TRIVIUM
A Stream Cipher Construction Inspired by Block Cipher Design Principles*

Christophe De Cannière[1,2]

[1] IAIK Krypto Group, Graz University of Technology
Inffeldgasse 16A, A–8010 Graz, Austria
christophe.decanniere@iaik.tugraz.at
[2] Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC,
Kasteelpark Arenberg 10, B–3001 Heverlee, Belgium

Abstract. In this paper, we propose a new stream cipher construction based on block cipher design principles. The main idea is to replace the building blocks used in block ciphers by equivalent stream cipher components. In order to illustrate this approach, we construct a very simple synchronous stream cipher which provides a lot of flexibility for hardware implementations, and seems to have a number of desirable cryptographic properties.

1 Introduction

In the last few years, widely used stream ciphers have started to be systematically replaced by block ciphers. An example is the A5/1 stream cipher used in the GSM standard. Its successor, A5/3, is a block cipher. A similar shift took place with wireless network standards. The security mechanism specified in the original IEEE 802.11 standard (called 'wired equivalent privacy' or WEP) was based on the stream cipher RC4; the newest standard, IEEE 802.11i, makes use of the block cipher AES.

The declining popularity of stream ciphers can be explained by different factors. The first is the fact that the security of block ciphers seems to be better understood. Over the last decades, cryptographers have developed a rather clear vision of what the internal structure of a secure block cipher should look like. This is much less the case for stream ciphers. Stream ciphers proposed in the past have been based on very different principles, and many of them have shown weaknesses. A second explanation is that efficiency, which has been the traditional motivation for choosing a stream cipher over a block cipher, has ceased to be a decisive factor in many applications: not only is the cost of computing power rapidly decreasing, today's block ciphers are also significantly more efficient than their predecessors.

* The work described in this paper has been partly supported by the European Commission under contract IST-2002-507932 (ECRYPT), by the Fund for Scientific Research – Flanders (FWO), and by the Austrian Science Fund (FWF project P18138).

Still, as pointed out by the eSTREAM Stream Cipher Project, it seems that stream ciphers could continue to play an important role in those applications where high througput remains critical and/or where resources are very restricted. This poses two challenges for the cryptographic community: first, restoring the confidence in stream ciphers, e.g., by developing simple and reliable design criteria; secondly, increasing the efficiency advantage of stream ciphers compared to block ciphers.

In this paper, we try to explore both problems. The first part of the article reviews some concepts which lie at the base of today's block ciphers (Sect. 3), and studies how these could be mapped to stream ciphers (Sects. 4–5). The design criteria derived this way are then used as a guideline to construct a simple and flexible hardware-oriented stream cipher in the second part (Sect. 6).

2 Security and Efficiency Considerations

Before devising a design strategy for a stream cipher, it is useful to first clearly specify what we expect from it. Our aim in this paper is to design hardware-oriented binary additive stream ciphers which are both efficient and secure. The following sections briefly discuss what this implies.

2.1 Security

The additive stream cipher which we intend to construct takes as input a k-bit secret key K and an n-bit IV. The cipher is then requested to generate up to 2^d bits of key stream $z_t = S_K(IV, t)$, $0 \leq t < 2^d$, and a bitwise exclusive OR of this key stream with the plaintext produces the ciphertext. The security of this additive stream cipher is determined by the extent to which it mimics a one-time pad, i.e., it should be hard for an adversary, who does not know the key, to distinguish the key stream generated by the cipher from a truly random sequence. In fact, we would like this to be as hard as we can possibly ask from a cipher with given parameters k, n, and d. This leads to a criterion called K-security [1], which can be formulated as follows:

Definition 1. *An additive stream cipher is called K-secure if any attack against this scheme would not have been significantly more difficult if the cipher had been replaced by a set of 2^k functions $S_K\colon \{0,1\}^n \times \{0,\ldots,2^d-1\} \to \{0,1\}$, uniformly selected from the set of all possible functions.*

The definition assumes that the adversary has access to arbitrary amounts of key stream, that he knows or can choose the a priory distribution of the secret key, that he can impose relations between different secret keys, etc.

Attacks against stream ciphers can be classified into two categories, depending on what they intend to achieve:

- *Key recovery attacks*, which try to deduce information about the secret key by observing the key stream.

– *Distinguishing attacks*, the goal of which is merely to detect that the key stream bits are not completely unpredictable.

Owing to their weaker objective, distinguishing attacks are often much easier to apply, and consequently harder to protect against. Features of the key stream that can be exploited by such attacks include periodicity, dependencies between bits at different positions, non-uniformity of distributions of bits or words, etc. In this paper we will focus in particular on linear correlations, as it appeared to be the weakest aspect in a number of recent stream cipher proposals such as SOBER-t*w* [2] and SNOW 1.0 [3]. Our first design objective will be to keep the largest correlations below safe bounds. Other important properties, such as a sufficiently long period, are only considered afterwards. Note that this approach differs from the way LFSR or T-function based schemes are constructed. The latter are typically designed by maximizing the period first, and only then imposing additional requirements.

2.2 Efficiency

In order for a stream cipher to be an attractive alternative to block ciphers, it must be efficient. In this paper, we will be targeting hardware applications, and a good measure for the efficiency of a stream cipher in this environment is the number of key stream bits generated per cycle per gate.

