d of a balanced Boolean function of n variables having order of correlation immunity c can be at most $d = n-c-1$. Boolean functions satisfying this equation have been called *optimized* [22]. Only optimized Boolean functions can have maximum nonlinearity [33]. Here we have $d = 4$, $n = 13$, and $c = 8$. Thus F is optimized.

ad (iii) A Boolean function f in n variables is *correlation immune* of order c if knowing the value of any c of the input variables of f gives no additional information on the value of the output of f (see [32], [6]). Equivalently, if $X_1, \ldots, X_n$ are independent symmetrically distributed binary random variables and $Z = f(X_1, \ldots, X_n)$, then for any choice of $k \leq c$ random variables $X_{i_1}, \ldots, X_{i_k}$ we have

$$\Pr(X_{i_1} = e_1, \ldots, X_{i_k} = e_k \mid Z = 1) = \frac{1}{2^k}$$

for all $(e_1, \ldots, e_k) \in \{0,1\}^k$. High order of correlation immunity is important to counter correlation attacks.

ad (iv) The *Hamming distance* $d(f, g)$ between two Boolean functions f and g in n variables is defined by

$$d(f, g) = \mid \{\mathbf{a} \in \mathbb{F}_2^n : f(\mathbf{a}) \neq g(\mathbf{a})\} \mid .$$

The nonlinearity of f is defined by

$$\mathrm{NL}(f) = \min_l d(f, l),$$

where l runs through all the 2^{n+1} affine functions in n variables. The maximum possible value of the nonlinearity of a balanced Boolean function of n variables having order of correlation immunity c is $2^{n-1} - 2^{c+1}$ for $(2n-7)/3 \leq c \leq n-2$ according to Tarannikov [33]. Here we have $n = 13$ and $c = 8$. Thus, the nonlinearity of F is maximum. A high nonlinearity of the combining function is necessary to counter correlation attacks.

ad (v) A Boolean function f in n variables of degree $d \geq 1$ has algebraic immunity b if for each nonzero Boolean function h in n variables of degree less than b there are $\mathbf{a}_1, \mathbf{a}_2 \in \mathbb{F}_2^n$ with

$$f(\mathbf{a}_1)h(\mathbf{a}_1) \neq 0 \quad \text{and} \quad (f(\mathbf{a}_2)+1)h(\mathbf{a}_2) \neq 0,$$

and b is the greatest integer having this property. Equivalently, if we regard f as an element of the ring $R = \mathbb{F}_2[x_1, \ldots, x_n]/(x_1^2 - x_1, \ldots, x_n^2 - x_n)$, then f has algebraic immunity b if there are no nonzero polynomials $h_1, h_2 \in R$ of degree less than b with

$$f * h_1 = \mathbf{0} \quad \text{or} \quad (f+1) * h_2 = \mathbf{0},$$

where $*$ denotes the multiplication rule in R and $\mathbf{0}$ is the zero element of R, and where b is the greatest integer for which this is true (see [24]). The algebraic immunity b of a Boolean function f can never exceed the degree of the function. This is obvious by considering the function $h = f + 1$, which has the same degree as f and which satisfies $f * h = \mathbf{0}$ in R. Here we have $d = b = 4$. Thus F has maximum algebraic immunity. The algebraic immunity 4 of F is not really necessary. In order to counter algebraic attacks it would be sufficient if the combining function F had algebraic immunity 2.

ad (vi) This property is necessary in order to be able to prove a robustness property of the keystream (see Section 4.3).

3.3 The combining function of ACHTERBAHN-80

The combining function G of ACHTERBAHN-80 is obtained from the combining function F of ACHTERBAHN-128 by setting the first variable x_0 and the last variable x_{12} of F to zero:

$$G(x_1, \ldots, x_{11}) = F(0, x_1, \ldots, x_{11}, 0).$$

The algebraic normal form of G reads

$$\begin{aligned}
G(x_1, x_2, \ldots, x_{11}) = {} & x_1 + x_2 + x_3 + x_4 + x_5 + x_7 + x_9 + x_{11} + x_2x_{10} + x_2x_{11} \\
& + x_4x_8 + x_5x_6 + x_6x_8 + x_6x_{10} + x_6x_{11} + x_7x_8 + x_8x_9 + x_8x_{10} \\
& + x_9x_{10} + x_9x_{11} + x_1x_2x_8 + x_1x_4x_{10} + x_1x_4x_{11} + x_1x_8x_9 \\
& + x_1x_9x_{10} + x_1x_9x_{11} + x_2x_3x_8 + x_2x_4x_8 + x_2x_4x_{10} + x_2x_4x_{11} \\
& + x_2x_7x_8 + x_2x_8x_{10} + x_2x_8x_{11} + x_2x_9x_{10} + x_2x_9x_{11} + x_3x_4x_8 \\
& + x_3x_8x_9 + x_4x_7x_8 + x_4x_8x_9 + x_5x_6x_8 + x_5x_6x_{10} + x_5x_6x_{11} \\
& + x_6x_8x_{10} + x_6x_8x_{11} + x_7x_8x_9 + x_8x_9x_{10} + x_8x_9x_{11} + x_1x_2x_3x_8 \\
& + x_1x_2x_7x_8 + x_1x_3x_5x_8 + x_1x_3x_8x_9 + x_1x_4x_8x_{10} + x_1x_4x_8x_{11} \\
& + x_1x_5x_7x_8 + x_1x_7x_8x_9 + x_1x_8x_9x_{10} + x_1x_8x_9x_{11} + x_2x_3x_4x_8 \\
& + x_2x_3x_5x_8 + x_2x_4x_7x_8 + x_2x_4x_8x_{10} + x_2x_4x_8x_{11} + x_2x_5x_7x_8 \\
& + x_2x_8x_9x_{10} + x_2x_8x_9x_{11} + x_3x_4x_8x_9 + x_4x_7x_8x_9 + x_5x_6x_8x_{10} \\
& + x_5x_6x_8x_{11}.
\end{aligned}$$

The combining function G has the following properties:

(i) G is balanced;
(ii) G has algebraic degree 4;
(iii) G is correlation immune of order 6;
(iv) G has maximum nonlinearity 896;
(v) G has maximum algebraic immunity 4;
(vi) Each variable of G appears in at least one monomial of degree 4 such that the shift register lengths corresponding to the variables in that monomial are pairwise relatively prime.

3.4 The feedback shift registers

All feedback shift registers deployed in the keystream generator of ACHTERBAHN-128/80 are primitive and nonlinear. Each shift register is described by its feedback function. We first give a description of the feedback functions in terms of logical gates whose definition and hardware costs can be found in Table 2 in the Preliminaries. The representations show that each feedback function can be implemented with logical depth three using 2-input gates and 3-input gates only.

$$\begin{aligned}A_0(x_0, x_1, \ldots, x_{20}) = \mathrm{XOR}(&\mathrm{XOR}(x_{15}, \mathrm{XOR}(x_0, x_2, x_3)),\\ &\mathrm{XOR}(\mathrm{AND}(x_4, x_7), \mathrm{XOR}(x_5, x_6, x_8), \mathrm{MUX}(x_4, x_5; x_6)),\\ &\mathrm{MUX}(\mathrm{MUX}(x_{11}, x_{12}; x_2), \mathrm{AND}(x_2, x_6, x_{13}); \mathrm{MUX}(x_1, x_{10}; x_9)));\end{aligned}$$

$$\begin{aligned}A_1(x_0, x_1, \ldots, x_{21}) = \mathrm{XOR}(&\mathrm{XOR}(x_{15}, \mathrm{XOR}(x_0, x_5, x_8)),\\ &\mathrm{XOR}(\mathrm{AND}(x_5, x_{11}), \mathrm{MUX}(x_{13}, x_3; x_1), \mathrm{MUX}(x_6, x_4; x_{12})),\\ &\mathrm{MUX}(\mathrm{MUX}(x_1, x_9; x_7), \mathrm{MUX}(x_4, x_{12}; x_{10}); \mathrm{AND}(x_1, x_{11}, x_{14})));\end{aligned}$$

$$\begin{aligned}A_2(x_0, x_1, \ldots, x_{22}) = \mathrm{XOR}(&\mathrm{XOR}(x_{16}, \mathrm{XOR}(x_0, x_4, x_{13})),\\ &\mathrm{XOR}(\mathrm{AND}(x_{12}, x_{14}), \mathrm{MUX}(x_1, x_9; x_7), \mathrm{MUX}(x_1, x_4; x_6)),\\ &\mathrm{MUX}(\mathrm{MUX}(x_5, x_8; x_{11}), \mathrm{MUX}(x_{10}, x_3; x_{11}); \mathrm{AND}(x_1, x_9, x_{15})));\end{aligned}$$

$$\begin{aligned}A_3(x_0, x_1, \ldots, x_{23}) = \mathrm{XOR}(&\mathrm{XOR}(x_{18}, \mathrm{XOR}(x_0, x_3, x_8)),\\ &\mathrm{XOR}(\mathrm{AND}(x_1, x_{11}), \mathrm{MUX}(x_2, x_{14}; x_{13}), \mathrm{MUX}(x_{12}, x_4; x_{13})),\\ &\mathrm{MUX}(\mathrm{MUX}(x_6, x_1; x_{15}), \mathrm{MUX}(x_{14}, x_{16}; x_9); \mathrm{MAJ}(x_2, x_5, x_7)));\end{aligned}$$

$$\begin{aligned}A_4(x_0, x_1, \ldots, x_{24}) = \mathrm{XOR}(&\mathrm{XOR}(x_{20}, \mathrm{XOR}(x_0, x_1, x_{11})),\\ &\mathrm{XOR}(\mathrm{AND}(x_4, x_{12}), \mathrm{MUX}(x_1, x_3; x_5), \mathrm{MUX}(x_6, x_7; x_{16})),\\ &\mathrm{MUX}(\mathrm{MAJ}(x_8, x_{15}, x_{17}), \mathrm{MUX}(x_{14}, x_{13}; x_{12}); \mathrm{MUX}(x_5, x_3; x_2)));\end{aligned}$$

$$\begin{aligned}A_5(x_0, x_1, \ldots, x_{25}) = \mathrm{XOR}(&\mathrm{XOR}(x_{21}, \mathrm{XOR}(x_{15}, x_{16}, x_{17})),\\ &\mathrm{XOR}(\mathrm{XOR}(x_0, x_4, x_5), \mathrm{AND}(x_3, x_6), \mathrm{MUX}(x_4, x_{18}; x_2)),\\ &\mathrm{MUX}(\mathrm{MUX}(x_4, x_{12}; x_{13}), \mathrm{MUX}(x_{14}, x_{11}; x_7); \mathrm{MAJ}(x_3, x_{10}, x_{15})));\end{aligned}$$

$$\begin{aligned}A_6(x_0, x_1, \ldots, x_{26}) = \mathrm{XOR}(&\mathrm{XOR}(x_{25}, \mathrm{XOR}(x_0, x_4, x_{15})),\\ &\mathrm{XOR}(\mathrm{AND}(x_1, x_{12}), \mathrm{MUX}(x_{10}, x_6; x_{17}), \mathrm{MUX}(x_3, x_8; x_1)),\\ &\mathrm{MUX}(\mathrm{MUX}(x_{10}, x_{14}; x_{13}), \mathrm{MAJ}(x_2, x_{16}, x_{17}); \mathrm{AND}(x_5, x_{11}, x_{18})));\end{aligned}$$

$$A_7(x_0, x_1, \ldots, x_{27}) = \mathrm{XOR}(\mathrm{XOR}(x_{25}, \mathrm{XOR}(x_0, x_5, x_{18})),$$
$$\mathrm{XOR}(\mathrm{AND}(x_4, x_{12}), \mathrm{MUX}(x_1, x_{17}; x_2), \mathrm{MUX}(x_{20}, x_{14}; x_{16})),$$
$$\mathrm{MUX}(\mathrm{MUX}(x_{18}, x_{15}; x_{10}), \mathrm{AND}(x_1, x_2, x_{13}); \mathrm{AND}(x_7, x_9, x_{19})));$$

$$A_8(x_0, x_1, \ldots, x_{28}) = \mathrm{XOR}(\mathrm{XOR}(x_{24}, \mathrm{XOR}(x_{17}, x_{18}, x_{21})),$$
$$\mathrm{XOR}(\mathrm{AND}(x_1, x_4), \mathrm{XOR}(x_0, x_2, x_{11}), \mathrm{MUX}(x_{10}, x_8; x_{21})),$$
$$\mathrm{MUX}(\mathrm{AND}(x_8, x_9, x_{18}), \mathrm{MUX}(x_{13}, x_6; x_{15}); \mathrm{MUX}(x_{19}, x_{16}; x_{14})));$$

$$A_9(x_0, x_1, \ldots, x_{29}) = \mathrm{XOR}(\mathrm{XOR}(x_{28}, \mathrm{XOR}(x_0, x_1, x_{18})),$$
$$\mathrm{XOR}(\mathrm{AND}(x_2, x_8), \mathrm{MUX}(x_{12}, x_{19}; x_{10}), \mathrm{MUX}(x_{10}, x_{14}; x_{22})),$$
$$\mathrm{MUX}(\mathrm{MUX}(x_7, x_{18}; x_4), \mathrm{MAJ}(x_1, x_9, x_{21}); \mathrm{MAJ}(x_3, x_5, x_8)));$$

$$A_{10}(x_0, x_1, \ldots, x_{30}) = \mathrm{XOR}(\mathrm{XOR}(x_{25}, \mathrm{XOR}(x_6, x_{15}, x_{18})),$$
$$\mathrm{XOR}(\mathrm{XOR}(x_0, x_2, x_5), \mathrm{AND}(x_{14}, x_{19}), \mathrm{MUX}(x_{17}, x_{12}; x_{21})),$$
$$\mathrm{MUX}(\mathrm{MUX}(x_{20}, x_{18}; x_8), \mathrm{MAJ}(x_4, x_{12}, x_{19}); \mathrm{MUX}(x_{22}, x_7; x_{21})));$$

$$A_{11}(x_0, x_1, \ldots, x_{31}) = \mathrm{XOR}(\mathrm{XOR}(x_{28}, \mathrm{XOR}(x_8, x_{17}, x_{22})),$$
$$\mathrm{XOR}(\mathrm{AND}(x_{13}, x_{15}), \mathrm{XOR}(x_0, x_3, x_5), \mathrm{MUX}(x_5, x_7; x_{19})),$$
$$\mathrm{MUX}(\mathrm{MUX}(x_8, x_2; x_{13}), \mathrm{AND}(x_4, x_{11}, x_{24}); \mathrm{MUX}(x_{12}, x_{14}; x_7)));$$

$$A_{12}(x_0, x_1, \ldots, x_{32}) = \mathrm{XOR}(\mathrm{XOR}(x_{30}, \mathrm{XOR}(x_9, x_{10}, x_{23})),$$
$$\mathrm{XOR}(\mathrm{XOR}(x_0, x_2, x_7), \mathrm{AND}(x_{15}, x_{16}), \mathrm{MUX}(x_{25}, x_{15}; x_{13})),$$
$$\mathrm{MUX}(\mathrm{MUX}(x_{15}, x_{12}; x_{16}), \mathrm{MAJ}(x_1, x_{14}, x_{18}); \mathrm{MUX}(x_8, x_{24}; x_{17}))).$$

The algebraic normal forms of the feedback functions are given by:

$$A_0(x_0, x_1, \ldots, x_{20}) = x_0 + x_2 + x_3 + x_4 + x_5 + x_6 + x_8 + x_{11} + x_{15} + x_1x_{11}$$
$$+ x_2x_{11} + x_2x_{12} + x_4x_6 + x_4x_7 + x_5x_6 + x_1x_2x_{11} + x_1x_2x_{12}$$
$$+ x_1x_9x_{11} + x_9x_{10}x_{11} + x_1x_2x_6x_{13} + x_1x_2x_9x_{11} + x_1x_2x_9x_{12}$$
$$+ x_2x_9x_{10}x_{11} + x_2x_9x_{10}x_{12} + x_1x_2x_6x_9x_{13} + x_2x_6x_9x_{10}x_{13};$$

$$A_1(x_0, x_1, \ldots, x_{21}) = x_0 + x_1 + x_5 + x_6 + x_8 + x_{13} + x_{15} + x_1x_3 + x_1x_7$$
$$+ x_1x_{13} + x_4x_{12} + x_5x_{11} + x_6x_{12} + x_7x_9 + x_1x_{11}x_{14}$$
$$+ x_1x_4x_{11}x_{14} + x_1x_7x_{11}x_{14} + x_1x_4x_{10}x_{11}x_{14} + x_1x_7x_9x_{11}x_{14}$$
$$+ x_1x_{10}x_{11}x_{12}x_{14};$$

$$A_2(x_0, x_1, \ldots, x_{22}) = x_0 + x_4 + x_5 + x_{13} + x_{16} + x_1x_6 + x_1x_7 + x_4x_6 + x_5x_{11}$$
$$+ x_7x_9 + x_8x_{11} + x_{12}x_{14} + x_1x_5x_9x_{15} + x_1x_9x_{10}x_{15} + x_1x_3x_9x_{11}x_{15}$$
$$+ x_1x_5x_9x_{11}x_{15} + x_1x_8x_9x_{11}x_{15} + x_1x_9x_{10}x_{11}x_{15};$$

$$A_3(x_0, x_1, \ldots, x_{23}) = x_0 + x_2 + x_3 + x_6 + x_8 + x_{12} + x_{18} + x_1x_{11} + x_1x_{15}$$
$$+ x_2x_{13} + x_4x_{13} + x_6x_{15} + x_{12}x_{13} + x_{13}x_{14} + x_2x_5x_6 + x_2x_5x_{14}$$
$$+ x_2x_6x_7 + x_2x_7x_{14} + x_5x_6x_7 + x_5x_7x_{14} + x_1x_2x_5x_{15} + x_1x_2x_7x_{15}$$
$$+ x_1x_5x_7x_{15} + x_2x_5x_6x_{15} + x_2x_5x_9x_{14} + x_2x_5x_9x_{16} + x_2x_6x_7x_{15}$$
$$+ x_2x_7x_9x_{14} + x_2x_7x_9x_{16} + x_5x_6x_7x_{15} + x_5x_7x_9x_{14} + x_5x_7x_9x_{16};$$

$$A_4(x_0,x_1,\ldots,x_{24}) = x_0 + x_6 + x_{11} + x_{20} + x_1x_5 + x_3x_5 + x_4x_{12} + x_5x_{14} + x_6x_{16} + x_7x_{16} + x_8x_{15} + x_8x_{17} + x_{15}x_{17} + x_2x_3x_{14} + x_2x_5x_{14} + x_5x_8x_{15} + x_5x_8x_{17} + x_5x_{12}x_{13} + x_5x_{12}x_{14} + x_5x_{15}x_{17} + x_2x_3x_8x_{15} + x_2x_3x_8x_{17} + x_2x_3x_{12}x_{13} + x_2x_3x_{12}x_{14} + x_2x_3x_{15}x_{17} + x_2x_5x_8x_{15} + x_2x_5x_8x_{17} + x_2x_5x_{12}x_{13} + x_2x_5x_{12}x_{14} + x_2x_5x_{15}x_{17};$$

$$A_5(x_0,x_1,\ldots,x_{25}) = x_0 + x_4 + x_5 + x_{15} + x_{16} + x_{17} + x_{21} + x_2x_4 + x_2x_{18} + x_3x_6 + x_4x_{13} + x_{12}x_{13} + x_3x_4x_{10} + x_3x_4x_{15} + x_3x_{10}x_{14} + x_3x_{14}x_{15} + x_4x_{10}x_{15} + x_{10}x_{14}x_{15} + x_3x_4x_{10}x_{13} + x_3x_4x_{13}x_{15} + x_3x_7x_{10}x_{11} + x_3x_7x_{10}x_{14} + x_3x_7x_{11}x_{15} + x_3x_7x_{14}x_{15} + x_3x_{10}x_{12}x_{13} + x_3x_{12}x_{13}x_{15} + x_4x_{10}x_{13}x_{15} + x_7x_{10}x_{11}x_{15} + x_7x_{10}x_{14}x_{15} + x_{10}x_{12}x_{13}x_{15};$$

