

# Linear Approximations for 2-round Trivium

Meltem Sönmez Turan<sup>1</sup>, Orhun Kara<sup>2</sup>

<sup>1</sup> Institute of Applied Mathematics, Middle East Technical University  
Ankara, Turkey

`msonmez@metu.edu.tr`

<sup>2</sup> TUBITAK-UEKAE, Gebze, Turkey

`orhun@uekae.tubitak.gov.tr`

**Abstract.** Trivium, designed by De Cannière and Preneel, is one of the focus ciphers of Phase II for the eSTREAM project. In this paper, we model the initialization part of Trivium as an 8-round function where each round consists of 144 Trivium clocks, and analyze the security margin in terms of number of rounds. This is an open question. Nevertheless, we give some partial answers. As one example, we apply Matsui’s linear cryptanalysis to 2-round Trivium and give a linear approximation with bias  $2^{-31}$ . In addition, we analyze the completeness property of the initialization function. We propose a new input to the initialization of Trivium that has better diffusion properties. However, the security margin of the new proposal is also an open question. We conjecture that an  $R$ -round Trivium is secure if each register bit is affected by all the key and IV bits in  $R$  round.

**Keywords:** Linear approximations, Trivium, Stream ciphers

## 1 Introduction

Linear cryptanalysis, introduced by Matsui [1], is an effective known plaintext attack against block ciphers. It exploits some statistical correlations between input and output bits. For a block cipher with  $k$ -bit key  $(k_1, \dots, k_k)$ ,  $n$ -bit plaintext  $(p_1, \dots, p_n)$  and ciphertext  $(c_1, \dots, c_n)$ , the aim the attack is to find the index sets  $I, J, L$  such that

$$\sum_{i \in I} k_i + \sum_{j \in J} p_j = \sum_{l \in L} c_l \quad (1)$$

holds with probability  $p = 1/2 + \epsilon$ ,  $\epsilon \neq 0$ .

Some variants of linear cryptanalysis are applied to stream ciphers. The most famous example may be the correlation attacks mounted on LFSR based stream ciphers [2] [3] [4]. Some linear approximations between key and keystream bits are utilized. Another example is proposed by Golić [5]. In [6], a linear approximation for  $t$ -functions is used to attack the TSC stream ciphers. In [7], a new method to find biased linear approximations without searching all possible linear relations individually is presented and is used to distinguish the output of the stream cipher Pomaranch.

In this paper, we introduce a new version of linear cryptanalysis on stream ciphers. The analysis is a kind of resynchronization attack. We consider the initialization phase (key and IV loading) of a stream cipher as an iterated function. Then, we apply Matsui’s linear cryptanalysis to the initialization phase by finding approximations for each iteration and combining them by piling-up lemma. As an example, we consider the initialization phase of Trivium as 8-round function and find a linear approximation for 2-round Trivium with a bias of  $2^{-31}$  for a subset of key and IVs.

Trivium [8], one of the focus algorithms in eSTREAM project, is a synchronous binary additive stream cipher. Previously, two attacks have been presented for the analysis of Trivium and none of them has complexity less than exhaustive search. In [9], the linear sequential circuit approximations are used to evaluate the strength of Trivium against distinguishing attacks. The correlation coefficient is calculated as  $2^{-72}$ , and the complexity to distinguish the output is  $O(2^{144})$ . According to the paper, it is not possible to find a linear function with correlation coefficient larger than  $2^{-40}$  using linear sequential circuit approximations. Another study [10] tries to find 288 unknown internal state bits by solving systems of equations. Solving this system has complexity  $O(2^{164})$ , which is more than exhaustive search complexity. Also, in terms of randomness properties, no statistical weaknesses are observed [11].

We concentrate on the initialization of Trivium which consists of 1152 clockings. Trivium is one of the fastest ciphers proposed in eSTREAM project [12]. However, initialization of Trivium with 1152-clockings may hinder the speed in platforms where resynchronization is performed very often. For instance, frequent initializations of Trivium may slow down it more than five times in a frame based encryption like GSM over-the-air privacy standard since length of each frame is 228 bits. We propose a new input to the initialization function of Trivium which provides a faster diffusion of key bits and IV bits into the register. Moreover, we introduce some open problems on security margins of Trivium. We conjecture that  $R$ -round Trivium is secure if each register bit is affected by all the key and IV bits in  $R$ -round.

In the following section, the framework of finding linear approximations is presented. In section 3, description of Trivium is given. In Section 4, a linear approximation for 2-round Trivium is presented. A new proposal for initialization by which it gets harder to find linear approximations, is given in Section 5 and conclusion and future studies are summarized in the last section.

## 2 Framework

Stream ciphers can be treated as a collection of Boolean functions,

$$F_i : F_2^k \times F_2^v \rightarrow F_2, i = 1, 2, \dots$$

that generate  $z_i$ , the  $i^{th}$  output bit of keystream using  $k$  bit key and  $v$  bit IV. Each  $F_i$  is affected from both initialization and keystream generation phases. A linear approximation to  $F_i$ ’s with bias  $\epsilon > 2^{-k/2}$  would return one bit key information, if it is possible to resynchronize the cipher  $\epsilon^{-2}$  times.