There are two ways to obtain an efficient scheme according to this measure. The first approach is illustrated by A5/1, and consists in minimizing the number of gates. A5/1 is extremely compact in hardware, but it cannot generate more than one bit per cycle. The other approach, which was chosen by the designers of PANAMA [4], is to dramatically increase the number of bits per cycle. This allows to reduce the clock frequency (and potentially also the power consumption) at the cost of an increased gate count. As a result, PANAMA is not suited for environments with very tight area constraints. Similarly, designs such as A5/1 will not perform very well in systems which require fast encryption at a low clock frequency. One of the objectives of this paper is to design a flexible scheme which performs reasonably well in both situations.

3 How Block Ciphers are Designed

As explained above, the first requirement we impose on the construction is that it generates key streams without exploitable linear correlations. This problem is very similar to the one faced by block cipher designers. Hence, it is natural to attempt to borrow some of the techniques used in the block cipher world. The ideas relevant to stream ciphers are briefly reviewed in the following sections.

3.1 Block Ciphers and Linear Characteristics

An important problem in the case of block ciphers is that of restricting linear correlations between input and output bits in order to thwart linear cryptanalysis [5]. More precisely, let P be any plaintext block and C the corresponding

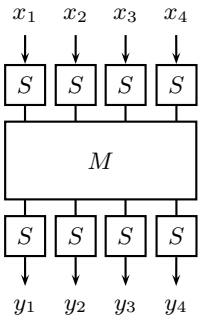

Fig. 1. Three layers of a block cipher

ciphertext under a fixed secret key, then any linear combination of bits

$$\Gamma_P^{\mathsf{T}} \cdot P + \Gamma_C^{\mathsf{T}} \cdot C \,,$$

where the column vectors Γ_P and Γ_C are called *linear masks*, should be as balanced as possible. That is, the correlation

$$c = 2 \cdot \frac{|\{P \mid \Gamma_P^{\mathsf{T}} \cdot P = \Gamma_C^{\mathsf{T}} \cdot C\}|}{|\{P\}|} - 1$$

has to be close to 0 for any Γ_P and Γ_C. The well-established way to achieve this consists in alternating two operations. The first splits blocks into smaller words which are independently fed into nonlinear substitution boxes (S-boxes); the second step recombines the outputs of the S-boxes in a linear way in order to 'diffuse' the nonlinearity. The result, called a substitution-permutation network, is depicted in Fig. 1.

In order to estimate the strength of a block cipher against linear cryptanalysis, one will typically compute bounds on the correlation of *linear characteristics*. A linear characteristic describes a possible path over which a correlation might propagate through the block cipher. It is a chain of linear masks, starting with a plaintext mask and ending with a ciphertext mask, such that every two successive masks correspond to a nonzero correlation between consecutive intermediate values in the cipher. The total correlation of the characteristic is then estimated by multiplying the correlations of all separate steps (as dictated by the so-called Piling-up Lemma).

3.2 Branch Number

Linear diffusion layers, which can be represented by a matrix multiplication $Y = M \cdot X$, do not by themselves contribute in reducing the correlation of a characteristic. Clearly, it suffices to choose $\Gamma_X = M^{\mathsf{T}} \cdot \Gamma_Y$, where M^{T} denotes the transpose of M, in order to obtain perfectly correlating linear combinations of X and Y:

$$\Gamma_Y^{\mathsf{T}} \cdot Y = \Gamma_Y^{\mathsf{T}} \cdot MX = (M^{\mathsf{T}}\Gamma_Y)^{\mathsf{T}} \cdot X = \Gamma_X^{\mathsf{T}} \cdot X \,.$$

However, diffusion layers play an important indirect role by forcing characteristics to take into account a large number of nonlinear S-boxes in the neighboring layers (called *active* S-boxes). A useful metric in this context is the *branch number* of M.

Definition 2. *The branch number of a linear transformation M is defined as*

$$\mathcal{B} = \min_{\Gamma_Y \neq 0} [\mathrm{w_h}(\Gamma_Y) + \mathrm{w_h}(M^{\mathsf{T}}\Gamma_Y)] ,$$

where $\mathrm{w_h}(\Gamma)$ represents the number of nonzero words in the linear mask Γ.

The definition above implies that any linear characteristic traversing the structure shown in Fig. 1 activates at least $\mathcal{B}$ S-boxes. The total number of active S-boxes throughout the cipher multiplied by the maximal correlation over a single S-box gives an upper bound for the correlation of the characteristic.

The straightforward way to minimize this upper bound is to maximize the branch number $\mathcal{B}$. It is easy to see that $\mathcal{B}$ cannot exceed $m + 1$, with m the number of words per block. Matrices M that satisfy this bound (known as the Singleton bound) can be derived from the generator matrices of maximum distance separable (MDS) block codes.

Large MDS matrices are expensive to implement, though. Therefore, it is often more efficient to use smaller matrices, with a relatively low branch number, and to connect them in such a way that linear patterns with a small number of active S-boxes cannot be chained together to cover the complete cipher. This was the approach taken by the designers of Rijndael [6].

4 From Blocks to Streams

In this section, we try to adapt the concepts described above to a system where the data is not processed in blocks, but rather as a stream.

Since data enters the system one word at a time, each layer of S-boxes in Fig. 1 can be replaced by a single S-box which substitutes individual words as they arrive. A general mth-order linear filter can take over the task of the diffusion matrix. The new system is represented in Fig. 2, where D denotes the delay operator (usually written as z^{-1} in signal processing literature), and f and g are linear functions.