$$A_6(x_0,x_1,\ldots,x_{26}) = x_0 + x_3 + x_4 + x_{15} + x_{25} + x_1x_3 + x_1x_8 + x_1x_{12} + x_6x_{17} + x_{10}x_{13} + x_{10}x_{17} + x_{13}x_{14} + x_5x_{10}x_{11}x_{18} + x_2x_5x_{11}x_{16}x_{18} + x_2x_5x_{11}x_{17}x_{18} + x_5x_{10}x_{11}x_{13}x_{18} + x_5x_{11}x_{13}x_{14}x_{18} + x_5x_{11}x_{16}x_{17}x_{18};$$

$$A_7(x_0,x_1,\ldots,x_{27}) = x_0 + x_1 + x_5 + x_{20} + x_{25} + x_1x_2 + x_2x_{17} + x_4x_{12} + x_{10}x_{15} + x_{10}x_{18} + x_{14}x_{16} + x_{16}x_{20} + x_7x_9x_{18}x_{19} + x_7x_9x_{10}x_{15}x_{19} + x_7x_9x_{10}x_{18}x_{19} + x_1x_2x_7x_9x_{13}x_{19};$$

$$A_8(x_0,x_1,\ldots,x_{28}) = x_0 + x_2 + x_{10} + x_{11} + x_{17} + x_{18} + x_{21} + x_{24} + x_1x_4 + x_8x_{21} + x_{10}x_{21} + x_{13}x_{19} + x_6x_{15}x_{19} + x_8x_9x_{18} + x_{13}x_{14}x_{16} + x_{13}x_{14}x_{19} + x_{13}x_{15}x_{19} + x_6x_{14}x_{15}x_{16} + x_6x_{14}x_{15}x_{19} + x_8x_9x_{18}x_{19} + x_{13}x_{14}x_{15}x_{16} + x_{13}x_{14}x_{15}x_{19} + x_8x_9x_{14}x_{16}x_{18} + x_8x_9x_{14}x_{18}x_{19};$$

$$A_9(x_0,x_1,\ldots,x_{29}) = x_0 + x_1 + x_7 + x_{10} + x_{12} + x_{18} + x_{28} + x_2x_8 + x_4x_7 + x_4x_{18} + x_{10}x_{12} + x_{10}x_{19} + x_{10}x_{22} + x_{14}x_{22} + x_3x_5x_7 + x_3x_7x_8 + x_5x_7x_8 + x_1x_3x_5x_9 + x_1x_3x_5x_{21} + x_1x_3x_8x_9 + x_1x_3x_8x_{21} + x_1x_5x_8x_9 + x_1x_5x_8x_{21} + x_3x_4x_5x_7 + x_3x_4x_5x_{18} + x_3x_4x_7x_8 + x_3x_4x_8x_{18} + x_3x_5x_9x_{21} + x_3x_8x_9x_{21} + x_4x_5x_7x_8 + x_4x_5x_8x_{18} + x_5x_8x_9x_{21};$$

$$A_{10}(x_0,x_1,\ldots,x_{30}) = x_0 + x_2 + x_5 + x_6 + x_{15} + x_{17} + x_{18} + x_{20} + x_{25} + x_8x_{18} + x_8x_{20} + x_{12}x_{21} + x_{14}x_{19} + x_{17}x_{21} + x_{20}x_{22} + x_4x_{12}x_{22} + x_4x_{19}x_{22} + x_7x_{20}x_{21} + x_8x_{18}x_{22} + x_8x_{20}x_{22} + x_{12}x_{19}x_{22} + x_{20}x_{21}x_{22} + x_4x_7x_{12}x_{21} + x_4x_7x_{19}x_{21} + x_4x_{12}x_{21}x_{22} + x_4x_{19}x_{21}x_{22} + x_7x_8x_{18}x_{21} + x_7x_8x_{20}x_{21} + x_7x_{12}x_{19}x_{21} + x_8x_{18}x_{21}x_{22} + x_8x_{20}x_{21}x_{22} + x_{12}x_{19}x_{21}x_{22};$$

$$A_{11}(x_0,x_1,\ldots,x_{31}) = x_0 + x_3 + x_{17} + x_{22} + x_{28} + x_2x_{13} + x_5x_{19} + x_7x_{19} + x_8x_{12} + x_8x_{13} + x_{13}x_{15} + x_2x_{12}x_{13} + x_7x_8x_{12} + x_7x_8x_{14} + x_8x_{12}x_{13} + x_2x_7x_{12}x_{13} + x_2x_7x_{13}x_{14} + x_4x_{11}x_{12}x_{24} + x_7x_8x_{12}x_{13} + x_7x_8x_{13}x_{14} + x_4x_7x_{11}x_{12}x_{24} + x_4x_7x_{11}x_{14}x_{24};$$

$$
\begin{aligned}
A_{12}(x_0, x_1, \ldots, x_{32}) = {} & x_0 + x_2 + x_7 + x_9 + x_{10} + x_{15} + x_{23} + x_{25} + x_{30} + x_8x_{15} \\
& + x_{12}x_{16} + x_{13}x_{15} + x_{13}x_{25} + x_1x_8x_{14} + x_1x_8x_{18} + x_8x_{12}x_{16} \\
& + x_8x_{14}x_{18} + x_8x_{15}x_{16} + x_8x_{15}x_{17} + x_{15}x_{17}x_{24} + x_1x_8x_{14}x_{17} \\
& + x_1x_8x_{17}x_{18} + x_1x_{14}x_{17}x_{24} + x_1x_{17}x_{18}x_{24} + x_8x_{12}x_{16}x_{17} \\
& + x_8x_{14}x_{17}x_{18} + x_8x_{15}x_{16}x_{17} + x_{12}x_{16}x_{17}x_{24} + x_{14}x_{17}x_{18}x_{24} \\
& + x_{15}x_{16}x_{17}x_{24}.
\end{aligned}
$$

Table 4 contains some properties of the shift registers A_j. Implementation related properties of the feedback shift registers can be found in Table 5.

A_j	N_j	L_j	$L_{\text{Prob},j}$	v_j	d_j	c_j	NL(A_j)	Diffusion λ_j
A_0	21	$2^{21}-5$	$> 2^{20}$	15	5	3	14336	46
A_1	22	$2^{22}-4$	$> 2^{21}$	15	5	3	15360	48
A_2	23	$2^{23}-2$	$> 2^{22}$	16	5	3	30720	51
A_3	24	$2^{24}-2$	$> 2^{23}$	17	4	4	61440	51
A_4	25	$> 2^{24.8}$	$> 2^{24}$	17	4	4	59392	48
A_5	26	$> 2^{24.8}$	$> 2^{25}$	17	4	5	59392	52
A_6	27	$> 2^{24.8}$	$> 2^{26}$	18	5	4	116736	51
A_7	28	$> 2^{24.8}$	$> 2^{27}$	18	6	4	123904	53
A_8	29	$> 2^{24.8}$	$> 2^{28}$	18	5	4	118784	58
A_9	30	$> 2^{24.8}$	$> 2^{29}$	17	4	3	62464	58
A_{10}	31	$> 2^{24.8}$	$> 2^{30}$	17	4	6	61440	61
A_{11}	32	$> 2^{24.8}$	$> 2^{31}$	17	5	4	57344	64
A_{12}	33	$> 2^{24.8}$	$> 2^{32}$	18	4	6	114688	54

Table 4: Properties of feedback shift registers

N_j is the length of the shift register A_j.

L_j is the linear complexity of the shift register A_j (i.e., the linear complexity of any nonzero output sequence of A_j). This number has been computed with the Berlekamp-Massey algorithm.

$L_{\text{Prob},j}$ is a lower bound for the linear complexity of the shift register A_j that has been determined by Algorithm B in Section 2.3 and which holds with probability $> 1 - 2^{-100}$.

v_j is the number of variables that occur explicitly in the algebraic normal form of the feedback function A_j.

d_j is the algebraic degree of A_j.

c_j is the order of correlation immunity of the feedback function A_j.

$\mathrm{NL}(A_j)$ is the nonlinearity of the feedback function A_j.

The diffusion parameter λ_j is the minimum number of clock cycles needed in order to transform any two initial states of the shift register A_j of Hamming distance 1 into shift register states of Hamming distance close to $N_j/2$. This parameter is determined experimentally. The diffusion parameter of a shift register should not be greater than 2.5 times the length of the shift register in order to provide short resynchronization times.

Shift register A_j	Number of taps	Design size in GE	Logical depth of 1-bit impl.	Logical depth of 8-bit impl.
A_0	15	31.00	3	5
A_1	15	29.50	3	5
A_2	16	28.75	3	5
A_3	17	30.25	3	5
A_4	17	30.25	3	5
A_5	17	31.75	3	5
A_6	18	29.25	3	5
A_7	18	28.50	3	5
A_8	18	31.00	3	5
A_9	17	30.00	3	5
A_{10}	17	31.75	3	5
A_{11}	17	31.00	3	5
A_{12}	18	31.75	3	5

Table 5: Hardware properties of shift registers

3.5 The key-loading algorithm

ACHTERBAHN-80 can be used with key lengths 40, 48, 56, 64, 72, and 80. All IV-lengths between 0 and 80 can be used provided that the IV-length is divisible by eight. ACHTERBAHN-128 accommodates all key lengths between 40 and 128 and all IV-lengths between 0 and 128 that are multiples of eight. We shall use the letters k and l to denote the key and IV length, respectively. Assume that the secret key K is given as the bit string $K = u_0 u_1 \dots u_{k-1}$, and that the initial value (or initial vector) is given as $IV = v_0 v_1 \dots v_{l-1}$. The key and IV-loading algorithm is defined as follows:

Step 1. The memory cells $D_0, D_1, \dots, D_{N_j-1}$ of the shift register A_j are filled with the first N_j key bits $u_0, u_1, \dots, u_{N_j-1}$. This is done for all thirteen shift registers in the keystream generator of ACHTERBAHN-128 and for all eleven shift registers in the keystream generator of ACHTERBAHN-80.

Step 2. Into each shift register A_j the remaining $k-N_j$ key bits $u_{N_j}, u_{N_j+1}, \dots, u_{k-1}$ are introduced, one after the other, according to Figure 4.

Step 3. Into each shift register A_j all l initial value bits $v_0, v_1, \dots, v_{l-1}$ are introduced in the same way as already described for the key bits in Step 2.

Step 4. Each shift register A_j emits one bit. The thirteen emitted bits are then compressed by the Boolean combining function F into one output bit. This output bit is immediately fed back into each shift register as depicted in Figure 4. The same output bit is fed into all thirteen shift registers. This operation is repeated 32 times. In the case of ACHTERBAHN-80, of course, only the eleven shift registers $A_1, \dots, A_{11}$ are involved and the used combining function is the function G defined in Section 3.3. Recall that the output bit of the shift register A_j at a particular time is defined to be the content of the shift register's memory cell D_{N_j-16} at that time.

Step 5. The content of the memory cell D_0 in each shift register A_j is overwritten with a 1. This operation makes sure that none of the shift registers gets initialized with the all zero state.

Step 6. Each shift A_j is clocked 64 times without emitting any output bit (warm-up).

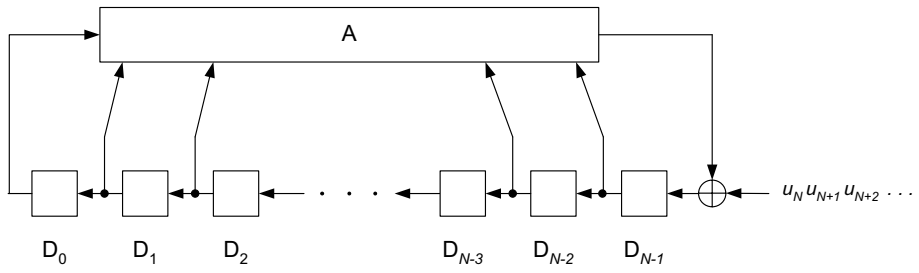


Figure 4: Bitwise introduction of key or IV-bits into a shift register

The states of the shift registers A_j at the end of Step 6 define the *initial state* of the keystream generator.

Step 1 can be implemented in different ways. We discuss two possible implementations. The first implementation aims to minimize the hardware costs of the keystream generator but leads to longer resynchronization times and provides less resistance to simple power analysis (SPA) attacks. In this implementation the first N_j key bits $u_0, u_1, \ldots, u_{N_j-1}$ are shifted into shift register A_j, bit by bit, according to Figure 5. It takes N_j clock cycles to get the key bits $u_0, u_1, \ldots, u_{N_j-1}$ into the register A_j by this method.

In the second implementation of Step 1, the first 16 key bits $u_0, u_1, \ldots, u_{15}$ are loaded simultaneously into each shift register A_j. The key bit u_i is loaded into cell D_{N_j-16+i} for $0 \leq i \leq 15$. Then the key bits $u_{16}, u_{17}, \ldots, u_{N_j-1}$ are shifted into register A_j according to Figure 5. This method requires $1 + N_j - 16$ clock cycles to fill the cells of the register A_j with the first N_j key bits.

The hardware costs of the second implementation of Step 1 are higher, since the 16 right-most flip-flops of each shift register must be scan flip-flops or one has to implement extra multiplexors. See the discussion at the end of Section 2.4.

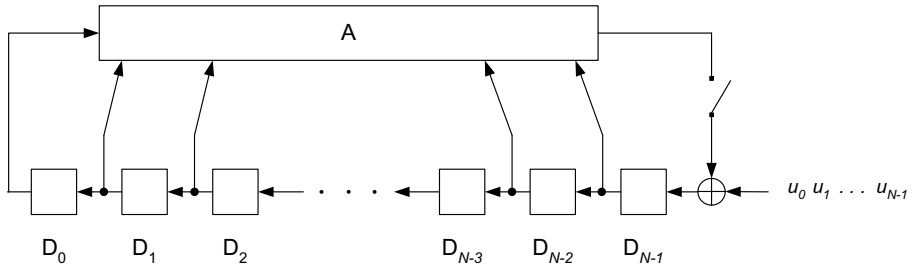


Figure 5: Bitwise introduction of the first N key bits into the shift register

The tasks of Steps 2, 3, 4, and 6 require $k - N_j$, l, 32, and 64 clock cycles, respectively. The task of Step 5 can be performed at the beginning of Step 6 in a way that no extra clock cycle is required. Thus Step 5 is counted with 0 clock cycles. To summarize, we obtain the following key/IV-loading times.

Fact 4. *The number of clock cycles needed to load a secret key of length k and an initial value of length l into the keystream generator of* ACHTERBAHN-128 *or* ACHTERBAHN-80 *is given by*

$$t_{\text{resync.1}} = 1 + \frac{k + l + 80}{q}$$

if in Step 1 the first 16 *key bits are loaded simultaneously into each shift register A_j. Otherwise we have*

$$t_{\text{resync.2}} = \frac{k + l + 96}{q}.$$

Here $q \in \{1, 2, 4, 8\}$ is the degree of parallelization.

Remark. Notice that Steps 1 and 2 depend only on the secret key K. This offers the possibility to reduce resynchronization times in some applications. The background

is that the secret key K remains constant for a longer period in time while the initial value IV changes frequently. The idea is to carry out Steps 1 and 2 only once for each key K and save the thus derived state of the keystream generator at the end of Step 2 in some external memory. If resynchronization is requested the state of the keystream generator at the end of Step 2 can be restored in an instant, so that the resynchronization time reduces to the time needed to perform the tasks in Steps $3-6$. We speak of *pure resynchronization times* if we refer to the time necessary to carry out Steps $3-6$. The pure resynchronization times for ACHTERBAHN-128 and ACHTERBAHN-80 are given by

$$t_{\text{resync.3}} = 1 + \frac{l+96}{q}.$$

4 Structural Properties

4.1 Linear complexity of keystream

It is well known [35], [16] that the linear complexity of the sum of two periodic sequences (with elements in a finite field) can never exceed the sum of the linear complexities of the sequences. Similarly, the linear complexity of the product of two periodic sequences cannot be greater than the product of the linear complexities of the sequences. In particular, if σ and τ are periodic sequences in $\mathbb{F}_2^\infty$, then

$$L(\sigma+\tau) \le L(\sigma) + L(\tau) \qquad \text{and} \qquad L(\sigma\tau) \le L(\sigma)L(\tau). \tag{6}$$

The keystream ζ of ACHTERBAHN-128 can be seen as the sum of several different periodic sequences most of which are the product of two, three or four sequences:

$$\zeta = F(\sigma_0, \sigma_1, \ldots, \sigma_{12}) = \sigma_0 + \sigma_1 + \sigma_2 + \cdots + \sigma_5\sigma_6\sigma_{11}\sigma_{12}. \tag{7}$$

For $j = 0, 1, \ldots, 12$, sequence σ_j is a nonzero output sequence of the FSR A_j of length N_j. The linear complexity of σ_j satisfies $L(\sigma_j) = L_j \le 2^{N_j} - 2$ for all j. Using now the inequalities (6), we obtain an upper bound for the linear complexity of the keystream:

$$L(\zeta) \le F(L_0, L_1, \ldots, L_{12}) \le F(2^{N_0} - 2,\, 2^{N_1} - 2,\, \ldots,\, 2^{N_{12}} - 2) < 2^{121}.$$

Here the function F is, of course, evaluated over the integers rather than over $\mathbb{F}_2$. In the same way, for the keystream ζ generated by ACHTERBAHN-80, we find

$$L(\zeta) \le G(L_1, L_2, \ldots, L_{11}) \le G(2^{N_1} - 2,\, 2^{N_2} - 2,\, \ldots,\, 2^{N_{11}} - 2) < 2^{119}.$$

Before we can establish lower bounds for the linear complexities of the keystreams of ACHTERBAHN-128 and ACHTERBAHN-80, we need some auxiliary results.

We first recall some results from Selmer [30, Chap. 4], and Zierler and Mills [35]. Let $f, g, \ldots, h$ be nonconstant binary polynomials without multiple roots in their respective splitting fields over $\mathbb{F}_2$ and with nonzero constant terms. Then $f \vee g \vee \cdots \vee h$ is defined to be the polynomial whose roots are the distinct[3] 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 $\mathbb{F}_2$. This follows from the fact that all conjugates (over $\mathbb{F}_2$) of a root of $f \vee g \vee \cdots \vee h$ are roots of $f \vee g \vee \cdots \vee h$.

Lemma 2. *Let $f, g, \ldots, h$ be irreducible binary polynomials with $f(0)g(0)\cdots h(0) \neq 0$ (i.e., none of the polynomials is equal to the irreducible polynomial $p(x) = x$). Let the canonical factorization of $f \vee g \vee \cdots \vee h$ over $\mathbb{F}_2$ be given by*

$$f \vee g \vee \cdots \vee h = b_1 \cdots b_d.$$

Then, for $j = 1, \ldots, d$, the degree of each polynomial b_j divides

$$l = \operatorname{lcm}(\deg(f), \deg(g), \ldots, \deg(h)).$$

[3] This is the definition of Zierler and Mills. Selmer defined the operation slightly different in so far that he defined $f \vee g$ (he used the notation $f \S\, g$) as the polynomial whose roots are all products $\alpha\beta$, where α is a root of f and β a root of g. The roots of $f \S\, g$ are then not necessarily distinct.