## 2.1 Searching for Linear Approximations

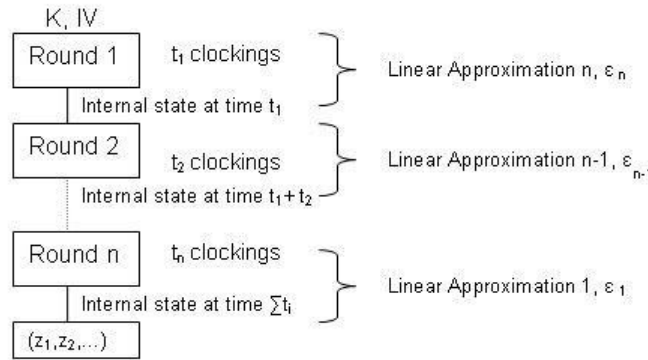
Searching for linear approximations is composed of three steps:

(i) **Selecting a subset of  $z_i$ .** In this step, the aim is to find the right hand side of the linear approximation. Selection of the subset of  $z_i$ 's or equivalently the subset of  $F_i$ 's is done such that  $\sum z_i$  or  $\sum F_i$  is affected from minimum number of internal state variables.

(ii) **Partitioning the initialization phase.** Initialization phase of stream ciphers consists of iteratively applying the same next state function to the internal state variables. To find a linear approximation for  $F_i$  efficiently, the initialization phase is partitioned into rounds with  $t_i$  clockings so that for each round it is possible to find approximations efficiently.

As given in Figure 1, the initialization phase of the cipher can be represented as  $n$  rounds, where  $t_i$  is the number of clockings in each round. The sum of  $t_i$ 's should be equal to the total number of clockings,  $T$ , in initialization.

In the extreme case, each round is composed of one clockings, then for each round, linear approximations with high biases can be found easily. However, as  $T$  gets large, the approximation for the whole cipher is likely to have a very small bias. On the other hand, if the number of clockings in each round is chosen to be very high, then finding linear approximations for each round becomes infeasible. This gives a trade-off between number of rounds and the selection of  $t_i$  values. Optimal selection of  $t_i$  values is an open question.



**Fig. 1.** Stream cipher composed of rounds

(iii) **Combining linear approximations.** The approximations found for each round are combined to find an approximation for the whole cipher. Approximations are selected so that all internal state bits are canceled out. Finally, the linear approximation based on key, IV and keystream is obtained. Bias of the approximation is found by using the piling-up lemma.

The found bias should be greater than  $2^{-k/2}$ . It is also possible to fix some of the IV bits to increase the bias, however this puts a restriction on the number of resynchronizations. In such cases, the attack requires chosen IVs. There may be other approximations valid for a subset of keys, with better biases.

### 3 The Stream Cipher Trivium

Trivium supports the usage of 80 bit key and 80 bit IV with internal state size of 288 bits. It is claimed to be suitable to generate up to  $2^{64}$  bits of keystream from a pair of key and IV.

#### 3.1 Initialization

80 bit key  $K(k_1, k_2, \dots, k_{80})$ , and IV  $(iv_1, iv_2, \dots, iv_{80})$ , is directly assigned to the internal state of the cipher  $(s_1, s_2, \dots, s_{288})$  and the remaining bits (except the last three) are set to zero. Then, the cipher is clocked over 4 full cycles without producing any output. The pseudo code of the initialization phase is given below.

---

#### Algorithm 3.1: LOADING KEY AND IV( $K, IV$ )

---

```

( $s_1, s_2, \dots, s_{93}$ )  $\leftarrow$  ( $k_1, k_2, \dots, k_{80}, 0, \dots, 0$ );
( $s_{94}, s_{95}, \dots, s_{177}$ )  $\leftarrow$  ( $iv_1, iv_2, \dots, iv_{80}, 0, \dots, 0$ );
( $s_{178}, s_{179}, \dots, s_{288}$ )  $\leftarrow$  ( $0, \dots, 0, 0, 1, 1, 1$ );
for  $i \leftarrow 1$  to  $4 \cdot 288$ 
  do
     $t_1 \leftarrow s_{66} + s_{91} \cdot s_{92} + s_{93} + s_{171}$ ;
     $t_2 \leftarrow s_{162} + s_{175} \cdot s_{176} + s_{177} + s_{264}$ ;
     $t_3 \leftarrow s_{243} + s_{286} \cdot s_{287} + s_{288} + s_{69}$ ;
    ( $s_1, s_2, \dots, s_{93}$ )  $\leftarrow$  ( $t_3, s_1, \dots, s_{92}$ );
    ( $s_{94}, s_{95}, \dots, s_{177}$ )  $\leftarrow$  ( $t_1, s_{94}, \dots, s_{176}$ );
    ( $s_{178}, s_{179}, \dots, s_{288}$ )  $\leftarrow$  ( $t_2, s_{178}, \dots, s_{287}$ );
return

```

---

#### 3.2 Keystream Generation

Keystream generation function is very similar to key and IV loading. The only difference is the filter function that generates the keystream  $z_i, i = 1, 2, \dots$ . The pseudo code of keystream generation is given below.