4.1 Polynomial Notation

Before analyzing the properties of this construction, we introduce some notations. First, we adopt the common convention to represent streams of words $x_0, x_1, x_2, \ldots$ as polynomials with coefficients in the finite field:

$$x(D) = x_0 + x_1 D + x_2 D^2 + \ldots .$$

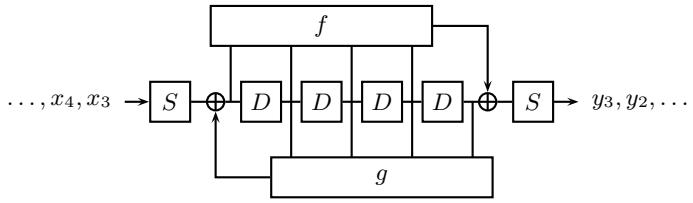

Fig. 2. Stream equivalent of Fig. 1

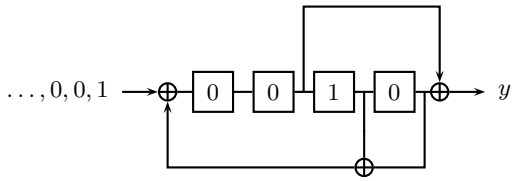

Fig. 3. A 4th-order linear filter

The rationale for this representation is that it simplifies the expression for the input/output relation of the linear filter, as shown in the following equation:

$$y(D) = \frac{f(D)}{g(D)} \cdot \big[x(D) + x^0(D)\big] + y^0(D)\,. \tag{1}$$

The polynomials f and g describe the feedforward and feedback connections of the filter. They can be written as

$$\begin{aligned} f(D) &= D^m \cdot \left(f_m D^{-m} + \cdots + f_1 D^{-1} + 1\right)\,, \\ g(D) &= 1 + g_1 D + g_2 D^2 + \cdots + g_m D^m\,. \end{aligned}$$

The Laurent polynomials x^0 and y^0 represent the influence of the initial state s^0, and are given by $x^0 = D^{-m} \cdot \big(s^0 \cdot g \bmod D^m\big)$ and $y^0 = D^{-m} \cdot \big(s^0 \cdot f \bmod D^m\big)$.

Example 1. The 4th-order linear filter depicted in Fig. 3 is specified by the polynomials $f(D) = D^4 \cdot (D^{-2} + 1)$ and $g(D) = 1 + D^3 + D^4$. Suppose that the delay elements are initialized as shown in the figure, i.e., $s^0(D) = D$. Knowing s^0, we can compute $x^0(D) = D^{-3}$ and $y^0(D) = D^{-1}$. Finally, using (1), we find the output stream corresponding to an input consisting, for example, of a single 1 followed by 0's (i.e., $x(D) = 1$):

$$\begin{aligned} y(D) &= \frac{D^{-1} + D + D^2 + D^4}{1 + D^3 + D^4} + D^{-1} \\ &= D + D^3 + D^5 + D^6 + D^7 + D^8 + D^{12} + D^{15} + D^{16} + D^{18} + \ldots \end{aligned}$$

4.2 Linear Correlations

In order to study correlations in a stream-oriented system we need a suitable way to manipulate linear combinations of bits in a stream. It will prove convenient to represent them as follows:

$$\mathrm{Tr}\left[[\gamma_x(D^{-1}) \cdot x(D)]_0\right] .$$

The operator $[\cdot]_0$ returns the constant term of a polynomial, and $\mathrm{Tr}(\cdot)$ denotes the trace to GF(2).[3] The coefficients of γ_x, called *selection polynomial*, specify which words of x are involved in the linear combination. In order to simplify expressions later on we also introduce the notation $\gamma^*(D) = \gamma(D^{-1})$. The polynomial γ^* is called the *reciprocal* polynomial of γ.

As before, the correlation between x and y for a given pair of selection polynomials is defined as

$$c = 2 \cdot \frac{|\{(x, s^0) \mid \mathrm{Tr}[[\gamma_x^* \cdot x]_0] = \mathrm{Tr}[[\gamma_y^* \cdot y]_0]\}|}{|\{(x, s^0)\}|} - 1 ,$$

where $\deg x \leq \max(\deg \gamma_x, \deg \gamma_y)$.

4.3 Propagation of Selection Polynomials

Let us now analyze how correlations propagate through the linear filter. For each selection polynomial γ_x at the input, we would like to determine a polynomial γ_y at the output (if it exists) such that the corresponding linear combinations are perfectly correlated, i.e.,

$$\mathrm{Tr}[[\gamma_x^* \cdot x]_0] = \mathrm{Tr}[[\gamma_y^* \cdot y]_0], \quad \forall x, s^0 .$$

If this equation is satisfied, then this will still be the case after replacing x by $x' = x + x^0$ and y by $y' = y + y^0$, since x^0 and y^0 only consist of negative powers, none of which can be selected by γ_x or γ_y. Substituting (1), we find

$$\mathrm{Tr}[[\gamma_x^* \cdot x']_0] = \mathrm{Tr}[[\gamma_y^* \cdot f/g \cdot x']_0], \quad \forall x, s^0 ,$$

which implies that $\gamma_x^* = \gamma_y^* \cdot f/g$. In order to get rid of negative powers, we define $f^\star = D^m \cdot f^*$ and $g^\star = D^m \cdot g^*$ (note the subtle difference between both stars), and obtain the equivalent relation

$$\gamma_y = g^\star / f^\star \cdot \gamma_x . \tag{2}$$

Note that neither of the selection polynomials γ_x and γ_y can have an infinite number of nonzero coefficients (if it were the case, the linear combinations would be undefined). Hence, they have to be of the form

$$\gamma_x = q \cdot f^\star / \gcd(f^\star, g^\star) \quad \text{and} \quad \gamma_y = q \cdot g^\star / \gcd(f^\star, g^\star) , \tag{3}$$

with $q(D)$ an arbitrary polynomial.