Proof. All roots of $f \vee g \vee \cdots \vee h$ lie in the splitting field of the polynomial $fg \cdots h$, which is the finite field $\mathbb{F}_{2^l}$. Consider an arbitrary root γ of $f \vee g \vee \cdots \vee h$. Let $b \in \mathbb{F}_2[x]$ be the minimal polynomial of γ (i.e., the uniquely determined irreducible polynomial in $\mathbb{F}_2[x]$ which has γ as a root). Then $\deg(b)$ divides l and the assertion follows. □

Lemma 3. *Let $f, g, \ldots, h$ be binary polynomials without multiple roots and with nonzero constant terms. The polynomial $f \vee g \vee \cdots \vee h \in \mathbb{F}_2[x]$ is irreducible if and only if the polynomials $f, g, \ldots, h$ are all irreducible and of pairwise relatively prime degrees. In this case*

$$\deg(f \vee g \vee \cdots \vee h) = \deg(f) \deg(g) \cdots \deg(h).$$

Proof. See Selmer [30, Chap. 4]. □

Lemma 4. *Let $f, g, \ldots, h$ be irreducible binary polynomials with $f(0)g(0)\cdots h(0) \neq 0$. Let $\sigma = (s_n)_{n=0}^{\infty}$, $\tau = (t_n)_{n=0}^{\infty}$, $\ldots$, $\upsilon = (u_n)_{n=0}^{\infty}$ be binary periodic sequences with minimal polynomials $f, g, \ldots, h$, respectively. Then $f \vee g \vee \cdots \vee h$ is a characteristic polynomial of the product sequence $\sigma\tau\cdots\upsilon = (s_n t_n \cdots u_n)_{n=0}^{\infty}$.*

Proof. See Lidl and Niederreiter [21, Chap. 8] or Zierler and Mills [35]. □

Lemma 5. *Let $f, g, \ldots, h$ be irreducible binary polynomials of pairwise relatively prime degrees and with $f(0)g(0)\cdots h(0) \neq 0$. Let $\sigma = (s_n)_{n=0}^{\infty}$, $\tau = (t_n)_{n=0}^{\infty}$, $\ldots$, $\upsilon = (u_n)_{n=0}^{\infty}$ be binary periodic sequences with minimal polynomials $f, g, \ldots, h$, respectively. Then $f \vee g \vee \cdots \vee h$ is the minimal polynomial of $\sigma\tau\cdots\upsilon = (s_n t_n \cdots u_n)_{n=0}^{\infty}$.*

Proof. See Selmer [30, Chap. 4]. □

Lemma 6. *Let $S, T, \ldots, U$ be positive integers. Let $\sigma = (s_n)_{n=0}^{\infty}$, $\tau = (t_n)_{n=0}^{\infty}$, $\ldots$, $\upsilon = (u_n)_{n=0}^{\infty}$ be binary periodic sequences with $\mathrm{per}(\sigma) = 2^S - 1$, $\mathrm{per}(\tau) = 2^T - 1$, $\ldots$, $\mathrm{per}(\upsilon) = 2^U - 1$. Let the canonical factorization over $\mathbb{F}_2$ of the minimal polynomial of $\sigma\tau\cdots\upsilon = (s_n t_n \cdots u_n)_{n=0}^{\infty}$ be*

$$m_{\sigma\tau\cdots\upsilon} = c_1 c_2 \cdots c_k.$$

Then $\deg(c_j)$ divides $\mathrm{lcm}(S, T, \ldots, U)$ for $j = 1, \ldots, k$.

Proof. Consider the minimal polynomials of $\sigma, \tau, \ldots, \upsilon$, and their canonical factorizations over $\mathbb{F}_2$:

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

By Proposition 4A (see Appendix A), we can write σ as the sum of sequences σ_i such that σ_i has minimal polynomial f_i, $1 \leq i \leq s$. The sequences $\tau, \ldots, \upsilon$ can be decomposed similarly. Thus we can write

$$\sigma = \sum_{i=1}^{s} \sigma_i, \qquad \tau = \sum_{j=1}^{t} \tau_j, \qquad \ldots, \qquad \upsilon = \sum_{k=1}^{u} \upsilon_k,$$

which implies that

$$\sigma\tau\cdots\upsilon = \sum_{i=1}^{s}\sum_{j=1}^{t}\cdots\sum_{k=1}^{u}\sigma_i\tau_j\cdots\upsilon_k.$$

By Lemma 4, each sequence $\sigma_i\tau_j\cdots\upsilon_k$ has $f_i\vee g_j\vee\cdots\vee h_k$ as a characteristic polynomial. It is well known, and easy to show, that the sum of a finite number of periodic sequences has as a characteristic polynomial the least common multiple of the characteristic polynomials of the individual sequences (see e.g., [21, Theorem 8.55]). It therefore follows that

$$c = \operatorname{lcm}(\{f_i \vee g_j \vee \cdots \vee h_k : 1 \le i \le s,\ 1 \le j \le t, \ldots, 1 \le k \le u\}) \tag{8}$$

is a characteristic polynomial of the product sequence $\sigma\tau\cdots\upsilon$.

Consider an arbitrary but fixed polynomial $f_i \vee g_j \vee \cdots \vee h_k$. By Lemma 2, $f_i \vee g_j \vee \cdots \vee h_k$ is the product of distinct irreducible binary polynomials whose degrees divide $\operatorname{lcm}(\deg(f_i), \deg(g_j), \ldots, \deg(h_k))$. By Lemma 1, $\deg(f_i)$ divides S, $\deg(g_j)$ divides T, ..., $\deg(h_k)$ divides U. It follows that all irreducible factors of the polynomial $f_i \vee g_j \vee \cdots \vee h_k$ have degrees dividing $\operatorname{lcm}(S, T, \ldots, U)$. Since each polynomial $f_i \vee g_j \vee \cdots \vee h_k$, $1 \le i \le s$, $1 \le j \le t$, ..., $1 \le k \le u$, has this property, we conclude that the characteristic polynomial $c \in \mathbb{F}_2[x]$ of $\sigma\tau\cdots\upsilon$ in (8) is the product of distinct irreducible binary polynomials whose degrees divide $\operatorname{lcm}(S, T, \ldots, U)$. Since the minimal polynomial of a periodic sequence divides any characteristic polynomial of the sequence, the proof is complete. □

Lemma 7. *Let $S, T, \ldots, U$ be pairwise relatively prime integers greater than 1. Let $\sigma = (s_n)_{n=0}^{\infty}$, $\tau = (t_n)_{n=0}^{\infty}$, ..., $\upsilon = (u_n)_{n=0}^{\infty}$ be binary periodic sequences of least periods $2^S - 1$, $2^T - 1$, ..., $2^U - 1$, respectively. If the canonical factorizations over $\mathbb{F}_2$ of the minimal polynomials of $\sigma, \tau, \ldots, \upsilon$ are*

$$m_\sigma = \prod_{i=1}^{s} f_i, \qquad m_\tau = \prod_{j=1}^{t} g_j, \qquad \ldots, \qquad m_\upsilon = \prod_{k=1}^{u} h_k, \tag{9}$$

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

$$m_{\sigma\tau\cdots\upsilon} = \prod_{i=1}^{s}\prod_{j=1}^{t}\cdots\prod_{k=1}^{u}(f_i \vee g_j \vee \cdots \vee h_k). \tag{10}$$

In fact, this is the canonical factorization of the minimal polynomial of $\sigma\tau\cdots\upsilon$ in $\mathbb{F}_2[x]$.

Proof. See Appendix A or [8, Lemma 7]. □

The following corollary to Lemma 7 is also a special case of Golić [14, Theorem 3] and of Rueppel and Staffelbach [29, Theorem 4].

Corollary 1. *Let $\sigma, \tau, \ldots, \upsilon$ be binary periodic sequences of least periods $2^S - 1$, $2^T - 1$, ..., $2^U - 1$, and linear complexities $L(\sigma), L(\tau), \ldots, L(\upsilon)$, respectively. If the integers $S, T, \ldots, U$ are pairwise relatively prime and greater than 1, then the product sequence $\sigma\tau\cdots\upsilon$ has linear complexity*

$$L(\sigma\tau\cdots\upsilon) = L(\sigma)L(\tau)\cdots L(\upsilon).$$

Proof. Let the minimal polynomials of the sequences $\sigma, \tau, \ldots, \upsilon$ be given by the expressions in (9). Using Lemma 3, Lemma 7, and $L(\sigma\tau\cdots\upsilon) = \deg(m_{\sigma\tau\cdots\upsilon})$, we get

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

□

Theorem 1. *The linear complexity of the keystream ζ of* ACHTERBAHN-128 *is lower bounded by*

$$L_2L_8L_{10}(L_4 + L_9 - 59) \leq L(\zeta).$$

Using the fact that $L_2 = 2^{23} - 2$ and $L_j \geq 2^{24.8}$ for $4 \leq j \leq 12$ (results derived by means of the Berlekamp-Massey algorithm), we get

$$L(\zeta) > 2^{98}.$$

Using $L_j \geq 2^{N_j - 1}$ for $j \geq 4$ (probabilistic estimation derived by means of Algorithm B in Section 2.3), we obtain

$$L(\zeta) > 2^{110}.$$

The same lower bounds hold for the keystream of ACHTERBAHN-80.

Proof. Consider the algebraic normal form of the Boolean combining function $F(x_0, x_1, \ldots, x_{12})$ of ACHTERBAHN-128 as specified in Section 3.2. The algebraic normal form of the function contains 124 monomials, among them the three monomials:

$$x_2x_8x_{10}, \qquad x_2x_4x_8x_{10}, \qquad x_2x_8x_9x_{10}. \tag{11}$$

Notice that each monomial contains the variables x_2, x_8, and x_{10}. The shift register lengths corresponding to these variables are $N_2 = 23$, $N_8 = 29$, and $N_{10} = 31$. The keystream $\zeta = F(\sigma_0, \sigma_1, \ldots, \sigma_{12})$ is the sum of 124 different sequences. The three sequences that correspond to the monomials in (11) are

$$\sigma_2\sigma_8\sigma_{10}, \qquad \sigma_2\sigma_4\sigma_8\sigma_{10}, \qquad \sigma_2\sigma_8\sigma_9\sigma_{10}. \tag{12}$$

Let the minimal polynomials m_2, m_8, m_{10} of the sequences $\sigma_2, \sigma_8, \sigma_{10}$ be given by their respective canonical factorizations

$$m_2 = \prod_{i_2=1}^{d_2} f_{i_2}^{(2)}, \qquad m_8 = \prod_{i_8=1}^{d_8} f_{i_8}^{(8)}, \qquad m_{10} = \prod_{i_{10}=1}^{d_{10}} f_{i_{10}}^{(10)}.$$

By Lemma 1 all (irreducible) polynomials $f_{i_2}^{(2)}$ have degree 23, all polynomials $f_{i_8}^{(8)}$ have degree 29, and all polynomials $f_{i_{10}}^{(10)}$ have degree 31. It follows now from Lemma 7 that the minimal polynomial of the sequence $\sigma_2\sigma_8\sigma_{10}$ consists of $d_2d_8d_{10}$ distinct irreducible binary polynomials all of which have degree $23 \cdot 29 \cdot 31 = 20677$. Lemma 7, in conjunction with Lemma 3, implies that all irreducible factors of the minimal polynomials of $\sigma_2\sigma_4\sigma_8\sigma_{10}$ and $\sigma_2\sigma_8\sigma_9\sigma_{10}$ have degrees divisible by 20677.

The three monomials in (11) are the only monomials in the algebraic normal form of F containing all three variables x_2, x_8, and x_{10} simultaneously. It follows then from Lemma 6 and Lemma 7 that among the 124 sequences appearing in the sum (7), the three sequences in (12) are the only ones whose minimal polynomials contain irreducible factors whose degrees are multiples of 20677. In other words, the minimal polynomials of the three sequences in (12) are relatively prime to the minimal polynomials of the other 121 sequences in (7). It follows that the minimal polynomial m_ω of

$$\omega = \sigma_2\sigma_8\sigma_{10} + \sigma_2\sigma_4\sigma_8\sigma_{10} + \sigma_2\sigma_8\sigma_9\sigma_{10} \tag{13}$$

divides the minimal polynomial m_ζ of the keystream ζ. Hence, $\deg(m_\omega)$ yields a lower bound for the linear complexity of ζ. Consider the minimal polynomials m_4 and m_9 of the sequences σ_4 and σ_9:

$$m_4 = \prod_{i_4=1}^{d_4} f_{i_4}^{(4)} \qquad \text{and} \qquad m_9 = \prod_{i_9=1}^{d_9} f_{i_9}^{(9)}.$$

The sequence σ_4 is a nonzero output sequence of the primitive shift register A_4 of length $N_4 = 25$. Therefore, by Lemma 1, the irreducible polynomials $f_{i_4}^{(4)}$ have either degree 5 or degree 25. By the same lemma, as σ_9 is the nonzero output sequence of a primitive FSR of length $N_9 = 30$, we have $\deg(f_{i_9}^{(9)}) \in \{2, 3, 5, 6, 10, 15, 30\}$ for $1 \le i_9 \le d_9$.

According to Lemma 7, the canonical factorization of the minimal polynomial of $\sigma_2\sigma_4\sigma_8\sigma_{10}$ is given by

$$m_{\sigma_2\sigma_4\sigma_8\sigma_{10}} = \prod_{i_2=1}^{d_2} \prod_{i_4=1}^{d_4} \prod_{i_8=1}^{d_8} \prod_{i_{10}=1}^{d_{10}} \left(f_{i_2}^{(2)} \vee f_{i_4}^{(4)} \vee f_{i_8}^{(8)} \vee f_{i_{10}}^{(10)}\right).$$

The $d_2d_4d_8d_{10}$ distinct irreducible factors can have two different degrees:

$$\deg\left(f_{i_2}^{(2)} \vee f_{i_4}^{(4)} \vee f_{i_8}^{(8)} \vee f_{i_{10}}^{(10)}\right) = \begin{cases} 23 \cdot 5 \cdot 29 \cdot 31 = 103385, \\ 23 \cdot 25 \cdot 29 \cdot 31 = 516925. \end{cases}$$

The minimal polynomial of $\sigma_2\sigma_8\sigma_9\sigma_{10}$ is

$$m_{\sigma_2\sigma_8\sigma_9\sigma_{10}} = \prod_{i_2=1}^{d_2} \prod_{i_8=1}^{d_8} \prod_{i_9=1}^{d_9} \prod_{i_{10}=1}^{d_{10}} \left(f_{i_2}^{(2)} \vee f_{i_8}^{(8)} \vee f_{i_9}^{(9)} \vee f_{i_{10}}^{(10)}\right).$$

There are seven possible values for the degrees of the irreducible factors:

$$\deg\left(f_{i_2}^{(2)} \vee f_{i_8}^{(8)} \vee f_{i_9}^{(9)} \vee f_{i_{10}}^{(10)}\right) = \begin{cases} 23 \cdot 29 \cdot \ 2 \cdot 31 = \ 41354, \\ 23 \cdot 29 \cdot \ 3 \cdot 31 = \ 62031, \\ 23 \cdot 29 \cdot \ 5 \cdot 31 = 103385, \\ 23 \cdot 29 \cdot \ 6 \cdot 31 = 124062, \\ 23 \cdot 29 \cdot 10 \cdot 31 = 206770, \\ 23 \cdot 29 \cdot 15 \cdot 31 = 310155, \\ 23 \cdot 29 \cdot 30 \cdot 31 = 620310. \end{cases}$$

Let us separate the $d_2 d_8 d_9 d_{10}$ irreducible polynomials $f_{i_2}^{(2)} \vee f_{i_8}^{(8)} \vee f_{i_9}^{(9)} \vee f_{i_{10}}^{(10)}$, $1 \leq i_j \leq d_j$, $j = 2, 8, 9, 10$, into seven equivalence classes such that the polynomials in each class have the same degree. Since there are six irreducible binary polynomials of degree 5, the equivalence class containing all polynomials of degree 103385 can contain at most $6\, d_2 d_8 d_{10}$ different polynomials. Some, or all, of these polynomials might also appear in the canonical factorization of the minimal polynomial of $\sigma_2\sigma_4\sigma_8\sigma_{10}$. Therefore,

$$\begin{aligned} \deg(\gcd(m_{\sigma_2\sigma_4\sigma_8\sigma_{10}}, m_{\sigma_2\sigma_8\sigma_9\sigma_{10}})) &\leq 6\, d_2 d_8 d_{10} \cdot 103385 \\ &= 30 \cdot (23\, d_2)(29\, d_8)(31\, d_{10}) = 30\, L_2 L_8 L_{10}. \end{aligned} \tag{14}$$

Consider the sequence ω in (13). Since the minimal polynomial of $\sigma_2\sigma_8\sigma_{10}$ has only irreducible factors of degree 20677, it is relatively prime to both $m_{\sigma_2\sigma_4\sigma_8\sigma_{10}}$, and to $m_{\sigma_2\sigma_8\sigma_9\sigma_{10}}$. In order to get information about the minimal polynomial of the sum $\sigma_2\sigma_4\sigma_8\sigma_{10} + \sigma_2\sigma_8\sigma_9\sigma_{10}$, recall that every binary periodic sequence can be identified with a rational function of $\mathbb{F}_2(x)$ (see Appendix A). If the rational function that corresponds to the periodic sequence is in reduced form (numerator and denominator are relatively prime), then the polynomial in the denominator is the minimal polynomial of the sequence. The sum of two reduced rational functions with denominator polynomials a and b is a reduced rational function whose denominator polynomial is a multiple of $\mathrm{lcm}(a, b)/\gcd(a, b)$. Therefore, the minimal polynomial of the sum $\sigma_2\sigma_4\sigma_8\sigma_{10} + \sigma_2\sigma_8\sigma_9\sigma_{10}$ is a multiple of the polynomial

$$\frac{\mathrm{lcm}(m_{\sigma_2\sigma_4\sigma_8\sigma_{10}}, m_{\sigma_2\sigma_8\sigma_9\sigma_{10}})}{\gcd(m_{\sigma_2\sigma_4\sigma_8\sigma_{10}}, m_{\sigma_2\sigma_8\sigma_9\sigma_{10}})} = \frac{m_{\sigma_2\sigma_4\sigma_8\sigma_{10}} \cdot m_{\sigma_2\sigma_8\sigma_9\sigma_{10}}}{(\gcd(m_{\sigma_2\sigma_4\sigma_8\sigma_{10}}, m_{\sigma_2\sigma_8\sigma_9\sigma_{10}}))^2}.$$

Combining these facts, we get

$$\begin{aligned} L(\zeta) \geq \deg(m_\omega) \geq \deg(m_{\sigma_2\sigma_8\sigma_{10}}) + \deg(m_{\sigma_2\sigma_4\sigma_8\sigma_{10}}) + \deg(m_{\sigma_2\sigma_8\sigma_9\sigma_{10}}) \\ - 2\deg(\gcd(m_{\sigma_2\sigma_4\sigma_8\sigma_{10}}, m_{\sigma_2\sigma_8\sigma_9\sigma_{10}})). \end{aligned}$$

Using Corollary 1 and (14), we obtain

$$\begin{aligned} L(\zeta) &\geq L_2 L_8 L_{10} + L_2 L_4 L_8 L_{10} + L_2 L_8 L_9 L_{10} - 60\, L_2 L_8 L_{10} \\ &= L_2 L_8 L_{10}(L_4 + L_9 - 59). \end{aligned}$$

The derived lower bound for the linear complexity holds also for the keystream of ACHTERBAHN-80. This is immediate from the fact that the algebraic normal form of the combining function G of ACHTERBAHN-80 contains the three monomials in (11), and only monomials that appear in the algebraic normal form of F, the combining function of ACHTERBAHN-128. □

4.2 Period of keystream

Let f be a nonzero binary polynomial with $f(0) \neq 0$. The least positive integer e for which $f(x)$ divides $x^e - 1$ is called the *order* of f and denoted by $\mathrm{ord}(f)$. If $f \in \mathbb{F}_2[x]$ has degree d, then $\mathrm{ord}(f) \leq 2^d - 1$. If f is irreducible, then $\mathrm{ord}(f)$ is equal to the order of any root of f in the multiplicative group $\mathbb{F}_{2^d}^*$ of all nonzero elements of the finite field $\mathbb{F}_{2^d}$, so that $\mathrm{ord}(f)$ divides $2^d - 1$ in this case. The polynomial f is primitive if and only if $\mathrm{ord}(f) = 2^d - 1$. See [21, Chap. 3, Sec. 1] for more information.