[3] The trace from $GF(2^n)$ to $GF(2)$ is defined as $\mathrm{Tr}(a) = a + a^2 + a^4 + \cdots + a^{2^{n-1}}$.

Example 2. For the linear filter in Fig. 3, we have that $f^\star(D) = 1 + D^2$ and $g^\star(D) = D^4 \cdot (D^{-4} + D^{-3} + 1)$. In this case, $f^\star$ and $g^\star$ are coprime, i.e., $\gcd(f^\star, g^\star) = 1$. If we arbitrarily choose $q(D) = 1 + D$, we obtain a pair of selection polynomials

$$\gamma_x(D) = 1 + D + D^2 + D^3 \quad \text{and} \quad \gamma_y(D) = 1 + D^2 + D^4 + D^5 \,.$$

By construction, the corresponding linear combinations of input and output bits satisfy the relation

$$\mathrm{Tr}(x_0 + x_1 + x_2 + x_3) = \mathrm{Tr}(y_0 + y_2 + y_4 + y_5), \quad \forall x, s^0 \,.$$

4.4 Branch Number

The purpose of the linear filter, just as the diffusion layer of a block cipher, will be to force linear characteristics to pass through as many active S-boxes as possible. Hence, it makes sense to define a branch number here as well.

Definition 3. *The branch number of a linear filter specified by the polynomials f and g is defined as*

$$\begin{aligned}\mathcal{B} &= \min_{\gamma_x \neq 0} [\mathrm{w_h}(\gamma_x) + \mathrm{w_h}(g^\star / f^\star \cdot \gamma_x)] \\ &= \min_{q \neq 0} [\mathrm{w_h}(q \cdot f^\star / \gcd(f^\star, g^\star)) + \mathrm{w_h}(q \cdot g^\star / \gcd(f^\star, g^\star))] \,,\end{aligned}$$

where $\mathrm{w_h}(\gamma)$ represents the number of nonzero coefficients in the selection polynomial γ.

From this definition we immediately obtain the following upper bound on the branch number

$$\mathcal{B} \leq \mathrm{w_h}(f^\star) + \mathrm{w_h}(g^\star) \leq 2 \cdot (m + 1) \,. \tag{4}$$

Filters for which this bound is attained can be derived from MDS convolutional $(2, 1, m)$-codes [7]. For example, one can verify that the 4th-order linear filter over $\mathrm{GF}(2^8)$ with

$$\begin{aligned}f(D) &= D^4 \cdot \left(\mathtt{02}_x D^{-4} + D^{-3} + D^{-2} + \mathtt{02}_x D^{-1} + 1\right) \,, \\ g(D) &= 1 + \mathtt{03}_x D + \mathtt{03}_x D^2 + D^3 + D^4 \,,\end{aligned}$$

has a branch number of 10. Note that this example uses the same field polynomial as Rijndael, i.e., $x^8 + x^4 + x^3 + x + 1$.

5 Constructing a Key Stream Generator

In the previous section, we introduced S-boxes and linear filters as building blocks, and presented some tools to analyze how they interact. Our next task is to determine how these components can be combined into a key stream generator. Again, block ciphers will serve as a source of inspiration.

5.1 Basic Construction

A well-known way to construct a key stream generator from a block cipher is to use the cipher in output feedback (OFB) mode. This mode of operation takes as input an initial data block (called initial value or IV), passes it through the block cipher, and feeds the result back to the input. This process is iterated and the consecutive values of the data block are used as key stream. We recall that the block cipher itself typically consists of a sequence of rounds, each comprising a layer of S-boxes and a linear diffusion transformation.

By taking the very same approach, but this time using the stream cipher components presented in Sect. 4, we obtain a construction which, in its simplest form, might look like Fig. 4(a). The figure represents a key stream generator

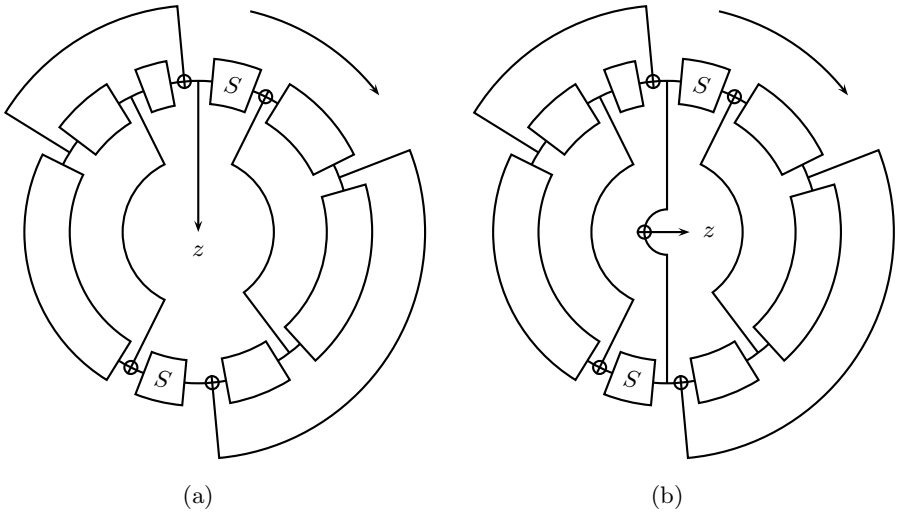

Fig. 4. Two-round key stream generators

consisting of two 'rounds', where each round consists of an S-box followed by a very simple linear filter. Data words traverse the structure in clockwise direction, and the output of the second round, which also serves as key stream, is fed back to the input of the first round.