The concept of the order of a polynomial comes into play for the following reason: If σ is a binary periodic sequence with minimal polynomial m_σ, then the least period of σ is equal to the order of the minimal polynomial of σ (see [21, Theorem 8.44]). This fact will be used in the proof of the next theorem. We shall also need the following two lemmas.

Lemma 8. *Let $g_1, \ldots, g_k$ be pairwise relatively prime nonzero polynomials over $\mathbb{F}_2$ with $g_1(0) \cdots g_k(0) \neq 0$, and let $h = g_1 \cdots g_k$. Then*

$$\mathrm{ord}(h) = \mathrm{lcm}(\mathrm{ord}(g_1), \ldots, \mathrm{ord}(g_k)).$$

Proof. See [21, Theorem 3.9]. □

Lemma 9. *Let $f, g, \ldots, h$ be irreducible binary polynomials of pairwise relatively prime degrees and with $f(0)g(0)\cdots h(0) \neq 0$. Then*

$$\mathrm{ord}(f \vee g \vee \cdots \vee h) = \mathrm{ord}(f)\,\mathrm{ord}(g) \cdots \mathrm{ord}(h).$$

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, $\mathrm{ord}(f)$ 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 $\mathrm{ord}(g)$ and both numbers divide $2^b - 1$. By hypothesis, the greatest common divisor of a and b is 1, so that $\gcd(2^a - 1, 2^b - 1) = 1$. It follows that α and β are elements of relatively prime orders in the group $\mathbb{F}_{2^{ab}}^*$. By Lemma 3, 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 $\mathrm{ord}(f \vee g) = \mathrm{ord}(\alpha\beta) = \mathrm{ord}(\alpha)\,\mathrm{ord}(\beta) = \mathrm{ord}(f)\,\mathrm{ord}(g)$. □

Theorem 2. *The least period of the keystream ζ of* ACHTERBAHN-128 *is*

$$\mathrm{per}(\zeta) = \mathrm{lcm}(2^{N_0} - 1, 2^{N_1} - 1, \ldots, 2^{N_{12}} - 1) > 2^{298},$$

where $N_j = 21 + j$ are the lengths of the shift registers A_j, $j = 0, 1, \ldots, 12$,. The least period of the keystream of ACHTERBAHN-80 *is given by*

$$\mathrm{per}(\zeta) = \mathrm{lcm}(2^{N_1} - 1, 2^{N_2} - 1, \ldots, 2^{N_{11}} - 1) > 2^{268}.$$

Proof. Clearly, the least period of the keystream ζ produced by the keystream generator of ACHTERBAHN-128 cannot be greater than the least common multiple of the least periods of the individual nonzero shift register output sequences $\sigma_0, \sigma_1, \ldots, \sigma_{12}$. In other words,

$$\mathrm{per}(\zeta) \le \mathrm{lcm}(2^{N_0} - 1, 2^{N_1} - 1, \ldots, 2^{N_{12}} - 1).$$

We shall show that equality holds.

The algebraic normal form of the combining function $F(x_0, x_1, \ldots, x_{12})$ of ACHTERBAHN-128 contains the seven monomials

$$\begin{gathered} x_0x_5x_8x_{10}, \quad x_1x_4x_8x_{10}, \quad x_2x_3x_4x_8, \quad x_2x_4x_7x_8, \\ x_2x_4x_{11}x_{12}, \quad x_2x_8x_9x_{10}, \quad x_5x_6x_8x_{10}. \end{gathered}$$

Notice that each variable $x_0, x_1, \ldots, x_{12}$ occurs in at least one of those monomials. For each of the monomials, the lengths of the shift registers corresponding to the variables in the monomial are pairwise relatively prime. For example, for the first monomial we have $N_0 = 21$, $N_5 = 26$, $N_8 = 29$, and $N_{10} = 31$.

For $j = 0, 1, \ldots, 12$, let $f_j \in \mathbb{F}_2[x]$ be a primitive polynomial of degree N_j which divides the minimal polynomial m_j of the shift register A_j. (We can find such primitive polynomials f_j even if we do not know the minimal polynomial m_j using Algorithm A in Section 2.3.)

The selected seven monomials give rise to the following seven product sequences

$$\begin{gathered} \sigma_0\sigma_5\sigma_8\sigma_{10}, \quad \sigma_1\sigma_4\sigma_8\sigma_{10}, \quad \sigma_2\sigma_3\sigma_4\sigma_8, \quad \sigma_2\sigma_4\sigma_7\sigma_8, \\ \sigma_2\sigma_4\sigma_{11}\sigma_{12}, \quad \sigma_2\sigma_8\sigma_9\sigma_{10}, \quad \sigma_5\sigma_6\sigma_8\sigma_{10}, \end{gathered} \tag{15}$$

which appear among the sequences in

$$\zeta = F(\sigma_0, \sigma_1, \ldots, \sigma_{12}) = \sigma_0 + \sigma_1 + \sigma_2 + \cdots + \sigma_5\sigma_6\sigma_{11}\sigma_{12}. \tag{16}$$

For each of the sequences in (15) we know the structure of its minimal polynomial by Lemma 7. In particular, we know that the minimal polynomial of the first sequence $\sigma_0\sigma_5\sigma_8\sigma_{10}$ contains the irreducible polynomial $g_1 = f_0 \vee f_5 \vee f_8 \vee f_{10}$ as a factor, that the minimal polynomial of the second sequence $\sigma_1\sigma_4\sigma_8\sigma_{10}$ contains the irreducible polynomial $g_2 = f_1 \vee f_4 \vee f_8 \vee f_{10}$ as a factor, etc. An important implication of Lemma 6 and Lemma 7 is that each of the irreducible polynomials g_j, $1 \le j \le 7$, divides the minimal polynomial of one and only one of the 124 sequences in (16). Since the irreducible polynomials $g_1, \ldots, g_7$ are distinct (they have pairwise different degrees), we conclude that the polynomial

$$h = \prod_{j=1}^{7} g_j$$

is a divisor of the minimal polynomial of ζ. This, of course, implies that $\mathrm{ord}(h) \le \mathrm{ord}(m_\zeta)$. Since $g_1, \ldots, g_7$ are pairwise relatively prime polynomials,

$$\mathrm{ord}(h) = \mathrm{lcm}(\mathrm{ord}(g_1), \ldots, \mathrm{ord}(g_7)) \tag{17}$$

according to Lemma 8.

Apply Lemma 9 to the polynomial $g_1 = f_0 \vee f_5 \vee f_8 \vee f_{10}$ to obtain

$$\mathrm{ord}(g_1) = \mathrm{ord}(f_0)\,\mathrm{ord}(f_5)\,\mathrm{ord}(f_8)\,\mathrm{ord}(f_{10}) = (2^{21}-1)(2^{26}-1)(2^{29}-1)(2^{31}-1).$$

Similarly, we get

$$\begin{aligned}
\mathrm{ord}(g_2) &= \mathrm{ord}(f_1)\,\mathrm{ord}(f_4)\,\mathrm{ord}(f_8)\,\mathrm{ord}(f_{10}) = (2^{22}-1)(2^{25}-1)(2^{29}-1)(2^{31}-1),\\
\mathrm{ord}(g_3) &= \mathrm{ord}(f_2)\,\mathrm{ord}(f_3)\,\mathrm{ord}(f_4)\,\mathrm{ord}(f_8) = (2^{23}-1)(2^{24}-1)(2^{25}-1)(2^{29}-1),\\
\mathrm{ord}(g_4) &= \mathrm{ord}(f_2)\,\mathrm{ord}(f_4)\,\mathrm{ord}(f_7)\,\mathrm{ord}(f_8) = (2^{23}-1)(2^{25}-1)(2^{28}-1)(2^{29}-1),\\
\mathrm{ord}(g_5) &= \mathrm{ord}(f_2)\,\mathrm{ord}(f_4)\,\mathrm{ord}(f_{11})\,\mathrm{ord}(f_{12}) = (2^{23}-1)(2^{25}-1)(2^{32}-1)(2^{33}-1),\\
\mathrm{ord}(g_6) &= \mathrm{ord}(f_2)\,\mathrm{ord}(f_8)\,\mathrm{ord}(f_9)\,\mathrm{ord}(f_{10}) = (2^{23}-1)(2^{29}-1)(2^{30}-1)(2^{31}-1),\\
\mathrm{ord}(g_7) &= \mathrm{ord}(f_5)\,\mathrm{ord}(f_6)\,\mathrm{ord}(f_8)\,\mathrm{ord}(f_{10}) = (2^{26}-1)(2^{27}-1)(2^{29}-1)(2^{31}-1).
\end{aligned}$$

Substituting the right-hand sides into formula 17, we obtain

$$\mathrm{ord}(h) = \mathrm{lcm}(2^{21}-1, 2^{22}-1, \ldots, 2^{33}-1). \tag{18}$$

Since $\mathrm{ord}(h) \le \mathrm{ord}(m_\zeta)$, $\mathrm{ord}(m_\zeta) = \mathrm{per}(\zeta)$, and the right-hand side of (18) is an upper bound for $\mathrm{per}(\zeta)$, we conclude that

$$\mathrm{per}(\zeta) = \mathrm{lcm}(2^{21}-1, 2^{22}-1, \ldots, 2^{33}-1).$$

The assertion concerning the least period of the keystream of ACHTERBAHN-80 is proved in the same way. □

4.3 Robustness of keystream

The keystream generator of ACHTERBAHN is a specific realization of what we shall call a *primitive FSR combination generator*. The concept generalizes the well known and widely studied LFSR-based combination generator in which the output sequences of several linear feedback shift registers with primitive characteristic polynomials are combined by a suitable combining function.

A primitive FSR combination generator consists of a Boolean combining function $F : \mathbb{F}_2^n \to \mathbb{F}_2$ and n primitive binary FSR's $A_1, \ldots, A_n$. We shall use the notation $\mathrm{KSG}(F; A_1, \ldots, A_n)$. For each possible initialization of the shift registers $A_1, \ldots, A_n$ with nonzero initial states the produced binary periodic sequence ζ is called a *keystream* of $\mathrm{KSG}(F; A_1, \ldots, A_n)$. Two different keystreams may be translation distinct, or they may be shifted versions of each other.[4]

The combining function F of a FSR combination generator $\mathrm{KSG}(F; A_1, \ldots, A_n)$ should not depend on any of its variables linearly. For if it does, say, if F has the form

$$F(x_1, \ldots, x_n) = x_1 + G(x_2, \ldots, x_n),$$

[4]If the lengths of the primitive FSR's $A_1, \ldots, A_n$ are pairwise relatively prime, then all keystreams of $\mathrm{KSG}(F; A_1, \ldots, A_n)$ are shifted versions of one sequence.

then we can do the following. Let σ_j be a nonzero output sequence of the FSR A_j, $1 \leq j \leq n$. Let $p_1 = \mathrm{per}(\sigma_1)$ be the least period of σ_1. Consider the keystream ζ.

$$\zeta = F(\sigma_1, \ldots, \sigma_n) = \sigma_1 + G(\sigma_2, \ldots, \sigma_n).$$

The linear operator $g(T) = T^{p_1} - I$ annihilates σ_1. Applying $g(T)$ to the left-hand side and right-hand side of the above equation we get

$$g(T)\zeta = g(T)G(\sigma_2, \ldots, \sigma_n).$$

The sequence $\eta = g(T)\zeta$ depends only on the $n-1$ shift registers $A_2, \ldots, A_n$.

In order to prevent this undesirable phenomenon it is, however, not sufficient to demand that F does not depend linearly on any of its variables as can be seen by the following example. Let F have algebraic degree ≥ 3 and be of the form

$$F(x_1, \ldots, x_n) = x_1x_2 + x_2x_3 + G(x_4, \ldots, x_n).$$

Let $p_j = \mathrm{per}(\sigma_j)$ for $j = 1, 2, 3$, let $a = \mathrm{lcm}(p_1, p_2)$, and let $b = \mathrm{lcm}(p_2, p_3)$. Then a is a period of $\sigma_1\sigma_2$ and b is a period of $\sigma_2\sigma_3$. The binary polynomial

$$g(x) = (x^a - 1)(x^b - 1) = x^{a+b} + x^b + x^a + 1$$

is a characteristic polynomial of $\sigma_1\sigma_2 + \sigma_2\sigma_3$. Therefore, if we apply the linear operator $g(T)$ to both sides of

$$\zeta = \sigma_1\sigma_2 + \sigma_2\sigma_3 + G(\sigma_4, \ldots, \sigma_n),$$

we obtain the sequence

$$\eta = g(T)\zeta = g(T)G(\sigma_4, \ldots, \sigma_n).$$

The information inherent in the first three shift registers A_1, A_2, A_3 (the initial states of the shift registers and the form of the feedback functions) is no longer present in the "manipulated" keystream $g(T)\zeta$. This observation leads us to the following design rule.

Rule 1: If the combining function $F(x_1, \ldots, x_n)$ has algebraic degree d and is given by its algebraic normal form, then each variable x_j, $1 \leq j \leq n$, should appear in at least one monomial of degree d.[5]

While this rule excludes the existence of sparse polynomials g of relatively low degree that have the above demonstrated undesirable effect on the keystream, the rule is not strong enough to imply the nonexistence of polynomials up to a certain degree which have that undesirable effect on the keystream. We need a stronger rule.

Rule 2: For each variable x_j, $1 \leq j \leq n$, there should be a monomial of degree d in the algebraic normal form of F containing x_j such that the lengths of the FSR's that are assigned to the variables of the monomial are pairwise relatively prime.

[5]The Boolean combining function R of ACHTERBAHN-1 violated this rule. It contained four of its eight variables linearly.

Definition 3. A keystream ζ of $\mathrm{KSG}(F; A_1, \ldots, A_n)$ is called *r-robust*, or is said to have *degree of robustness* r, if for all binary polynomials g with $\deg(g) \leq r$, the sequence $g(T)\zeta$ depends on all n shift registers $A_1, \ldots, A_n$. The sequence $g(T)\zeta$ is called *independent* of a particular shift register A_j, $j \in \{1, \ldots, n\}$, if there exists a polynomial $h \in \mathbb{F}_2[x]$ such that $g(T)\zeta = h(T)\zeta'$, where ζ' is a keystream of some "smaller" keystream generator $\mathrm{KSG}(F'; A_{i_1}, \ldots, A_{i_k})$ with a combining function $F' : \mathbb{F}_2^k \to \mathbb{F}_2$ of algebraic degree not greater than the algebraic degree of F, and where $\{A_{i_1}, \ldots, A_{i_k}\}$ is a subset of $\{A_1, \ldots, A_n\}$ not containing A_j. The sequence $g(T)\zeta$ is said to *depend* on the FSR A_j if it is not independent of A_j.

Theorem 3. *All keystreams produced by the keystream generator of* ACHTERBAHN-80 *and* ACHTERBAHN-128 *have degree of robustness* $\geq 2^{95}$.

In preparation for the proof of the theorem we introduce and discuss the concept of the *dominator* of a periodic sequence.

Definition 4. Let σ be a binary periodic sequence with minimal polynomial $m_\sigma \in \mathbb{F}_2[x]$. The *dominator* of σ, denoted by $\mathrm{dom}(\sigma)$, is the product of all irreducible binary polynomials of maximum degree appearing in the canonical factorization of m_σ over $\mathbb{F}_2$. The dominator of a primitive binary FSR is the dominator of any nonzero output sequence of the shift register.

We wish to derive a lower bound for the degree of the dominator of each NLFSR $A_0, A_1, \ldots, A_{12}$ deployed in the keystream generator of ACHTERBAHN-128/80. To this end we need the next lemma.

Lemma 10. *The product $I(q, n; x)$ of all monic irreducible polynomials in $\mathbb{F}_q[x]$ of degree n is given by*

$$I(q, n; x) = \prod_{d|n} \left(x^{q^{n/d}} - x\right)^{\mu(d)},$$

where the product is extended over all positive divisors d of n, and μ denotes the Moebius function.

Proof. This is Theorem 3.29 in the finite field text book [21] of Lidl and Niederreiter. □

Let σ_j be any nonzero output sequence of the FSR A_j, $0 \leq j \leq 12$. From Section 3.4 we know that $L(\sigma_0) = 2^{21} - 5$, $L(\sigma_1) = 2^{22} - 4$, $L(\sigma_2) = 2^{23} - 2$, $L(\sigma_3) = 2^{24} - 2$. It follows that for $j = 0, 1, 2, 3$, the minimal polynomial of σ_j contains *all* irreducible binary polynomials of degree N_j. Thus, we have

$$\mathrm{dom}(\sigma_j) = I(2, N_j; x) = \prod_{d|N_j} \left(x^{2^{N_j/d}} - x\right)^{\mu(d)} \quad \text{for } j = 0, 1, 2, 3,$$

and therefore,

$$\deg(\mathrm{dom}(\sigma_j)) = \sum_{d|N_j} \mu(d) 2^{N_j/d} \quad \text{for } j = 0, 1, 2, 3.$$

See Table 6 for the numerical values.

j	$\deg(\mathrm{dom}(\sigma_j))$	$\deg(\mathrm{dom}(\sigma_j))/L(\sigma_j)$
0	2 097 018	99.99%
1	4 192 254	99.95%
2	8 388 606	100%
3	16 772 880	99.97%

Table 6: Degrees of dominators of the first four FSR's

For the shift registers A_j, $4 \le j \le 12$, we do not know the exact value of $L_j = L(\sigma_j)$, but we know that $L_j \ge 2^{24.8}$. Recall that the minimal polynomial m_j of σ_j has the form

$$m_j = \prod_{i=1}^{d_j} f_i^{(j)}$$

with distinct irreducible polynomials $f_i^{(j)} \in \mathbb{F}_2[x]$ whose degrees divide N_j and are greater than 1. Assume that the polynomials are numbered in a way that in $f_1^{(j)}, f_2^{(j)}, \ldots, f_{d_j}^{(j)}$ the polynomials of degree N_j come first. Write $m_j = g_j h_j$ with $g_j = \mathrm{dom}(\sigma_j)$. That is,

$$g_j = \prod_{i=1}^{c_j} f_i^{(j)} \qquad \text{and} \qquad h_j = \prod_{i=c_j+1}^{d_j} f_i^{(j)},$$

where $\deg(f_i^{(j)}) = N_j$ for $1 \le i \le c_j$, and $\deg(f_i^{(j)}) < N_j$ for $c_j + 1 \le i \le d_j$. Then the polynomial

$$\prod_{\substack{d|N_j \\ 1<d<N_j}} I(2, d; x)$$

is a multiple of h_j. It follows that

$$\deg(h_j) \le \sum_{d|N_j}^{*} \deg(I(2, d; x)) = \sum_{d|N_j}^{*} \sum_{e|d} \mu(e) 2^{d/e},$$

where the asterix indicates that the corresponding sum is extended over all divisors d of N_j with $1 < d < N_j$. Finally, we obtain

$$\deg(\mathrm{dom}(\sigma_j)) = \deg(m_j) - \deg(h_j) \ge L(\sigma_j) - \sum_{d|N_j}^{*} \sum_{e|d} \mu(e) 2^{d/e} > 2^{24.79}. \qquad (19)$$

The algebraic normal form of the combining function F of ACHTERBAHN-128 contains 124 monomials. Therefore, any keystream of $\mathrm{KSG}(F; A_0, \ldots, A_{12})$ is the sum of 124 different periodic sequences (compare equation (16)). Among these sequences the following eleven sequences are of particular interest.

$$\begin{array}{llll} \sigma_0\sigma_5\sigma_8\sigma_{10}, & \sigma_1\sigma_4\sigma_8\sigma_{10}, & \sigma_2\sigma_3\sigma_4\sigma_8, & \sigma_2\sigma_4\sigma_7\sigma_8, \\ \sigma_2\sigma_4\sigma_7\sigma_{12}, & \sigma_2\sigma_4\sigma_8\sigma_{10}, & \sigma_2\sigma_4\sigma_8\sigma_{11}, & \sigma_2\sigma_4\sigma_{10}\sigma_{12}, \\ \sigma_2\sigma_4\sigma_{11}\sigma_{12}, & \sigma_2\sigma_8\sigma_9\sigma_{10}, & \sigma_5\sigma_6\sigma_8\sigma_{10}. & \end{array} \qquad (20)$$

Each sequence is the product of four shift register sequences and the lengths of the corresponding FSR's are pairwise relatively prime. Consider, e.g., the first sequence $\sigma_0\sigma_5\sigma_8\sigma_{10}$. The corresponding shift register lengths are $N_0 = 21$, $N_5 = 26$, $N_8 = 29$, $N_{10} = 31$.