While the scheme proposed above has some interesting structural similarities with a block cipher in OFB mode, there are important differences as well. The most fundamental difference comes from the fact that linear filters, as opposed to diffusion matrices, have an internal state. Hence if the algorithm manages to keep this state (or at least parts of it) secret, then this eliminates the need for a separate key addition layer (another important block cipher component, which we have tacitly ignored so far).

5.2 Analysis of Linear Characteristics

As stated before, the primary goal in this paper is to construct a scheme which generates a stream of seemingly uncorrelated bits. More specifically, we would like the adversary to be unable to detect any correlation between linear combinations of bits at different positions in the key stream. In the following sections, we will see that the study of linear characteristics provides some guidance on how to design the components of our scheme in order to reduce the magnitude of these correlations.

Applying the tools from Sect. 4 to the construction in Fig. 4(a), we can easily derive some results on the existence of low-weight linear characteristics. The term 'low-weight' in this context refers to a small number of active S-boxes. Since we are interested in correlations which can be detected by an adversary, we need both ends of the characteristic to be accessible from the key stream. In order to construct such characteristics, we start with a selection polynomial γ_u at the input of the first round, and analyze how it might propagate through the cipher.

First, the characteristic needs to cross an S-box. The S-box preserves the positions of the non-zero coefficients of γ_u, but might modify their values. For now, however, let us only consider characteristics for which the values are preserved as well. Under this assumption and using (2), we can compute the selection polynomials γ_v and γ_w at the input and the output of the second round:

$$\gamma_v = g_1^\star / f_1^\star \cdot \gamma_u \quad \text{and} \quad \gamma_w = g_2^\star / f_2^\star \cdot \gamma_v \,.$$

Since all three polynomials γ_u, γ_v, and γ_w need to be finite, we have that

$$\gamma_u = q \cdot f_1^\star f_2^\star / d \,, \quad \gamma_v = q \cdot g_1^\star f_2^\star / d \,, \quad \text{and} \quad \gamma_w = q \cdot g_1^\star g_2^\star / d \,,$$

with $d = \gcd(f_1^\star f_2^\star, g_1^\star f_2^\star, g_1^\star g_2^\star)$ and q an arbitrary polynomial. Note that since both γ_u and γ_w select bits from the key stream z, they can be combined into a single polynomial $\gamma_z = \gamma_u + \gamma_w$.

The number of S-boxes activated by a characteristic of this form is given by $\mathcal{W} = \mathrm{w_h}(\gamma_u) + \mathrm{w_h}(\gamma_v)$. The minimum number of active S-boxes over this set of characteristics can be computed with the formula

$$\mathcal{W}_{\min} = \min_{q \neq 0} [\mathrm{w_h}(q \cdot f_1^\star f_2^\star / d) + \mathrm{w_h}(q \cdot g_1^\star f_2^\star / d)] \,,$$

from which we derive that

$$\mathcal{W}_{\min} \leq \mathrm{w_h}(f_1^\star f_2^\star) + \mathrm{w_h}(g_1^\star f_2^\star) \leq \mathrm{w_h}(f_1^\star) \cdot \mathrm{w_h}(f_2^\star) + \mathrm{w_h}(g_1^\star) \cdot \mathrm{w_h}(f_2^\star) \,.$$

Applying this bound to the specific example of Fig. 4(a), where $\mathrm{w_h}(f_i^\star) = \mathrm{w_h}(g_i^\star) = 2$, we conclude that there will always exist characteristics with at most 8 active S-boxes, no matter where the taps of the linear filters are positioned.

5.3 An Improvement

We will now show that this bound can potentially be doubled by making the small modification shown in Fig. 4(b). This time, each non-zero coefficient in the selection polynomial at the output of the key stream generator needs to propagate to both the upper and the lower part of the scheme. By constructing linear characteristics in the same way as before, we obtain the following selection polynomials:

$$\gamma_u = q \cdot \frac{f_1^\star f_2^\star + f_1^\star g_2^\star}{d}\,, \quad \gamma_v = q \cdot \frac{f_1^\star f_2^\star + g_1^\star f_2^\star}{d}\,, \quad \text{and} \quad \gamma_z = q \cdot \frac{f_1^\star f_2^\star + g_1^\star g_2^\star}{d}\,,$$

with $d = \gcd(f_1^\star f_2^\star + f_1^\star g_2^\star, f_1^\star f_2^\star + g_1^\star f_2^\star, f_1^\star f_2^\star + g_1^\star g_2^\star)$. The new upper bounds on the minimum number of active S-boxes are given by

$$\begin{aligned}\mathcal{W}_{\min} &\le \mathrm{w_h}(f_1^\star f_2^\star + f_1^\star g_2^\star) + \mathrm{w_h}(f_1^\star f_2^\star + g_1^\star f_2^\star) \\ &\le 2 \cdot \mathrm{w_h}(f_1^\star) \cdot \mathrm{w_h}(f_2^\star) + \mathrm{w_h}(f_1^\star) \cdot \mathrm{w_h}(g_2^\star) + \mathrm{w_h}(g_1^\star) \cdot \mathrm{w_h}(f_2^\star)\,,\end{aligned}$$

or, in the case of Fig. 4(b), $\mathcal{W}_{\min} \le 16$. In general, if we consider extensions of this scheme with r rounds and $\mathrm{w_h}(f_i^\star) = \mathrm{w_h}(g_i^\star) = w$, then the bound takes the form:

$$\mathcal{W}_{\min} \le r^2 \cdot w^r\,. \tag{5}$$

This result suggests that it might not be necessary to use a large number of rounds, or complicated linear filters, to ensure that the number of active S-boxes in all characteristics is sufficiently large. For example, if we take $w = 2$ as before, but add one more round, the bound jumps to 72.

Of course, since the bound we just derived is an upper bound, the minimal number of active S-boxes might as well be much smaller. First, some of the product terms in $f_1^\star f_2^\star + f_1^\star g_2^\star$ or $f_1^\star f_2^\star + g_1^\star f_2^\star$ might cancel out, or there might exist a $q \neq d$ for which $\mathrm{w_h}(\gamma_u) + \mathrm{w_h}(\gamma_v)$ suddenly drops. These cases are rather easy to detect, though, and can be avoided during the design. A more important problem is that we have limited ourselves to a special set of characteristics, which might not necessarily include the one with the minimal number of active S-boxes. However, if the feedback and feedforward functions are sparse, and the linear filters sufficiently large, then the bound is increasingly likely to be tight. On the other hand, if the state of the generator is sufficiently small, then we can perform an efficient search for the lowest-weight characteristic without making any additional assumption.

This last approach allows to show, for example, that the smallest instance of the scheme in Fig. 4(b) for which the bound of 16 is actually attained, consists of two 11th-order linear filters with

$$\begin{aligned} f_1^\star(D) &= 1 + D^{10}\,, & g_1^\star(D) &= D^{11} \cdot (D^{-3} + 1)\,, \\ f_2^\star(D) &= 1 + D^{9}\,, & g_2^\star(D) &= D^{11} \cdot (D^{-8} + 1)\,. \end{aligned}$$

5.4 Linear Characteristics and Correlations

In the sections above, we have tried to increase the number of active S-boxes of linear characteristics. We now briefly discuss how this number affects the correlation of key stream bits. This problem is treated in several papers in the context of block ciphers (see, e.g., [6]).

We start with the observation that the minimum number of active S-boxes $\mathcal{W}_{\min}$ imposes a bound on the correlation c_c of a linear characteristic:

$$c_c^2 \leq (c_s^2)^{\mathcal{W}_{\min}},$$

where c_s is the largest correlation (in absolute value) between the input and the output values of the S-box. The squares c_c^2 and c_s^2 are often referred to as *linear probability*, or also *correlation potential*. The inverse of this quantity is a good measure for the amount of data that the attacker needs to observe in order to detect a correlation.

What makes the analysis more complicated, however, is that many linear characteristics can contribute to the correlation of the same combination of key stream bits. This occurs in particular when the scheme operates on words, in which case there are typically many possible choices for the coefficients of the intermediate selection polynomials describing the characteristic (this effect is called *clustering*). The different contributions add up or cancel out, depending on the signs of c_c. If we now assume that these signs are randomly distributed, then we can use the approach of [6, Appendix B] to derive a bound on the expected correlation potential of the key stream bits:

$$E(c^2) \leq (c_s^2)^{\mathcal{W}_{\min}-n}. \tag{6}$$

The parameter n in this inequality represents the number of degrees of freedom in the choice for the coefficients of the intermediate selection polynomials.

For the characteristics propagating through the construction presented in Sect. 5.3, one will find, in non-degenerate cases, that the values of $n = r \cdot (r-1) \cdot w^{r-1}$ non-zero coefficients can be chosen independently. Hence, for example, if we construct a scheme with $w = 2$ and $r = 3$, and if we assume that it attains the bound given in (5), then we expect the largest correlation potential to be at most $c_s^{2 \cdot 48}$. Note that this bound is orders of magnitude higher than the contribution of a single characteristic, which has a correlation potential of at most $c_s^{2 \cdot 72}$.

Remark 1. In order to derive (6), we replaced the signs of the contributing linear characteristics by random variables. This is a natural approach in the case of block ciphers, where the signs depend on the value of the secret key. In our case, however, the signs are fixed for a particular scheme, and hence they might, for some special designs, take on very peculiar values. This happens for example when $r = 2$, w is even, and all non-zero coefficients of f_i and g_i equal 1 (as in the example at the end of the previous section). In this case, all signs will be positive, and we obtain a significantly worse bound:

$$c^2 \leq (c_s^2)^{\mathcal{W}_{\min}-2 \cdot n}.$$

6 Trivium

In this final section, we present an experimental 80-bit key stream cipher based on the approach outlined above. Because of space restrictions, we limit ourselves to a very rough sketch of some basic design ideas behind the scheme. The complete specifications of the cipher, which was submitted to the eSTREAM Stream Cipher Project under the name Trivium, can be found at `http://www.ecrypt.eu.org/stream/` [8].

6.1 A Bit-Oriented Design

The main idea of Trivium's design is to turn the general scheme of Sect. 5.3 into a bit-oriented stream cipher. The first motivation is that bit-oriented schemes are typically more compact in hardware. A second reason is that, by reducing the word-size to a single bit, we may hope to get rid of the clustering phenomenon which, as seen in the previous section, has a significant effect on the correlation.