Let for each $j = 0, 1, \ldots, 12$, the canonical factorization of the dominator of σ_j be given by

$$\mathrm{dom}(\sigma_j) = \prod_{i=1}^{c_j} f_i^{(j)}.$$

An application of Lemma 7 and Lemma 3 shows that the dominator of the sequence $\sigma_0\sigma_5\sigma_8\sigma_{10}$ is given by

$$w_1 = \mathrm{dom}(\sigma_0\sigma_5\sigma_8\sigma_{10}) = \prod_{i_0=1}^{c_0}\prod_{i_5=1}^{c_5}\prod_{i_8=1}^{c_8}\prod_{i_{10}=1}^{c_{10}} \left(f_{i_0}^{(0)} \vee f_{i_5}^{(5)} \vee f_{i_8}^{(8)} \vee f_{i_{10}}^{(10)}\right).$$

Selmer's Lemma 3 implies that the $c_0c_5c_8c_{10}$ distinct irreducible factors of w_1 all have degree $N_0N_5N_8N_{10} = 490854$. A variation of the proof of Corollary 1 shows that

$$\deg(w_1) = \deg(\mathrm{dom}(\sigma_0))\deg(\mathrm{dom}(\sigma_5))\deg(\mathrm{dom}(\sigma_8))\deg(\mathrm{dom}(\sigma_{10})).$$

Using $\deg(\mathrm{dom}(\sigma_0)) = 2097018$ (compare Table 6), and $\deg(\mathrm{dom}(\sigma_j)) > 2^{24.79}$ for $4 \le j \le 12$ (see inequality (19)), we conclude that

$$\deg(w_1) > 2^{95.36}.$$

Since all irreducible factors of w_1 have degree 490854, and—according to Lemma 6 and Lemma 7—$\sigma_0\sigma_5\sigma_8\sigma_{10}$ is the only sequence among the sequences in (16) whose minimal polynomial has an irreducible factor of degree 490854, it follows that the dominator w_1 divides the minimal polynomial of the keystream ζ.

What we have just shown for w_1 can be proved in the same way for the dominators $w_2, \ldots, w_{11}$ of the remaining ten sequences in (20). We collect the results in Table 7.

Proof of Therorem 3. Let ζ be a keystream of $\mathrm{KSG}(F; A_0, \ldots, A_{12})$, the keystream generator of ACHTERBAHN-128. Assume to the contrary that ζ has degree of robustness $< 2^{95}$. Then there exists a binary polynomial g with $\deg(g) < 2^{95}$ such that the sequence $\eta = g(T)\zeta$ is independent of at least one shift register $A_j \in \{A_0, \ldots, A_{12}\}$. That means, that there is a polynomial $h \in \mathbb{F}_2[x]$ and a keystream generator $\mathrm{KSG}(F'; A_{i_1}, \ldots, A_{i_k})$ producing a keystream ζ' such that $g(T)\zeta = h(T)\zeta'$, where F' has algebraic degree ≤ 4 and $\{A_{i_1}, \ldots, A_{i_k}\}$ is a proper subset of $\{A_0, \ldots, A_{12}\}$ which does not contain A_j. Select a product sequence from (20) which contains σ_j as a factor. Spot the dominator of the chosen product sequence in Table 7. For convenience of argumentation assume that $j = 0$. Then we have to consider the dominator $w_1 = \mathrm{dom}(\sigma_0\sigma_5\sigma_8\sigma_{10})$.

The dominator w_1 is the product of distinct binary irreducible polynomials of degree 490854. Set $\eta = g(T)\zeta$ and $\eta' = h(T)\zeta'$. We examine the minimal polynomials of η and η'. By Proposition 5A we have

$$m_\eta = \frac{m_\zeta}{\gcd(m_\zeta, g)}.$$

Dominator w_k	Degree of irreducible factors of w_k	Lower bound for $\deg(w_k)$
$w_1 = \mathrm{dom}(\sigma_0\sigma_5\sigma_8\sigma_{10})$	490 854	$2^{95.36}$
$w_2 = \mathrm{dom}(\sigma_1\sigma_4\sigma_8\sigma_{10})$	494 450	$2^{96.36}$
$w_3 = \mathrm{dom}(\sigma_2\sigma_3\sigma_4\sigma_8)$	400 200	$2^{96.57}$
$w_4 = \mathrm{dom}(\sigma_2\sigma_4\sigma_7\sigma_8)$	466 900	$2^{97.37}$
$w_5 = \mathrm{dom}(\sigma_2\sigma_4\sigma_7\sigma_{12})$	531 300	$2^{97.37}$
$w_6 = \mathrm{dom}(\sigma_2\sigma_4\sigma_8\sigma_{10})$	516 925	$2^{97.37}$
$w_7 = \mathrm{dom}(\sigma_2\sigma_4\sigma_8\sigma_{11})$	533 600	$2^{97.37}$
$w_8 = \mathrm{dom}(\sigma_2\sigma_4\sigma_{10}\sigma_{12})$	588 225	$2^{97.37}$
$w_9 = \mathrm{dom}(\sigma_2\sigma_4\sigma_{11}\sigma_{12})$	607 200	$2^{97.37}$
$w_{10} = \mathrm{dom}(\sigma_2\sigma_8\sigma_9\sigma_{10})$	620 310	$2^{97.37}$
$w_{11} = \mathrm{dom}(\sigma_5\sigma_6\sigma_8\sigma_{10})$	631 098	$2^{99.16}$

Table 7: Properties of the dominator of the product sequences

Since w_1 divides m_ζ, $\deg(w_1) > 2^{95.36}$, and $\deg(g) < 2^{95}$, we conclude that $\gcd(m_\eta, w_1) > 1$. It follows that the minimal polynomial m_η contains an irreducible factor of degree 490854. By Proposition $5A$,

$$m_{\eta'} = \frac{m_{\zeta'}}{\gcd(m_{\zeta'}, h)}\,.$$

In particular, $m_{\eta'}$ divides $m_{\zeta'}$. It follows from Lemma 6, and from Lemma 7 and Lemma 3, that the minimal polynomial of ζ' does not contain any irreducible factors of degree 490854. Thus, the polynomials m_η and $m_{\eta'}$ are different. This, of course, implies that the sequences η and η' are different and we arrive at the contradiction $g(T)\zeta \neq g(T)\zeta'$. This completes the proof of the assertion concerning the keystream of ACHTERBAHN-128. The proof for ACHTERBAHN-80 uses exactly the same arguments. □

Remark. The dominators $w_1, \ldots, w_{11}$ of the eleven sequences in (20) are pairwise relatively prime divisors of the minimal polynomial of the keystream ζ. Therefore, the product $w = w_1 \cdots w_{11}$ divides m_ζ, and $\deg(w)$ is a lower bound for the linear complexity of the keystream ζ of ACHTERBAHN-128. Thus, the sum of the entries in the last column of Table 7 yields the (slightly improved) lower bound

$$L(\zeta) \geq 2^{100.92}.$$

Regarding ACHTERBAHN-80 we must take into account that only the polynomials $w_2, w_3, w_4, w_6, w_7, w_{10}$, and w_{11} are divisors of the minimal polynomial of the keystream ζ. We obtain the lower bound

$$L(\zeta) \geq 2^{100.46}.$$

5 Analysis

5.1 Algebraic attacks

If we relate the bits of the initial state of the keystream generator of ACHTERBAHN-128 to the bits of the produced keystream, we obtain a system of multivariate polynomial equations in 351 unknowns of algebraic degree 114. In fact, the lengths of the thirteen shift registers in the keystream generator run through the integer values between 21 and 33, so that the size of the internal state is $13(21+33)/2 = 351$ bits, which explains the number of unknowns. To explain the asserted value for the degree of the algebraic equations we first consider a single primitive binary FSR of length N.

Let the initial state of the shift register be given by $(s_0, s_1, \ldots, s_{N-1})$. If the shift register is linear, then each output bit s_n, $n \geq 0$, of the shift register is the sum of a certain number of initial state bits taken from the set $\{s_0, s_1, \ldots, s_{N-1}\}$.[6] If the shift register is nonlinear then the output bit s_n, $n \geq 0$, is the sum of monomials taken from the set

$$\{s_0, s_1, \ldots, s_{N-1}, s_0s_1, s_0s_2, \ldots, s_{N-2}s_{N-1}, \ldots, s_1s_2\cdots s_{N-1}\},$$

where, for most shift registers, each monomial of the set will occur in the representation of some s_n. The set has cardinality $2^N - 2$ and contains all monomials that can be formed out of the initial state bits $s_0, s_1, \ldots, s_{N-1}$ except the two monomials 1 and $s_0s_1\cdots s_{N-1}$. The monomial 1 does not occur because the feedback function $A(x_0, \ldots, x_{N-1})$ of the NLFSR has the property $A(0, \ldots, 0) = 0$. The monomial $s_0s_1\cdots s_{N-1}$ of degree N does not occur because the feedback function A is balanced. The fact that all the other $2^N - 2$ monomials will occur in the representation of some s_n is not guaranteed for every nonlinear binary FSR. But it is a typical property that most NLFSR's have.

Consider the 4-stage primitive LFSR given by $L(x_0, x_1, x_2, x_3) = x_0 + x_1$. Let the initial state of the shift register be (a, b, c, d). The output bits of the shift register appearing in the first period are

$$a,\ b,\ c,\ d,\ a+b,\ b+c,\ c+d,\ a+b+d,\ a+c,\ b+d,\ a+b+c,\ b+c+d,$$
$$a+b+c+d,\ a+c+d,\ a+d.$$

Now consider the 4-stage primitive NLFSR given by $A(x_0, x_1, x_2, x_3) = x_0 + x_1 + x_2 + x_1x_3$. With the same initial state the output bits of the shift register appearing in the first period are

$$a,\ b,\ c,\ d,\ a+b+c+bd,\ b+d+ac+bc+bcd,\ a+b+acd,\ b+c+abd+bcd,$$
$$c+d+cd+abc+acd+bcd,\ a+b+c+d+ad+bd+abd+acd,$$
$$a+b+c+d+ab+ac+abc+abd+bcd,\ a+d+bc+cd+abc+acd,$$
$$a+c+ad+bd+abd,\ c+d+ab+ac+bd+abc,\ a+b+d+ac.$$

[6] If $f \in \mathbb{F}_2[x]$ is the characteristic polynomial of the linear feedback shift register and $x^n \equiv a_{N-1}x^{N-1} + \cdots + a_1x + a_0 \mod f(x)$, then $s_n = a_{N-1}s_{N-1} + \cdots + a_1s_1 + a_0s_0$.

We call the sequence of the first $2^N - 1$ output bits of a binary N-stage primitive feedback shift register, where each output bit is expressed as a multivariate polynomial in the initial state bits, the *monomial spectrum* of the shift register.[7] Figure 6 displays the monomial spectra of the two primitive feedback shift registers under discussion graphically.

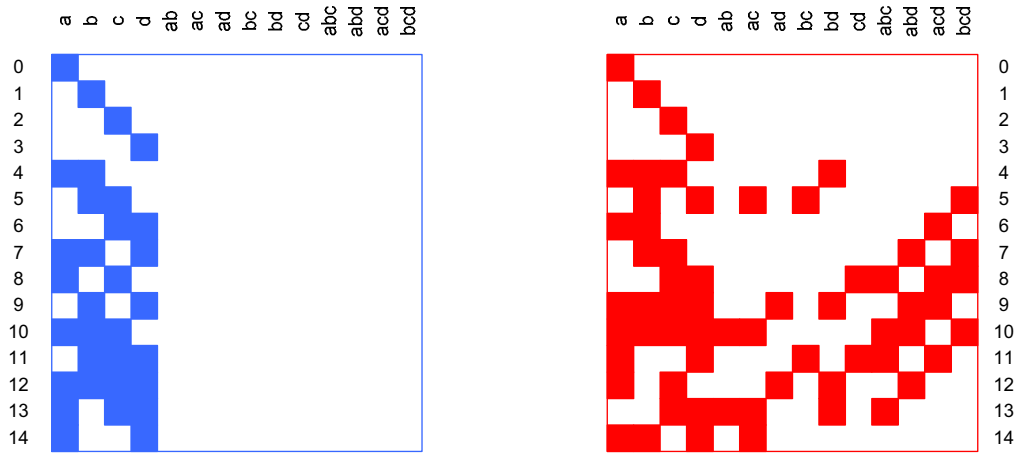


Figure 6: Comparison between primitive LFSR and primitive NLFSR

The following property of the monomial spectra of the shift registers $A_0, A_1, \ldots, A_{12}$ deployed in the keystream generator of ACHTERBAHN-128/80 has been investigated via computer calculations.

Fact 5. *For $2N_j \leq k \leq 2^{N_j} - N_j$, the kth entry in the monomial spectrum of A_j contains close to $2^{N_j - 1}$ different monomials and has in general degree $N_j - 1$.*

So much about a single primitive nonlinear feedback shift register. We have now to explore the effect of the Boolean combining function F. The algebraic normal form of F contains 48 monomials of degree 4, among them the monomial $x_5x_6x_{11}x_{12}$ (compare Section 3.2). Let σ_j, $0 \leq j \leq 12$, denote the nonzero output sequence of the shift register A_j corresponding to the given respective initial state. The sequence $\sigma_5\sigma_6\sigma_{11}\sigma_{12}$ is one component of the keystream ζ, which is the sum of 124 periodic sequences (see (16)). If we want to express the terms of the sequence $\sigma_5\sigma_6\sigma_{11}\sigma_{12}$ in terms of the initial state bits of the shift registers A_5, A_6, A_{11}, and A_{12}, then, by Fact 5, this will lead to a system of multivariate polynomial equations in which most equations have degree

$$(N_5 - 1) + (N_6 - 1) + (N_{11} - 1) + (N_{12} - 1) = 25 + 26 + 31 + 32 = 114.$$

[7] More precisely, let A be primitive binary FSR of length N. For each $n \geq 0$, the shift register A induces a Boolean function $f_n : \mathbb{F}_2^N \to \mathbb{F}_2$, which maps the initial state $(s_0, s_1, \ldots, s_{N-1})$ of the shift register A to the bit s_n of the corresponding output sequence $(s_n)_{n=0}^{\infty}$. The monomial spectrum of A is defined to be the sequence $(f_0, f_1, \ldots, f_{p-1})$, where each Boolean function $f_n = f_n(s_0, s_1, \ldots, s_{N-1})$, $0 \leq n \leq p-1$, is represented by its algebraic normal form and where $p = 2^N - 1$.

Since we need $2^{N_j}-2$ different monomials, formed by the N_j initial state bits of the shift register A_j, in order to express the bits of the sequence σ_j by the bits of the initial state of A_j, it follows that we need

$$(2^{N_5}-2)(2^{N_6}-2)(2^{N_{11}}-2)(2^{N_{12}}-2) \approx 2^{118}$$

different monomials in order to express the bits of the sequence $\sigma_5\sigma_6\sigma_{11}\sigma_{12}$ by the initial state bits of the four shift registers A_5, A_6, A_{11}, and A_{12}. Taking into account the effect of the other 123 monomial terms in the algebraic normal form of F, we conclude that we need more than 2^{120} different monomials, formed by the 351 initial state bits of the keystream generator, in order to express the keystream bits by the initial state bits. The estimate, however, relies on the fact that the combining function F has (maximum) algebraic immunity 4. If this was not the case, then there would exist a polynomial E in $R = \mathbb{F}_2[x_0,\ldots,x_{12}]/(x_0^2-x_0,\ldots,x_{12}^2-x_{12})$ such that $E*F=0$ or $E*F$ is a nonzero polynomial in R of degree <4. This could be exploited in the way described in [5] to reduce the complexity of the attack significantly.

The above system of algebraic equations can be solved by linearization: Each monomial is replaced by a new independent variable. This turns the system of algebraic equations of degree 114 in 351 unknowns into a system of linear equations in more than 2^{120} unknowns. To set up the system of linear equations one needs at least 2^{120} keystream bits. Since we restricted the frame length to 2^{64} (this is necessary to counter correlation attacks), the task to set up and solve the system of equations is impossible.

Let us assume that the attacker is given more than 2^{120} keystream bits so that he can set up the system of linear equations. The complexity for solving the system of equations is $O((2^{120})^{\omega}) = O(2^{285})$, where $\omega \approx 2.38$ is the exponent of fast matrix multiplication.

If we apply the above arguments to ACHTERBAHN-80, we find that the system of algebraic equations relating the keystream bits to the initial state bits has degree 110 and contains 297 unknowns. The corresponding system of linear equations has more than 2^{116} unknowns and can be solved in time $O(2^{276})$.

We collect the results in the following Fact.

Fact 6. *The time complexity for computing the initial state bits of the keystream generator from a sufficient number of keystream bits is* $O(2^{285})$ *for* ACHTERBAHN-128*, and* $O(2^{276})$ *for* ACHTERBAHN-80*. More than* 2^{120} *or* 2^{116}*, respectively, keystream bits are required to perform the task.*

At the beginning of this section we compared the monomial spectra of two primitive FSR's, one being linear the other one nonlinear. It is interesting to compare the monomial spectra of two nonlinear feedback shift registers, one being primitive the other one nonprimitive. Consider the 4-stage nonlinear and nonprimitive FSR defined by $B(x_0,x_1,x_2,x_3) = x_0+x_1+x_3+x_2x_3$. The shift register has four different cycles: (0), (01), (001), (0001101111). Thus the shift register can produce four translation distinct sequences of respective least periods 1, 2, 3, and 10. It follows that $\mathrm{lcm}(1,2,3,10) = 30$ is the least common period for all output sequences of the shift register. We call the least common multiple of the least periods of all possible

output sequences of an arbitrary FSR the *period* of the shift register. (In the case that the shift register is primitive the definition is compatible with Definition 2.) Thus, shift register B has period $p = 30$. The definition of the monomial spectrum of a primitive FSR given above can now be extended to an arbitrary binary FSR: The monomial spectrum of a binary FSR of period p is the sequence of the first p output bits of the shift register expressed in terms of the initial state bits.

The monomial spectrum of shift register B is displayed in Figure 7. If we compare the monomial spectrum of the nonprimitive shift register B with the monomial spectrum of the primitive shift register A in Figure 6, we observe that the simple cycle structure of A implies a more compact monomial spectrum. If we linearize the system of algebraic equations obtained by relating the output bits in the first period of the shift register with the initial state bits of the shift register, then A gives rise to a slightly overdefined system of linear equations (15 equations in 14 unknowns), whereas B gives rise to a strongly overdefined system of linear equations (30 equations in 14 unknowns).

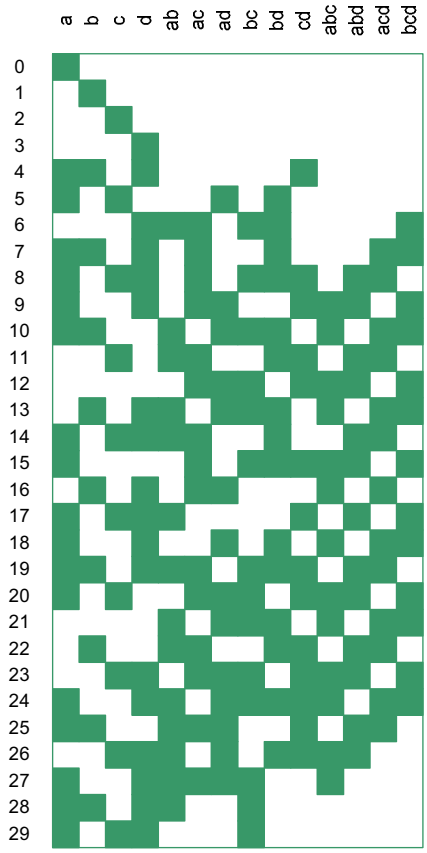


Figure 7: Monomial spectrum of a non-primitive NLFSR

5.2 Correlation attacks

We first consider the classical correlation attack of Siegenthaler [32]. The Boolean combining function G of ACHTERBAHN-80 is correlation immune of order 6. In order to mount a correlation attack of the type described in [32], the attacker must consider at least seven shift registers simultaneously. The sum of the lengths of the shortest seven FSR's of the keystream generator of ACHTERBAHN-80 is 175. Therefore, the complexity of Siegenthaler's correlation attack against ACHTERBAHN-80 is at least $O(2^{175})$. The Boolean combining function F of ACHTERBAHN-128 has order of resiliency 8. It follows that the complexity of Siegenthaler's attack is greater than $O(2^{225})$.