Of course, if we simply apply the previous scheme to bits instead of words, we run into the problem that the only two existing 1×1-bit S-boxes are both linear. In order to solve this problem, we replace the S-boxes by a component which, from the point of view of our correlation analysis, behaves in the same way: an exclusive OR with an external stream of unrelated but biased random bits. Assuming that these random bits equal 0 with probability $(1 + c_s)/2$, we will find as before that the output of this component correlates with the input with correlation coefficient c_s.

The introduction of this artificial 1×1-bit S-box greatly simplifies the correlation analysis, mainly because of the fact that the selection polynomial at the output of an S-box is now uniquely determined by the input. As a consequence, we neither need to make special assumptions about the values of the non-zero coefficients, nor to consider the effect of clustering: the maximum correlation in the key stream is simply given by the relation

$$c_{\max} = c_s^{\mathcal{W}_{\min}} . \tag{7}$$

The obvious drawback, however, is that the construction now relies on external streams of random bits, which have to be generated somehow. Trivium attempts to solve this problem by interleaving three identical key stream generators, where each generator obtains streams of biased bits (with $c_s = 1/2$) by ANDing together state bits of the two other generators.

6.2 Specifying the Parameters

Let us now specify suitable parameters for each of those three identical 'sub-generators'. Our goal is to keep all parameters as small and simple as possible, given a number of requirements.

The first requirement we impose is that the correlations in the key stream do not exceed 2^{-40}. Since each sub-generator will be fed with streams of bits

having correlation coefficient $c_s = 1/2$, we can derive from (7) that a minimum weight $\mathcal{W}_{\min}$ of at least 40 is needed. The smallest values of w and r for which this requirement could be satisfied (with a fairly large margin, in fact) are $w = 2$ and $r = 3$. As an additional requirement, we would like the minimum weight to reach the upper bound of (5) for the chosen values of w and r. In this case, this translates to the condition $\mathcal{W}_{\min} = 72$, which is fulfilled if $\mathrm{w_h}(\gamma_u) + \mathrm{w_h}(\gamma_v) + \mathrm{w_h}(\gamma_w) \geq 72$ for all $q \neq 0$, where

$$\gamma_u = q \cdot \frac{f_1^\star f_2^\star f_3^\star + f_1^\star f_2^\star g_3^\star + f_1^\star g_2^\star g_3^\star}{d}\,, \quad \gamma_v = \ldots\,, \quad \text{etc.}$$

Although the preceding sections have almost exclusively focused on linear correlations, other security properties such as periodicity remain important. Controlling the period of the scheme is difficult because of the non-linear interaction between the sub-generators, but we can try to decrease the probability of short cycles by maximizing the periods of the individual sub-generators after turning off the streams feeding their 1×1-bit S-boxes. The connection polynomial of these (completely linear) generators is given by $f_1^\star f_2^\star f_3^\star + g_1^\star g_2^\star g_3^\star$, and ideally, we would like this polynomial to be primitive. Our choice of w prevents this, though: for $w = 2$, the polynomial above is always divisible by $(D+1)^3$. Therefore, we just require that the remaining factor is primitive, and rely on the initialization of the state bits to avoid the few short cycles corresponding to the factor $(D+1)^3$ (see [8]).

Finally, we also impose some efficiency requirements. The first is that state bits of the sub-generators should not be used for at least 64/3 iterations, once they have been modified. This will provide the final scheme with the flexibility to generate up to 64 bits in parallel. Secondly, the length of the sub-generators should be as short as possible and a multiple of 32.

We can now exhaustively run over all possible polynomials $f_1^\star, \ldots, g_3^\star$ in order to find combinations for which all previous requirements are fulfilled simultaneously. Surprisingly enough, it turns out that the solution is unique:

$$\begin{aligned} f_1^\star(D) &= 1 + D^9\,, & g_1^\star(D) &= D^{31} \cdot (D^{-23} + 1)\,, \\ f_2^\star(D) &= 1 + D^5\,, & g_2^\star(D) &= D^{28} \cdot (D^{-26} + 1)\,, \\ f_3^\star(D) &= 1 + D^{15}\,, & g_3^\star(D) &= D^{37} \cdot (D^{-29} + 1)\,. \end{aligned}$$

In order to construct the final cipher, we interleave three of these sub-generators and interconnect them through AND-gates. Since the reasoning above does not suggest which state bits to use as inputs of the AND-gates, we simply choose to minimize the length of the wires. The resulting scheme is shown in Fig. 5. The 96 state bits $s_1, s_4, s_7, \ldots, s_{286}$ belong to the first sub-generator, $s_2, s_5, s_8, \ldots, s_{287}$ to the second one, etc.

6.3 Security and Efficiency Evaluation

The scheme presented above is currently being evaluated in the framework of the eSTREAM Stream Cipher Project, and to conclude this paper we briefly summarize the current status of the evaluation.

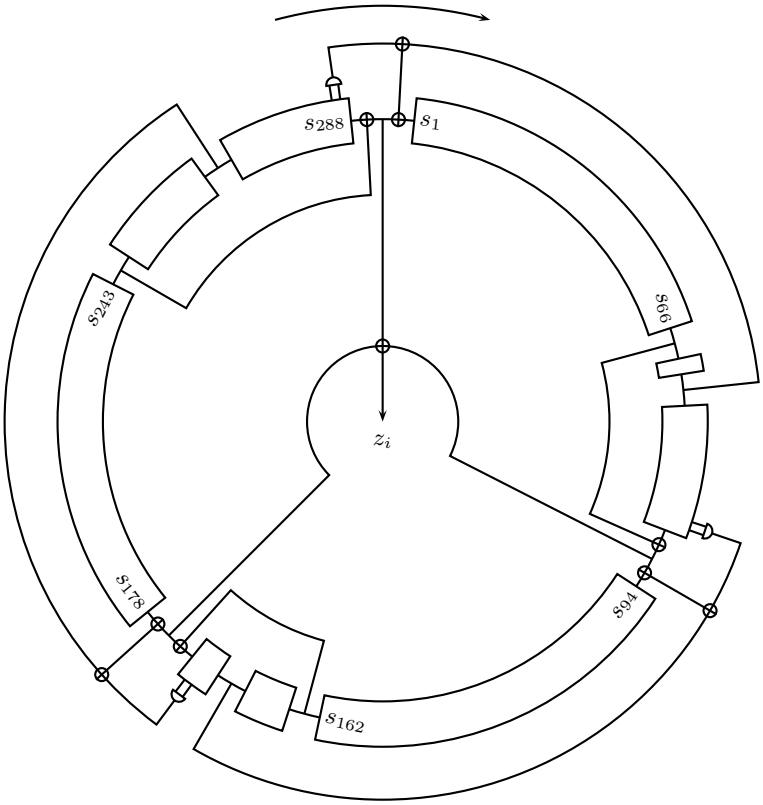

Fig. 5. Trivium

Table 1. Cryptanalytical results

Attack	Time	Data	Reference
Linear distinguisher	2^{144}	2^{144}	[8]
Guess-and-determine attack	2^{195}	288	[8]
Guess-and-determine attack	2^{135}	288	[9]
Solving system of equations	2^{164}	288	[10]
Exhaustive key search	2^{80}	80	

The complexities of the different attacks discovered so far are listed Table 1. The most efficient dedicated attack is a guess-and-determine attack presented by S. Khazaei [9]. However, with a time complexity of 2^{135}, it is still considerably less efficient than a generic exhaustive key search.

The hardware efficiency of TRIVIUM was independently evaluated by Gürkaynak et al. [11] and by Good et al. [12]. The first paper reports a 64-bit implementation in 0.25 µm 5-metal CMOS technology with a throughput per area ratio of $129\,\mathrm{Gbit/s \cdot mm^2}$, which is three times higher than for any other eSTREAM candidate. The second paper presents a compact FPGA implementation with an estimated equivalent number of gates of 2682, making TRIVIUM the second most compact candidate after Grain.

References

1. Daemen, J.: Cipher and hash function design. Strategies based on linear and differential cryptanalysis. PhD thesis, Katholieke Universiteit Leuven (1995)
2. Hawkes, P., Rose, G.G.: Primitive specification and supporting documentation for SOBER-t*w* submission to NESSIE. In: Proceedings of the First NESSIE Workshop, NESSIE (2000)
3. Ekdahl, P., Johansson, T.: SNOW – A new stream cipher. In: Proceedings of the First NESSIE Workshop, NESSIE (2000)
4. Daemen, J., Clapp, C.S.K.: Fast hashing and stream encryption with PANAMA. In Vaudenay, S., ed.: Fast Software Encryption, FSE'98. Volume 1372 of Lecture Notes in Computer Science., Springer-Verlag (1998) 60–74
5. Matsui, M.: Linear cryptanalysis method for DES cipher. In Helleseth, T., ed.: Advances in Cryptology – EUROCRYPT'93. Volume 765 of Lecture Notes in Computer Science., Springer-Verlag (1993) 386–397
6. Daemen, J., Rijmen, V.: The Design of Rijndael: AES — The Advanced Encryption Standard. Springer-Verlag (2002)
7. Rosenthal, J., Smarandache, R.: Maximum distance separable convolutional codes. Applicable Algebra in Engineering, Communication and Computing **10** (1999) 15–32
8. De Cannière, C., Preneel, B.: TRIVIUM — Specifications. eSTREAM, ECRYPT Stream Cipher Project, Report 2005/030 (2005) `http://www.ecrypt.eu.org/stream`.
9. Khazaei, S.: Re: A reformulation of TRIVIUM. Posted on the eSTREAM Forum (2006) `http://www.ecrypt.eu.org/stream/phorum/read.php?1,448`.
10. Raddum, H.: Cryptanalytic results on TRIVIUM. eSTREAM, ECRYPT Stream Cipher Project, Report 2006/039 (2006) `http://www.ecrypt.eu.org/stream`.
11. Gürkaynak, F.K., Luethi, P., Bernold, N., Blattmann, R., Goode, V., Marghitola, M., Kaeslin, H., Felber, N., Fichtner, W.: Hardware evaluation of eSTREAM candidates: Achterbahn, Grain, MICKEY, MOSQUITO, SFINKS, TRIVIUM, VEST, ZK-Crypt. eSTREAM, ECRYPT Stream Cipher Project, Report 2006/015 (2006) `http://www.ecrypt.eu.org/stream`.
12. Good, T., Chelton, W., Benaissa, M.: Review of stream cipher candidates from a low resource hardware perspective. eSTREAM, ECRYPT Stream Cipher Project, Report 2006/016 (2006) `http://www.ecrypt.eu.org/stream`.