A divide an conquer attack against Achterbahn-1 has been described in [18], [19]. The idea of the attack is to set some variables of the combining function to 0 or 1 and use the obtained subfunction in the attack. The attack will be successful under the following two conditions: (i) The sum of the lengths of the shift registers which correspond to the variables that are set to constant values must be less than the key length of the cipher. (ii) The obtained subfunction should be affine or quadratic and the number of monomials in the algebraic normal form of the subfunction should not be greater than the base-2 logarithm of the largest shift register length. If the conditions are not fulfilled the attack will not work or have a higher complexity than exhaustive key search. ACHTERBAHN-80 and ACHTERBAHN-128 are not threatened by this divide and conquer attack. One simply cannot derive subfunctions from the combining functions G and F of ACHTERBAHN-80 and ACHTERBAHN-128, respectively, that are suitable for the attack.

A time-memory trade-off attack was mounted against a modified version of Achterbahn-1 exploiting the fact that the combining function contained four of its eight variables linearly [19]. ACHTERBAHN-80 and ACHTERBAHN-128 are immune against this attack since their respective combining functions depend nonlinearly on all of its variables. (Compare the discussion in Section 4.3.)

A guess and determine attack

The attack was described in [19]. We explain the attack for the Boolean combining function of Achterbahn-1 which was given by

$$R(x_1, \ldots, x_8) = x_1 + x_2 + x_3 + x_4 + x_5x_7 + x_6x_7 + x_6x_8 + x_5x_6x_7 + x_6x_7x_8.$$

The function R agrees with the linear function

$$L(x_1, \ldots, x_8) = x_1 + x_2 + x_3 + x_4 + x_6$$

with probability $p = 3/4$. That is, we have $R(\mathbf{x}) = L(\mathbf{x})$ for 192 vectors $\mathbf{x} \in \mathbb{F}_2^8$.

Consider the keystream $\zeta = R(\sigma_1, \ldots, \sigma_8)$. The sequences $\sigma_1, \ldots, \sigma_8$ are the output sequences of the eight driving feedback shift registers. Each sequence σ_j, $1 \leq j \leq 8$, is uniquely determined by its initial state vector. The attacker aims to determine the initial state vector of a sequence σ_j.[8] Since $R(\mathbf{x}) = L(\mathbf{x})$ with

[8] In the case of Achterbahn-1 the knowledge of the initial state of one shift register was sufficient to recover the secret key. For ACHTERBAHN-128/80 it is not possible to recover the secret key from the known initial state of one shift register.

probability 3/4, it follows that the nth term, $n \geq 0$, of the sequence ζ agrees with the nth term of the sequence $\sigma_1 + \sigma_2 + \sigma_3 + \sigma_4 + \sigma_6$ with probability $p = 3/4$. Write $p = \frac{1}{2}(1 + \varepsilon) = \frac{1}{2}(1 + \frac{1}{2})$, so that ε is the correlation coefficient of R and L.

We use the notation

$$\zeta \overset{\varepsilon=1/2}{\approx} \sigma_1 + \sigma_2 + \sigma_3 + \sigma_4 + \sigma_6 \tag{21}$$

to express the fact that the terms of the two sequences agree with probability $p = \frac{1}{2}(1 + \varepsilon)$, where $\varepsilon = 1/2$.

Approximation (21) is equivalent to

$$\zeta + \sigma_1 \overset{\varepsilon=1/2}{\approx} \sigma_2 + \sigma_3 + \sigma_4 + \sigma_6. \tag{22}$$

Let p_j denote the least period of σ_j. Then the polynomial

$$g(x) = (x^{p_2} - 1)(x^{p_3} - 1)(x^{p_4} - 1)(x^{p_6} - 1)$$

is a characteristic polynomial of $\sigma_2 + \sigma_3 + \sigma_4 + \sigma_6$. Apply the linear operator $g(T)$ to both sides of (22). The right-hand side vanishes and we obtain

$$g(T)[\zeta + \sigma_1] \overset{\varepsilon'=\varepsilon^{16}}{\approx} \mathbf{0}, \tag{23}$$

where $\mathbf{0}$ denotes the zero sequence. The polynomial g has $2^4 = 16$ terms, so that the application of $g(T)$ to a sequence ϱ means that we add together termwise 16 shifted versions of the sequence ϱ. It is assumed—in analogy to the piling-up lemma—that if the original sequence is biased with ε, the new sequence $g(T)\varrho$ will be biased with $\varepsilon' = \varepsilon^{16}$. (Simulation results suggest that this is a reasonable assumption despite the fact that the requirements for an application of the piling-up lemma are not fulfilled.) The idea of the attack is to guess the sequence σ_1. The corresponding shift register has length $N_1 = 22$, so that there are 2^{22} possible initial states. Because of Step 5 of the key-loading algorithm only 2^{21} initial states are possible candidates.

For each possible sequence σ_1 one analyzes the sequence

$$g(T)[\zeta + \sigma_1] = g(T)\zeta + g(T)\sigma_1. \tag{24}$$

For the correct value of σ_1, the sequence will be biased with $\varepsilon' = 2^{-16}$. For a wrongly guessed σ_1 the sequence in (24) should be unbiased. The authors of [19] assumed that by investigating $(1/\varepsilon')^2 = 2^{32}$ terms of the sequence in (24), the correct initial state vector of the sequence can be identified. This is not true. The required number of sequence bits has to be larger—by about the factor of 16—in order to identify the correct sequence σ_1 with high probability. Thus one has to compute at least $2^4(\varepsilon')^{-2} = 2^{36}$ bits of the sequence in (24). The complexity of this step is $O(2^{36})$. For each of the 2^{21} candidate sequences σ_1, the sequence in (24) must be computed and analyzed. Thus the complexity of the attack is of the order $2^{21} \cdot 2^{36} = 2^{57}$.

Let us consider the effect of the guess and determine attack against ACHTERBAHN-80. The best linear approximation of the combining function G of ACHTERBAHN-80 is given by

$$L(x_1, \ldots, x_{11}) = x_1 + x_3 + x_4 + x_5 + x_6 + x_7 + x_{10}.$$

The correlation coefficient of L and G is $\varepsilon = -1/8$. Thus, for the keystream $\zeta = G(\sigma_1, \ldots, \sigma_{11})$, we have

$$\zeta \overset{\varepsilon=-1/8}{\approx} \sigma_1 + \sigma_3 + \sigma_4 + \sigma_5 + \sigma_6 + \sigma_7 + \sigma_{10}. \tag{25}$$

The best approach for the attack is to guess the three sequences σ_1, σ_3, and σ_4. There are 2^{68} possibilities. Write (25) in the equivalent form

$$\zeta + \sigma_1 + \sigma_3 + \sigma_4 \overset{\varepsilon=-1/8}{\approx} \sigma_5 + \sigma_6 + \sigma_7 + \sigma_{10},$$

and apply the linear operator $g(T)$ with

$$g(x) = (x^{p_5} - 1)(x^{p_6} - 1)(x^{p_7} - 1)(x^{p_{10}} - 1)$$

to both sides. Then for the correct guess $(\sigma_1, \sigma_3, \sigma_4)$, the sequence

$$g(T)[\zeta + \sigma_1 + \sigma_3 + \sigma_4] = g(T)\zeta + g(T)\sigma_1 + g(T)\sigma_3 + g(T)\sigma_4$$

will be biased with $\varepsilon' = \varepsilon^{2^4} = 2^{-48}$. We need to process 2^{100} terms of the sequence. The overall complexity of the attack is $O(2^{168})$.

The guess and determine attack is more threatening in conjunction with quadratic approximations. Consider the Boolean function

$$Q_1(x_1, \ldots, x_{11}) = x_1 + x_2 + x_3 + x_4x_7 + x_9x_{10}. \tag{26}$$

We have $Q_1(\mathbf{x}) = G(\mathbf{x})$ for 1056 vectors $\mathbf{x} \in \mathbb{F}_2^{11}$. In other words, the functions Q_1 and G agree with probability $p = \frac{1}{2}(1 + \frac{1}{32})$. We have

$$\zeta + \sigma_1 + \sigma_2 + \sigma_3 \overset{\varepsilon=2^{-5}}{\approx} \sigma_4\sigma_7 + \sigma_9\sigma_{10}. \tag{27}$$

Let $r_1 = \mathrm{per}(\sigma_4\sigma_7) = (2^{25} - 1)(2^{28} - 1)$ and $r_2 = \mathrm{per}(\sigma_9\sigma_{10}) = (2^{30} - 1)(2^{31} - 1)$. Compute

$$g(x) = (x^{r_1} - 1)(x^{r_2} - 1), \tag{28}$$

and apply $g(T)$ to both sides of (27) to obtain

$$g(T)[\zeta + \sigma_1 + \sigma_2 + \sigma_3] \overset{\varepsilon=2^{-20}}{\approx} \mathbf{0}.$$

We guess the sequences σ_1, σ_2, and σ_3 (2^{66} possibilities). We have to compute and to analyze $2^4(2^{20})^2 = 2^{44}$ terms of the sequence, so that the overall complexity of the attack is $O(2^{110})$.

For ACHTERBAHN-128 the most favorable approximation for the attack is given by the Boolean function

$$Q_2(x_0, x_1, \ldots, x_{12}) = x_0 + x_1 + x_2 + x_3 + x_4 + x_7x_{10} + x_8x_9. \tag{29}$$

The function Q_2 is correlated to the combining function F with $\varepsilon = 1/32$. The best strategy would be to guess the five sequences σ_0, σ_1, σ_2, σ_3, and σ_4. The time complexity of the attack in this case is $O(2^{154})$.

For ACHTERBAHN-80 cubic approximations are not relevant, since the least period of the product of any three sequences σ_j, $1 \le j \le 11$, is greater than 2^{64}, the frame length. For ACHTERBAHN-128, however, there are four possibilities that the product of three sequences σ_j, $0 \le j \le 12$, has least period $< 2^{64}$. These are the sequences

$$\sigma_0\sigma_1\sigma_3, \quad \sigma_0\sigma_1\sigma_7, \quad \sigma_0\sigma_1\sigma_{12}, \quad \sigma_1\sigma_3\sigma_{12}$$

whose least periods are slightly greater than $2^{62.60}$, $2^{62.42}$, $2^{62.19}$, and $2^{63.60}$, respectively. Therefore, in the case of ACHTERBAHN-128 we also have to consider cubic approximations. Only cubic approximations that contain any of the monomials $x_0x_1x_3$, $x_0x_1x_7$, $x_0x_1x_{12}$, $x_1x_3x_{12}$ are of interest. Cubic Boolean functions that are uncorrelated to F, or that contain other monomials of degree 3 are irrelevant. All cubic approximations that are relevant have correlation coefficients $\varepsilon \le 2^{-6}$ with F. One such cubic approximation is given by

$$C(x_0, x_1, \ldots, x_{12}) = x_4 + x_5 + x_6 + x_7 + x_8x_{10} + x_0x_1x_3.$$

Here we have $\varepsilon = 2^{-6}$. The best strategy is to guess the first four sequences. The time complexity of the attack is $O(2^{154})$.

Another guess and determine attack

This is a refinement of the above discussed guess and determine attack. The attack appeared in [15]. In the article [15], a stream cipher concept is investigated that has been announced in [11] but never got fully specified. The claim made by the authors of [15] that they can determine the initial state of the shortest FSR of that unspecified cipher from $2^{59.02}$ keystream bits is wrong. One needs more than 2^{63} keystream bits (the declared frame length for the announced cipher was 2^{63}). We shall discuss the issue in more detail in a separate paper.

Reconsider the quadratic approximation Q_1 for the combining function G in (26). Let $g(x) = (x^{r_1} - 1)(x^{r_2} - 1)$ be the polynomial in (28). We have

$$\zeta \overset{\varepsilon=2^{-5}}{\approx} \sigma_1 + \sigma_2 + \sigma_3 + \sigma_4\sigma_7 + \sigma_9\sigma_{10}.$$

Applying $g(T)$ to both sides gives

$$g(T)\zeta \overset{\varepsilon=2^{-20}}{\approx} g(T)\sigma_1 + g(T)\sigma_2 + g(T)\sigma_3. \tag{30}$$

For $r \ge 1$, the decimation operator D_r is a linear operator on $\mathbb{F}_2^\infty$ defined by $D_r\sigma = (s_{nr})_{n=0}^\infty$ for all $\sigma = (s_n)_{n=0}^\infty$ in $\mathbb{F}_2^\infty$. Let p_1 be the least period of σ_1. Consider the decimation operator D_{p_1}. For ease of notation set $D_{p_1} = D$, and apply D to both sides of (30). This yields

$$Dg(T)\zeta \overset{\varepsilon=2^{-20}}{\approx} Dg(T)\sigma_1 + Dg(T)\sigma_2 + Dg(T)\sigma_3. \tag{31}$$

We have

$$g(T)\sigma_1 = T^{r_1+r_2}\sigma_1 + T^{r_1}\sigma_1 + T^{r_2}\sigma_1 + \sigma_1.$$

Since $p_1 = \mathrm{per}(\sigma_1)$, the sequence $D\sigma_1$ is constant. It follows that the sequence $Dg(T)\sigma_1$ is constant. Hence (31) is equivalent to

$$Dg(T)[\zeta + \sigma_2 + \sigma_3] \overset{\varepsilon=2^{-20}}{\approx} \text{constant sequence}.$$

The strategy is again to guess the sequences σ_2 and σ_3. There are 2^{45} possibilities for (σ_2, σ_3). In order to determine the correct pair (σ_2, σ_3) one needs 2^{44} terms of the sequence $Dg(T)\zeta$. That is, one needs $p_1 2^{44} = (2^{22} - 1)2^{44} \approx 2^{66}$ terms of the sequence $g(T)\zeta$, and at least that many terms of the sequence ζ. Since the frame length of ACHTERBAHN-80 is 2^{64}, the attack cannot be carried out. If the attacker was given more than 2^{66} keystream bits, the time complexity of the attack would be $O(2^{89})$.

If we apply the decimation attack to the keystream of ACHTERBAHN-128 using the quadratic approximation in (29), we find that the attack requires more than 2^{65} keystream bits. If they were made available to the attacker, the complexity of the attack would be $O(2^{134})$.

6 Implementation

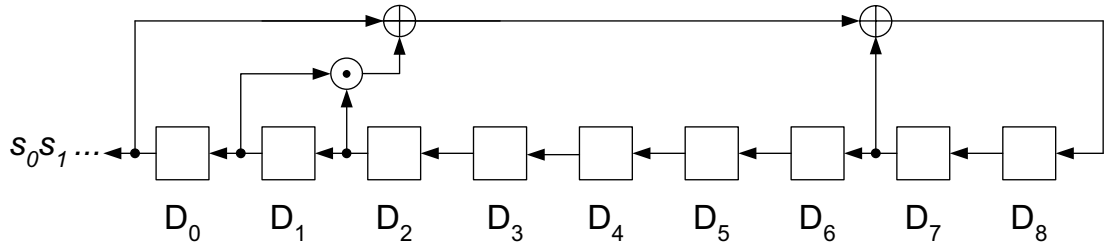
6.1 Parallel implementations

Achterbahn is a hardware oriented stream cipher design. In a straightforward implementation of the keystream generator one bit of keystream is produced per clock cycle. The speed of keystream generation, and hence the encryption speed, can be increased by a factor of 2, 4, or 8, respectively, by implementing the Boolean combining function several times and by using parallel implementations of the driving feedback shift registers. We shall speak of the 1-bit, 2-bit, 4-bit, and 8-bit implementation of the keystream generator, respectively, when one, two, four, or eight keystream bits are produced per clock cycle.

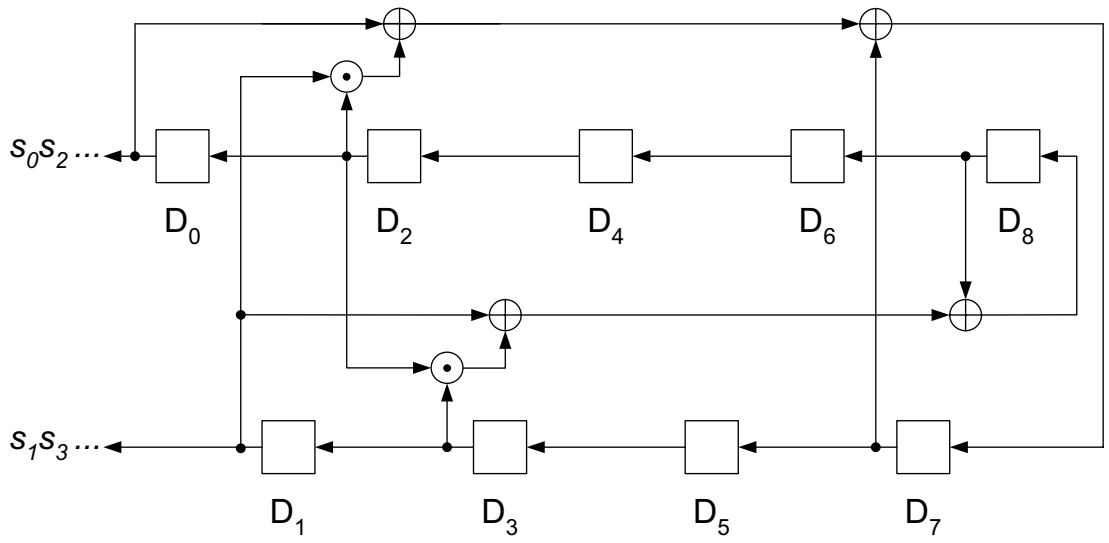
In a traditional implementation of a feedback shift register the content of cell D_k is shifted into cell D_{k-1} and the content of one memory cell is emitted. In a parallel implementation of the shift register, with degree of parallelization $q \geq 1$, the content of cell D_k is shifted into cell D_{k-q} and the contents of q memory cells (e.g., the contents of $D_0, D_1, \ldots, D_{q-1}$) are emitted at each clock pulse. The method is best explained in an example: Consider the 9-stage nonlinear FSR defined by the feedback function

$$A(x_0, x_1, \ldots, x_8) = x_0 + x_7 + x_1 x_2.$$

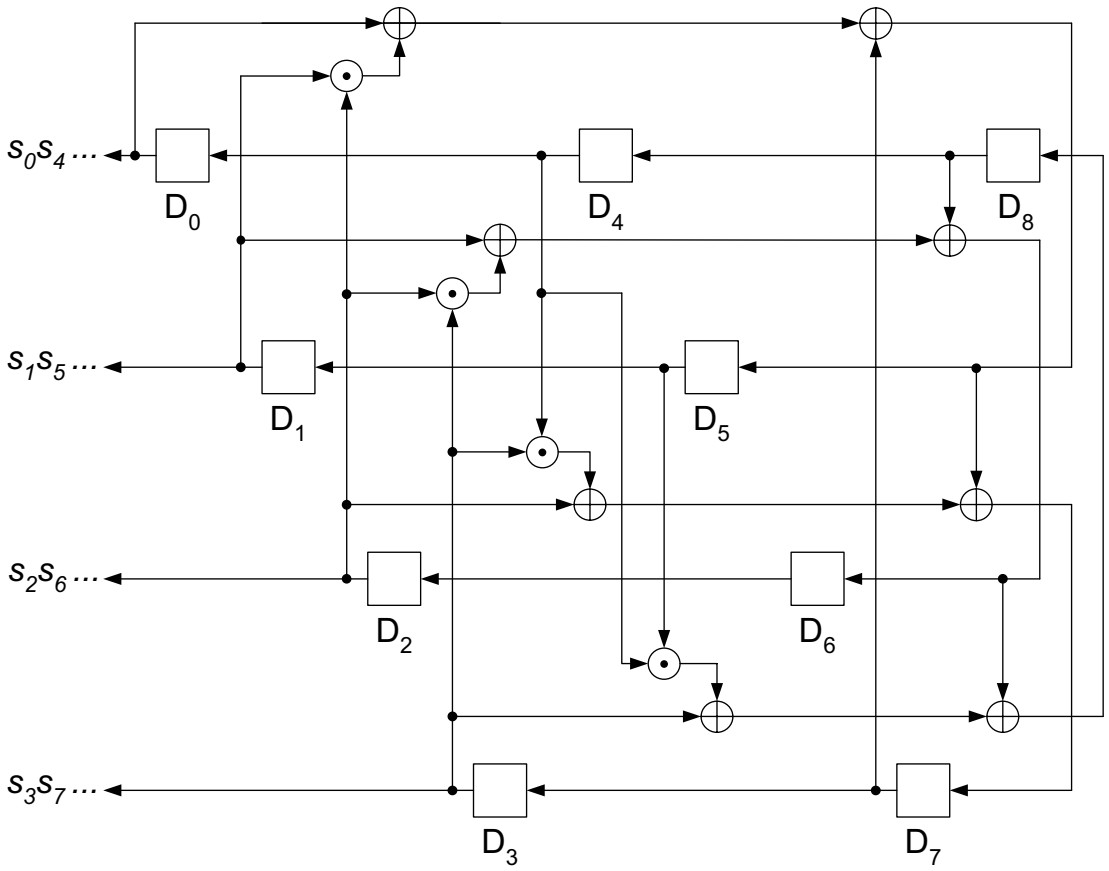
Let D_0 be the output cell of the shift register. Then the traditional implementation of the shift register looks like this:



A parallel implementation of the FSR with degree of parallelization $q = 2$ has the following form:



A 4-bit implementation of the shift register is given by:



It is important to select shift registers for which the logical depth of the feedback function is still relatively small in the intended parallel implementation. The shift registers A_j, $0 \leq j \leq 12$, specified in Section 3.4, were constructed in a way such that all feedback functions have the same logical depth. The logical depth is three for all shift registers in the 1-bit implementation, and grows to five in an 8-bit implementation of the shift register. In each feedback function $A_j(x_0, x_1, \ldots, x_{N_j-1})$, the last seven variables do not occur in any monomial of degree ≥ 2. (Most of these variables do not occur at all.) This is a measure to ensure that the logical depth will still be small in all parallel implementations up to the parallelization degree 8.

The increased performance of the underlying feedback shift registers results in a corresponding increase of speed of keystream generation provided that the Boolean combining function is implemented q times, were q is the degree of parallelization of the shift registers. If $q = 8$ is chosen, then the keystream generator produces one byte of keystream per clock cycle. If $\zeta = (z_n)_{n=0}^{\infty}$ is the keystream, then the first emitted byte has the form $(z_0, z_1, z_2, z_3, z_4, z_5, z_6, z_7)$, and the second byte is $(z_8, z_9, z_{10}, z_{11}, z_{12}, z_{13}, z_{14}, z_{15})$, and so on. It is important to note that in any parallel implementation of the keystream generator exactly the same keystream bits are produced as in the 1-bit implementation. A parallel implemention of the keystream does not only increase the encryption rate, it also cuts down resynchronization times.

6.2 Throughput and design size

We implemented the keystream generator of ACHTERBAHN-128/80 and of ACHTERBAHN-80 using high-level hardware description language VHDL and a standard synthesis compiler. We used a low-Vt 1.5V standard cell library targeting 130 nm CMOS technology. For the synthesis, worst case conditions (125° C junction temperature and 10% voltage drop) were assumed.[9] We present the synthesis results for two different implementations of the keystream generator of ACHTERBAHN-128/80 and ACHTERBAHN-80.

The first implementation aims to minimize hardware costs and has slightly higher resynchronization times. In this implementation all flip-flops of the keystream generator are without reset and scan functionality (see the discussion at the end of Section 2.4). In this implementation the first N_j key bits are loaded into each shift register bit by bit. We shall refer to this implementation of Achterbahn as the implementation *without SPA countermeasures*.

In the second implementation each shift register A_j contains 16 scan flip-flops. The first 16 key bits are loaded into each shift register A_j simultaneously within one clock cycle. We shall refer to this second implementation of the keystream generator as the implementation of Achterbahn *with SPA countermeasures*.

Table 8 contains the design sizes of the implementation of Achterbahn without SPA countermeasures for degrees of parallelization 1, 2, 4, and 8. The design sizes are given in NAND gate equivalents (GE). The Table contains also values for the hardware efficiency for the various parallel implementations. The hardware efficiency is defined as the number of keystream bits produced per clock cycle divided by the design size in units of 1000 GE. Table 9 contains the synthesis results for the implementation of Achterbahn with SPA countermeasures.

	ACHTERBAHN-128/80		**ACHTERBAHN-80**	
	Design size in GE	Hardware efficiency	Design size in GE	Hardware efficiency
1-bit implementation	2538	0.39	2188	0.46
2-bit implementation	3058	0.65	2633	0.76
4-bit implementation	4082	0.98	3518	1.14
8-bit implementation	6183	1.29	5315	1.51

Table 8: Synthesis results for Achterbahn without SPA counter measures

.

The design sizes listed in the tables are valid for operating frequencies up to 400 MHz. For higher frequencies the design size is a function of the required throughput. For instance, the 8-bit implementation of ACHTERBAHN-80 without SPA countermeasures has design size 5447 GE at a throughput of 2 Gbit/s, and a design size of 8651 GE at a throughput of 8 Gbit/s.

[9] Under typical conditions (25° C junction temperature at 1.5 V) for high operating frequencies the obtained design sizes are about 10% smaller.

	ACHTERBAHN-128/80		ACHTERBAHN-80	
	Design size in GE	Hardware efficiency	Design size in GE	Hardware efficiency
1-bit implementation	2892	0.34	2476	0.40
2-bit implementation	3344	0.60	2872	0.70
4-bit implementation	4342	0.92	3714	1.08
8-bit implementation	6349	1.26	5446	1.47

Table 9: Synthesis results for Achterbahn with SPA counter measures

Figure 8 shows the design sizes for the various parallel implementations of ACHTERBAHN-128/80 without SPA counter measures in the throughput range from 1 Mbit/s to 8 Gbit/s. Each plotted symbol corresponds to a synthesis result. The synthesis results for the 1-bit implementation are indicated by circles. The synthesis results for the 2-bit, 4-bit, and 8-bit implementation are indicated by diamonds, squares, and triangles, respectively. Filled symbols correspond to synthesis results in which the Boolean combining function has been pipelined.

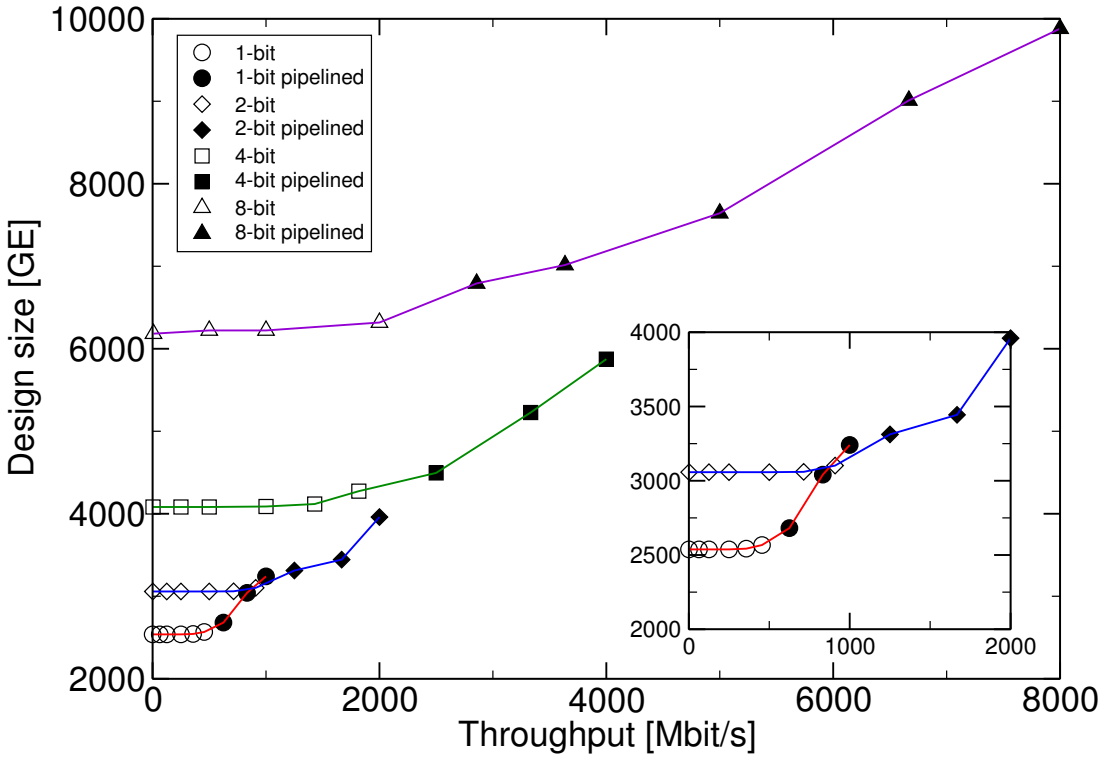


Figure 8: Throughput versus design size for ACHTERBAHN-128/80 without SPA countermeasures

Since the feedback shift registers A_j have by construction a small logical depth (three in the 1-bit implementation, five in the 8-bit implementation), the critical path of the design lies in the Boolean combining function which has logical depth

nine. If the operating frequency is increased beyond a certain value, it makes sense to pipeline the Boolean combining function. For the 1-bit implementation, pipelining is appropriate if the required throughput becomes greater than 500 Mbit/s. It is only necessary to insert one pipeline stage into the combining function which costs 7 flip-flops in the 1-bit implementation. In the 2-bit, 4-bit, and 8-bit implementation, one needs 14, 28, and 56 flip-flops, respectively.

At a throughput of about 900 Mbit/s the design size of the pipelined 1-bit implementation reaches the design size of the unpipelined 2-bit implementation. At a throughput of approximately 2 Gbit/s the design size of the 2-bit implementation reaches the design size of the 4-bit implementation. At about 4.2 Gbit/s the design size of the 4-bit implementation reaches the design size of the 8-bit implementation.

Figure 9 gives the design size/throughput results for ACHTERBAHN-80 without SPA countermeasures. The synthesis results for the implementations with SPA counter measures can be found in Figure 10. The design size of ACHTERBAHN-80 lies between the design size of the reduced version of Achterbahn-1 and the full version of Achterbahn-1 (see [9, p. 28]). The performance values at high frequencies are slightly better for ACHTERBAHN-80 due to the more efficient FSR's.[10]

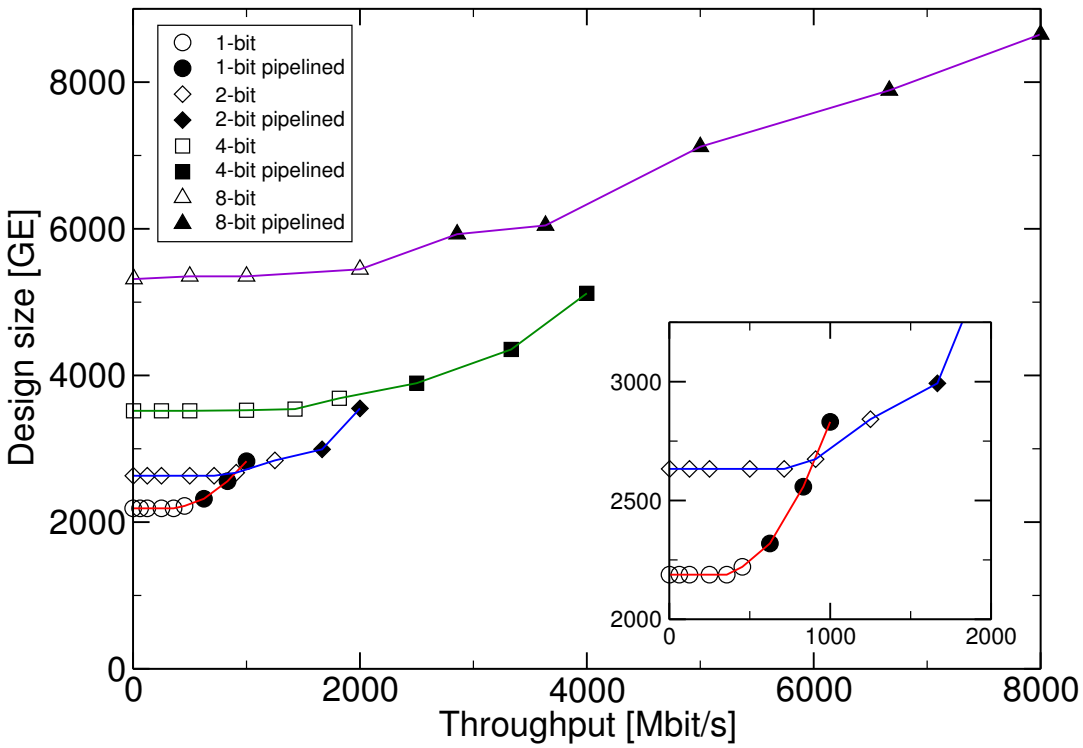


Figure 9: Throughput versus design size for ACHTERBAHN-80 without SPA countermeasures

[10]Incorrect results regarding the efficiency of the parallel implementations of Achterbahn-1 have been reported in [13].

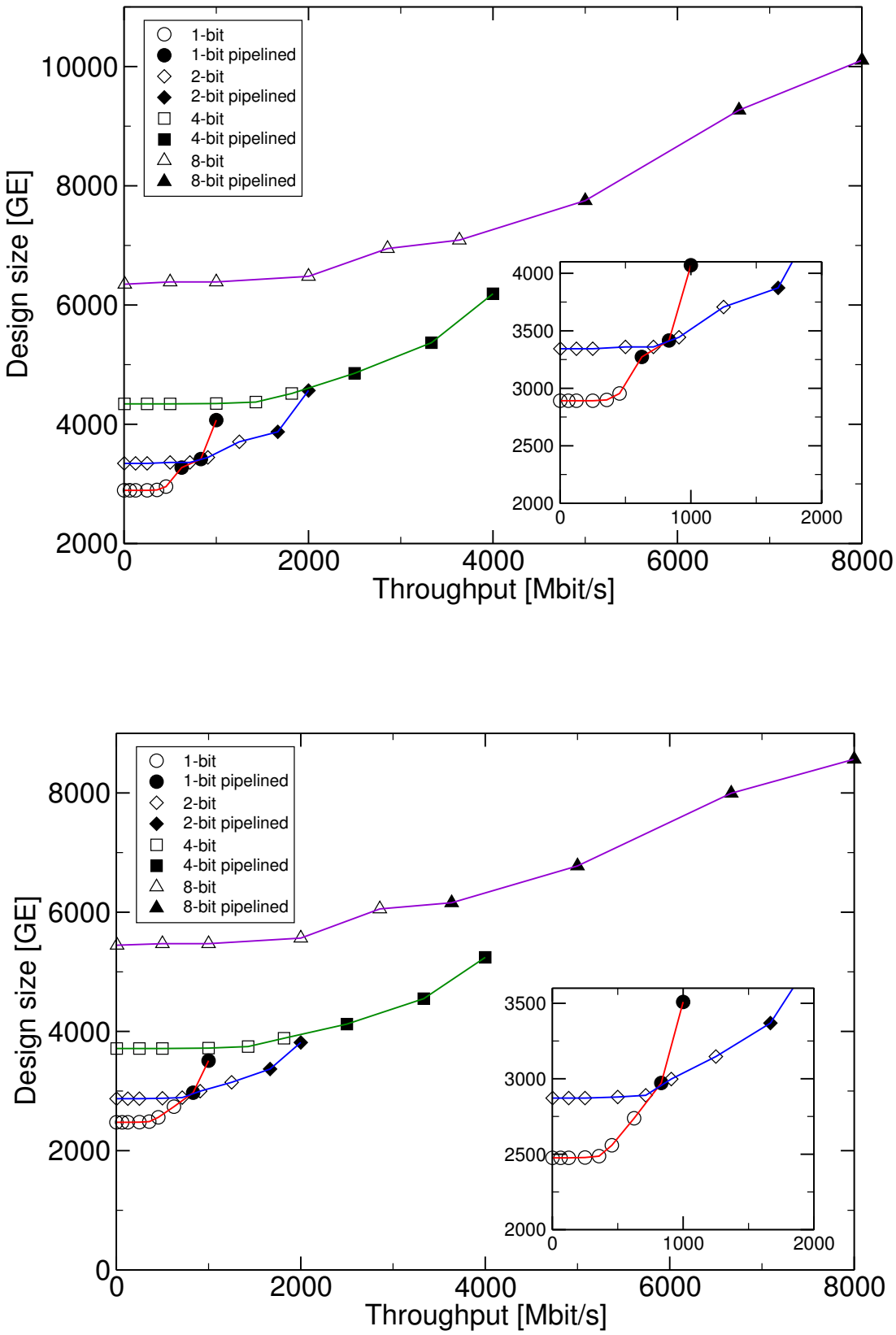


Figure 10: Throughput versus design size for ACHTERBAHN-128/80 (above) and ACHTERBAHN-80 (below) with SPA countermeasures

6.3 Resynchronization times

In many applications short resynchronization times are at least as important as high encryption speed. By *resynchronization time* we mean the number of clock cycles necessary to initialize the keystream generator with the secret key K and the public initial value IV. The resynchronization time $t_{\text{resync.1}}$ for ACHTERBAHN-128/80 with SPA countermeasures is given by the formula

$$t_{\text{resync.1}} = 1 + \frac{k + l + 80}{q},$$

where $k \in \{40, 48, \ldots, 120, 128\}$ is the key length, $l \in \{0, 8, 16, \ldots, 120, 128\}$ is the IV-length, and $q \in \{1, 2, 4, 8\}$ denotes the degree of parallelization. Table 10 contains the resynchronization times for a subset of key and IV-lengths.

Key size	IV-lengths							IV-lengths						
	0	32	64	80	96	112	128	0	32	64	80	96	112	128
	1-bit implementation							2-bit implementation						
64	145	177	209	225	241	257	273	73	89	105	113	121	129	137
80	161	193	225	241	257	273	289	81	97	113	121	129	137	145
96	177	209	241	257	273	289	305	89	105	121	129	137	145	153
112	193	225	257	273	289	305	321	97	113	129	137	145	153	161
128	209	241	273	289	305	321	337	105	121	137	145	153	161	169
	4-bit implementation							8-bit implementation						
64	37	45	53	57	61	65	69	19	23	27	29	31	33	35
80	41	49	57	61	65	69	73	21	25	29	31	33	35	37
96	45	53	61	65	69	73	77	23	27	31	33	35	37	39
112	49	57	65	69	73	77	81	25	29	33	35	37	39	41
128	53	61	69	73	77	81	85	27	31	35	37	39	41	43

Table 10: Resynchronization times for Achterbahn-128/80

The resynchronization times $t_{\text{resync.2}}$ for ACHTERBAHN-128/80 without SPA counter measures are given by the formula

$$t_{\text{resync.2}} = \frac{k + l + 96}{q},$$

where k, l, and q range over the same set of values as above.

By *pure resynchronization times* $t_{\text{resync.3}}$ we mean the time necessary to load the initial value only. The corresponding formula reads

$$t_{\text{resync.3}} = 1 + \frac{l + 96}{q}.$$

Table 11 lists the pure resynchronization times for some IV-lengths. See Section 3.5 for more explanations.

	IV-lengths						
	0	32	64	80	96	112	128
1-bit implementation	97	129	161	177	193	209	225
2-bit implementation	49	65	81	89	97	105	113
4-bit implementation	25	33	41	45	49	53	57
8-bit implementation	13	17	21	23	25	27	29

Table 11: Pure resynchronization times for Achterbahn-128/80

Acknowledgements

We would like to thank Tor Helleseth, Cees J.A. Jansen, Lothrop Mittenthal, and Harald Niederreiter for their feedback on our design proposal. We also wish to thank Yuriy Tarannikov for his important work on Boolean combining functions without which Achterbahn would look quite different.

7 Appendix A

In this appendix we collect some basic facts about the minimal polynomial of a periodic sequence with elements in a finite field $\mathbb{F}_q$. We shall refer to the results of this section at various occasions during the discussions in the main part of this report. In the main part of the report only sequences with elements in the binary field $\mathbb{F}_2$ will occur but it takes no extra effort to present the results in this section in the context of a general finite field $\mathbb{F}_q$.

Let $\mathbb{F}_q^\infty$ denote the $\mathbb{F}_q$-vector space of all sequences $\sigma = (s_n)_{n=0}^\infty$ of elements s_n of $\mathbb{F}_q$, where addition and scalar multiplication are performed termwise. A sequence $\sigma = (s_n)_{n=0}^\infty$ in $\mathbb{F}_q^\infty$ is called *periodic* if there is a positive integer p such that $s_{n+p} = s_n$ for all $n \geq 0$. The smallest such p is called the *least period* of σ. We use the notation $\mathrm{per}(\sigma)$ to designate the least period of the periodic sequence σ.

A useful linear operator on the vector space $\mathbb{F}_q^\infty$ is the *shift operator* T defined by $T\sigma = (s_{n+1})_{n=0}^\infty$ for all $\sigma \in \mathbb{F}_q^\infty$. Any polynomial $g \in \mathbb{F}_q[x]$ gives rise to a linear operator $g(T)$ on $\mathbb{F}_q^\infty$. We say that a polynomial $g \in \mathbb{F}_q[x]$ *annihilates* a periodic sequence $\sigma \in \mathbb{F}_q^\infty$ if $g(T)\sigma$ is the zero sequence $\mathbf{0}$.

Definition 5. Let σ be a periodic sequence in $\mathbb{F}_q^\infty$. Every polynomial $g \in \mathbb{F}_q[x]$ that annihilates σ is called a *characteristic polynomial* of σ.

Let $\sigma \in \mathbb{F}_q^\infty$ be periodic. The set

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

of all characteristic polynomials of σ is an ideal of the polynomial ring $\mathbb{F}_q[x]$. Since J_σ contains at least one nonzero polynomial, for instance, $g(x) = x^{\mathrm{per}(\sigma)} - 1$, and $\mathbb{F}_q[x]$ is a principal ideal domain, there is a unique monic polynomial $m_\sigma \in \mathbb{F}_q[x]$ which generates J_σ. That is,

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

This polynomial is called the *minimal polynomial* of σ. Thus, the minimal polynomial of a periodic sequence σ is the uniquely determined monic polynomial in $\mathbb{F}_q[x]$ that divides each characteristic polynomial of σ. The *linear complexity* of σ is defined as the degree of the minimal polynomial of σ. The order of the minimal polynomial of σ is equal to the least period of σ [21, Theorem 8.44].

Another interesting approach to the minimal polynomial of a periodic sequence makes use of generating functions. We assign to an arbitrary sequence $\sigma = (s_n)_{n=0}^\infty$ in $\mathbb{F}_q^\infty$ the generating function

$$G_\sigma(x) = s_0x^{-1} + s_1x^{-2} + s_2x^{-3} + \cdots,$$

which is 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.

The use of generating functions of the above form in the study of sequences with elements from an arbitrary field, instead of the frequently used formal power series $s_0 + s_1x + s_2x^2 + \cdots$, was suggested by Niederreiter in [27], [28], where the following two propositions appeared.

Proposition 1A. *Let $\sigma = (s_n)_{n=0}^{\infty}$ be a sequence of elements of $\mathbb{F}_q$, and let g be a monic polynomial over $\mathbb{F}_q$ with $g(0) \neq 0$. Then σ is a periodic sequence with characteristic polynomial g if and only if*

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

with $f \in \mathbb{F}_q[x]$ and $\deg(f) < \deg(g)$.

Proof. See Niederreiter [27], [28]. □

Proposition 2A. *Let $\sigma = (s_n)_{n=0}^{\infty}$ be a sequence of elements of $\mathbb{F}_q$, and let m be a monic polynomial over $\mathbb{F}_q$ with $m(0) \neq 0$. Then σ is a periodic sequence with minimal polynomial m if and only if*

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

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

Proof. This follows from Proposition 1A and the definition of the minimal polynomial, see [27], [28]. □

The next result is due to Laksov [20].

Propositon 3A. *Let $\sigma = (s_n)_{n=0}^{\infty}$ be a periodic sequence of elements of $\mathbb{F}_q$ with least period p. Then the minimal polynomial $m \in \mathbb{F}_q[x]$ of σ is given by*

$$m(x) = \frac{x^p - 1}{\gcd(x^p - 1, f(x))},$$

where $f(x) = s_0 x^{p-1} + s_1 x^{p-2} + \cdots + s_{p-1}$.

Proof. This follows immediately from the preceding two propositions. See also Laksov [20, Lemma 3]. □

Propositon 4A. *Let $\sigma_1, \ldots, \sigma_r$ be periodic sequences in $\mathbb{F}_q^{\infty}$ with minimal polynomials $m_i \in \mathbb{F}_q[x]$, $1 \leq i \leq r$. If the polynomials $m_1, \ldots, m_r$ are pairwise relatively prime, then the minimal polynomial of the sum $\sigma = \sigma_1 + \cdots + \sigma_r$ is equal to the product $m_1 \cdots m_r$. Conversely, let σ be a 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 $m_1, \ldots, m_r \in \mathbb{F}_q[x]$. Then, for each $i = 1, \ldots, r$, there exists a uniquely determined periodic sequence σ_i with minimal polynomial $m_i \in \mathbb{F}_q[x]$ such that $\sigma = \sigma_1 + \cdots + \sigma_r$.*

Proof. The first part of the proposition coincides with Theorem 8.57 in Lidl und Niederreiter [21, p. 426]. A proof of the second part can be found in [12, Korollar 2.5] and in [8, Lemma 6]. □

If σ is a periodic sequence in $\mathbb{F}_q^\infty$, then for any polynomial $f \in \mathbb{F}_q[x]$, the sequence $\tau = f(T)\sigma$ is also a periodic sequence in $\mathbb{F}_q^\infty$. A natural question then is to ask how the minimal polynomials of σ and τ are related to each other.

Propositon 5A. *Let σ be a periodic sequence in $\mathbb{F}_q^\infty$ with minimal polynomial $m_\sigma \in \mathbb{F}_q[x]$, and let f be a nonzero polynomial over $\mathbb{F}_q$. Then the minimal polynomial of $\tau = f(T)\sigma$ is given by*

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

Proof. There are at least three different proofs available in the literature for the assertion: Niederreiter [26, Lemma 1], Blackburn [1, Proposition 1], Gammel and Göttfert [8, Lemma 3]. □

Propositon 6A. *Let σ be a periodic sequence in $\mathbb{F}_q^\infty$ with charcteristic polynomial $g \in \mathbb{F}_q[x]$, and let f be a nonzero polynomial over $\mathbb{F}_q$. Then*

$$\frac{g}{\gcd(f, g)}$$

is a characteristic polynomial of $\tau = f(T)\sigma$.

Proof. The minimal polynomial m_σ divides the characteristic polynomial g. It follows that $m_\sigma/\gcd(f, m_\sigma)$ divides $g/\gcd(f, g)$ for every polynomial $f \in \mathbb{F}_q[x]$. For the given nonzero polynomial f, the first expression is equal to the minimal polynomial of $\tau = f(T)\sigma$ by Proposition 5A. Hence, $g/\gcd(f, g)$ is a characteristic polynomial of τ. □

We conclude this section with the proof of Lemma 7.

Proof of Lemma 7. 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 on page 27. By Lemma 1, the irreducible polynomials $f_1, \ldots, f_s \in \mathbb{F}_2[x]$ are distinct and $\deg(f_i)$ divides S for $1 \le i \le s$. Similarly, the irreducible polynomials $g_1, \ldots, g_t$ are distinct and $\deg(g_j)$ divides T for $1 \le j \le t$. 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 Proposition 4A, 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 $1 \le i \le s$, and τ_j is a binary periodic sequence with minimal polynomial g_j for $1 \le j \le t$. 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 Lemma 3 and Lemma 5 of Section 4.1, 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 $f_i \vee g_j$, $1 \le i \le s$, $1 \le j \le t$, are distinct. Another application of Proposition 4A therefore shows that the minimal polynomial of $\sigma\tau$ has the form

$$m_{\sigma\tau} = \prod_{i=1}^{s}\prod_{j=1}^{t}(f_i \vee g_j). \tag{32}$$

It remains to show that the polynomials $f_i \vee g_j$, $1 \le i \le s$, $1 \le j \le t$, 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 monic 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

$$\gamma = \alpha\beta = \alpha'\beta', \tag{33}$$

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 (33) we obtain

$$\frac{\alpha}{\alpha'} = \frac{\beta'}{\beta}. \tag{34}$$

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 (34) must be equal to 1. Hence $\alpha = \alpha'$ and $\beta = \beta'$. This, however, implies $f_i = f_i'$ and $g_j = g_j'$. □

8 Appendix B

Let A be a binary primitive N-stage FSR, and let f be an irreducible polynomial over $\mathbb{F}_2$. We want to know whether or not f is a divisor of the minimal polynomial of A (compare Definition 2). We can restrict ourselves to irreducible polynomials whose degrees divide N and are greater than 1. Any irreducible polynomial f which does not meet these requirements is definitely not a factor of the minimal polynomial of A by Lemma 1.

Proposition 1B. *Let A be a binary primitive N-stage FSR, and let f be a binary polynomial whose degree d divides N and is greater than 1. Then f divides the minimal polynomial of A if and only if the first d terms of the sequence $\tau = h(T)\sigma$ are not all zero. Here $h(x) = (x^p - 1)/f(x)$, $p = 2^N - 1$, and σ is an arbitrary nonzero output sequence of A.*

Proof. Since $x^p - 1$ is a characteristic polynomial of σ, Proposition 6A implies that

$$f(x) = \frac{x^p - 1}{\gcd(x^p - 1, h(x))}$$

is a characteristic polynomial of τ and, therefore, a multiple of the minimal polynomial of τ. Since f is irreducible, we have either $m_\tau = f$ or $m_\tau = 1$. The latter, of course, means that τ is the zero sequence. Since τ has characteristic polynomial f and $\deg(f) = d$, $\tau = (t_n)_{n=0}^\infty$ is the zero sequence exactly if $(t_0, t_1, \ldots, t_{d-1})$ is the zero vector of $\mathbb{F}_2^d$. By Proposition 5A,

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

Thus, $m_\tau = 1$ if and only if $\gcd(m_\sigma, h) = m_\sigma$. The latter is true if and only if m_σ divides h. The minimal polynomial m_σ divides the polynomial $h(x) = (x^p - 1)/f(x)$ if and only if f is not a divisor of m_σ. □

Let $\sigma = (s_n)_{n=0}^\infty$ be a nonzero output sequence of the binary primitive N-stage FSR A, and let $\tau = (t_n)_{n=0}^\infty$, d, p, $f(x)$, and $h(x)$ be as in Proposition 1B. For $n \geq 0$ define the row vector $\mathbf{s}_n^{(d)} = (s_n, s_{n+1}, \ldots, s_{n+d-1})$. Furthermore, define $\mathbf{v} = (t_0, t_1, \ldots, t_{d-1})$ and write $h(x) = \sum_{i=0}^{p-d} c_i x^i$. By what we have just proved, f divides m_σ if and only if

$$\mathbf{v} = \sum_{i=0}^{p-d} c_i \mathbf{s}_i^{(d)} \neq \mathbf{0}. \tag{35}$$

If we want to compute the vector $\mathbf{v} \in \mathbb{F}_2^d$ by this formula, we encounter the following problem: The natural order in which the row vectors $\mathbf{s}_i^{(d)}$ are produced is $\mathbf{s}_0^{(d)}, \mathbf{s}_1^{(d)}, \ldots, \mathbf{s}_{p-d}^{(d)}$, whereas the order in which the coefficients c_i of the polynomial $h(x)$ are produced—using the division algorithm—is $c_{p-d}, c_{p-d-1}, \ldots, c_0$. This means that in order to compute the row vector $\mathbf{v}$ in (35), we need to store either a full portion of the period of σ or all positions i, $0 \leq i \leq p - d$, for which $c_i = 1$. For shift register lengths $N \geq 40$, this becomes a hardship.

There are two ways to overcome this problem: We can use a modified feedback function that produces the row vectors $\mathbf{s}_i^{(d)}$ in reverse order (see Walker [34, p. 373]).

Or we can use the reciprocal polynomial of f in the division algorithm. The reciprocal polynomial f^* of the polynomial $f(x) = x^d + a_{d-1}x^{d-1} + \cdots + a_1x + a_0$ is defined to be the polynomial

$$f^*(x) = x^{\deg(f)} f\Big(\frac{1}{x}\Big) = a_0x^d + a_1x^{d-1} + \cdots + a_{d-1}x + 1.$$

Replace in

$$\frac{x^p - 1}{f(x)} = h(x) = \sum_{i=0}^{p-d} c_i x^i$$

the variable x by $1/x$ and multiply the result by x^{p-d}. This yields

$$\frac{x^p - 1}{f^*(x)} = x^{p-d} h\Big(\frac{1}{x}\Big) = \sum_{i=0}^{p-d} c_i x^{p-d-i}.$$

In other words, the division algorithm performed for the polynomials $x^p - 1$ and $f^*(x)$ produces the coefficients c_i, that are needed in (35), in the desired order. This drastically reduces the storage requirements in the computation of $\mathbf{v}$. Putting everything together, we arrive at Algorithm A in Section 2.3.

References

[1] 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).

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

[3] N. G. de Bruijn: Acknowledgement of priority to C. Flye Sainte-Marie on the counting of 2^n zeros and ones that show each n-letter word exactly once, Technical Report TH 75-WSK-06, Technische Hogeschool Eindhoven, 1975.

[4] C. Flye Sainte-Marie: *L' Interméd. Math.* **1**, 107–110 (1894).

[5] 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, Berlin, 2003.

[6] T. W. Cusick: On constructing balanced correlation immune functions, *Sequences and their Applications: Proceedings of SETA '98* (C. Ding, T. Helleseth, and H. Niederreiter, eds.), pp. 184–190, Springer-Verlag, London, 1999.

[7] 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, Berlin, 1991.

[8] B. M. Gammel and R. Göttfert: Linear filtering of nonlinear shift register sequences, *Proc. of The Intern. Workshop on Coding and Cryptography — WCC 2005* (Ø. Ytrehus, ed.), Lecture Notes in Computer Science, vol. 3969, pp. 354-370, Springer-Verlag, Berlin, 2006.

[9] B. M. Gammel, R. Göttfert, and O. Kniffler: The Achterbahn stream cipher, eSTREAM, ECRYPT Stream Cipher Project, Report 2005/002, 29 April 2005. http://www.ecrypt.eu.org/stream/papers.html

[10] B. M. Gammel, R. Göttfert, and O. Kniffler: Improved Boolean combining functions for Achterbahn, eSTREAM, ECRYPT Stream Cipher Project, Report 2005/072, 14 October 2005. http://www.ecrypt.eu.org/stream/papers.html

[11] B. M. Gammel, R. Göttfert, and O. Kniffler: Status of Achterbahn and tweaks, *SASC 2006—Stream Ciphers Revisited* (Leuven, Belgium, 2-3 February 2006), Workshop Record, pp. 302–315.

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

[13] F. K. Gürkaynak, P. Luethi, N. Bernold, R. Blattmann, V. Goode, M. Marghitola, H. Kaeslin, N. Felber, and W. Fichtner: Hardware evaluation of eSTREAM candidates: Achterbahn, Grain, MICKEY, MOSQUITO, SFINKS,

Trivium, VEST, ZK-Crypt, *SASC 2006—Stream Ciphers Revisited* (Leuven, Belgium, 2-3 February 2006), Workshop Record, pp. 113–124.

[14] J. Dj. Golić: On the linear complexity of functions of periodic GF(q) sequences, *IEEE Trans. Inform. Theory* **35**, 69–75 (1989).

[15] M. Hell and T. Johansson: Cryptanalysis of Achterbahn-Version 2, http://www.ecrypt.eu.org/stream/papers.html

[16] T. Herlestam: On functions of linear shift register sequences, *Advances in Cryptology — EUROCRYPT '85* (F. Pichler, ed.), Lecture Notes in Computer Science, vol. 219, pp. 119–129, Springer-Verlag, Berlin, 1986.

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

[18] T. Johansson, W. Meier, and F. Muller: Cryptanalysis of Achterbahn, eSTREAM, ECRYPT Stream Cipher Project, Report 2005/064, 27 September 2005. http://www.ecrypt.eu.org/stream/papers.html

[19] T. Johansson, W. Meier, and F. Muller: Cryptanalysis of Achterbahn, *Proceedings of the 13th Fast Software Encryption Workshop – FSE 2006* (Graz, Austria, March 15–17, 2006), pp. 1–14, 2006.

[20] D. Laksov: Linear recurring sequences over finite fields, *Math. Scand.* **16**, 181–196 (1965).

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

[22] S. Maitra and P. Sarkar: Highly nonlinear resilient functions optimizing Siegenthaler's inequality, *Advances in Cryptology — CRYPTO '99* (M. Wiener, ed.), Lecture Notes in Computer Science, vol. 1666, pp. 198–215, Springer-Verlag, Berlin, 1999.

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

[24] W. Meier, E. Pasalic, and C. Carlet: Algebraic attacks and decomposition of Boolean functions, *Advances in Cryptology — EUROCRYPT 2004* (C. Cachin and J. Camenisch, eds.), Lecture Notes in Computer Science, vol. 3027, pp. 474–491, Springer-Verlag, Berlin, 2004.

[25] W. Meier and O. Staffelbach: Nonlinearity criteria for cryptographic functions, *Advances in Cryptology — EUROCRYPT '89* (J.-J. Quisquater and J. Vandewalle, eds.), Lecture Notes in Computer Science, vol. 434, pp. 549–562, Springer-Verlag, Berlin, 1990.

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

[27] H. Niederreiter: Sequences with almost perfect linear complexity profile, *Advances in Cryptology — EUROCRYPT '87* (D. Chaum and W. L. Price, eds.), Lecture Notes in Computer Science, vol. 304, pp. 37–51, Springer-Verlag, Berlin, 1988.

[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] R. A. Rueppel and O. J. Staffelbach: Products of linear recurring sequences with maximum complexity, *IEEE Trans. Inform. Theory* **IT-33**, 124–131 (1987).

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

[31] A. Shamir, J. Patarin, N. Courtois, and A. Klimov: Efficient algorithms for solving overdefined systems of multivariate polynomial equations, *Advances in Cryptology — EUROCRYPT 2000* (B. Preneel, ed.), Lecture Notes in Computer Science, vol. 1807, pp. 392–407, Springer-Verlag, Berlin, 2000.

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

[33] Y. Tarannikov: On resilient Boolean functions with maximum possible nonlinearity, *Advances in Cryptology — INDOCRYPT 2000*, Lecture Notes in Computer Science, vol. 1977, pp. 19–30, Springer-Verlag, Berlin, 2000.

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

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