---

**Algorithm 3.2:** KEYSTREAM GENERATION( $N$ )

---

```
for  $i \leftarrow 1$  to  $N$ 
do
   $t_1 \leftarrow s_{66} + s_{93}$ ;
   $t_2 \leftarrow s_{162} + s_{177}$ ;
   $t_3 \leftarrow s_{243} + s_{288}$ ;
   $z_i \leftarrow t_1 + t_2 + t_3$ ;
   $t_1 \leftarrow t_1 + s_{91} \cdot s_{92} + s_{171}$ ;
   $t_2 \leftarrow t_2 + s_{175} \cdot s_{176} + s_{264}$ ;
   $t_3 \leftarrow t_3 + s_{286} \cdot s_{287} + s_{69}$ ;
   $(s_1, s_2, \dots, s_{93}) \leftarrow (t_3, s_1, \dots, s_{92})$ ;
   $(s_{94}, s_{95}, \dots, s_{177}) \leftarrow (t_1, s_{94}, \dots, s_{176})$ ;
   $(s_{178}, s_{179}, \dots, s_{288}) \leftarrow (t_2, s_{178}, \dots, s_{287})$ ;
return
```

---

## 4 Linear Approximations for 2-round Trivium

The initialization phase of Trivium can be modeled as a block cipher with  $n$ -rounds. As a result of making a trade-off between number of rounds and number of clockings in each round, the number of clockings in each round is chosen to be 144. The more number of clockings in each round, the better approximations are found, but number of clockings should not be too large to prevent finding linear approximations exhaustively.

For 2-round Trivium, the followings

$$\begin{aligned} K &= (s_0(1), s_0(2), \dots, s_0(80)), \\ IV &= (s_0(94), s_0(95), \dots, s_0(173)), \\ z_1 &= s_{288}(66) + s_{288}(93) + s_{288}(162) + s_{288}(177) + s_{288}(243) + s_{288}(288) \end{aligned}$$

hold where  $s_t(i)$  is the  $i^{\text{th}}$  internal state bit at time  $t$ .

To check for possible trivial weaknesses of 2-round Trivium, diffusion of IV and key to internal state bits are examined using 1000 random key and IV pairs. All key and IV bits are diffused to all internal state bits, except the last seven internal state bits. However, this does not lead to any trivial weakness.

In this study, while selecting the subset of  $F_i$ 's to approximate, the only restriction is the total number of internal state variables that affect the keystream bits. Each  $z_i$  is generated using the modulo 2 summation of six internal state bits. A subset of  $z_i$ 's such that their summation includes less than six internal state bits after cancellations is not found. Therefore, the function to be approximated is chosen to be  $F_1$  that generates the first output bit,  $z_1$ , in terms of key and IV bits.

For 2-round Trivium, finding the equation  $F_1$  in terms of initial state variables is not efficient, therefore an approximation is found for each round and then

they are combined to find an approximation as given in Figure 2. The bias of the obtained approximation is found by the piling-up lemma.

The output bit  $z_1$  is the sum of bits  $s_{288}(66)$ ,  $s_{288}(93)$ ,  $s_{288}(162)$ ,  $s_{288}(177)$ ,  $s_{288}(243)$  and  $s_{288}(288)$ . The algebraic normal form of  $F_1$  is found exhaustively in terms of the internal state bit of  $t = 144$  as

$$\begin{aligned} z_1 = & s_{144}(6) + s_{144}(16).s_{144}(117) + s_{144}(31)s_{144}(32) + s_{144}(33) + s_{144}(57) + \\ & s_{144}(82).s_{144}(83) + s_{144}(84) + s_{144}(96) + s_{144}(97).s_{144}(98) + s_{144}(99) + \\ & s_{144}(111) + s_{144}(129) + s_{144}(142).s_{144}(143) + s_{144}(144) + s_{144}(150) + \\ & s_{144}(162) + s_{144}(163).s_{144}(164) + s_{144}(165) + s_{144}(186) + s_{144}(192) + \\ & s_{144}(208).s_{144}(209) + s_{144}(210) + s_{144}(231) + s_{144}(235).s_{144}(236) + \\ & s_{144}(237) + s_{144}(252) \end{aligned}$$

and its closest linear approximation is

$$\begin{aligned} z_1 = & s_{144}(6) + s_{144}(33) + s_{144}(57) + s_{144}(84) + s_{144}(96) + s_{144}(99) + \\ & s_{144}(111) + s_{144}(129) + s_{144}(144) + s_{144}(150) + s_{144}(162) + \quad (2) \\ & s_{144}(165) + s_{144}(186) + s_{144}(192) + s_{144}(210) + s_{144}(231) + \\ & s_{144}(237) + s_{144}(252) \end{aligned}$$

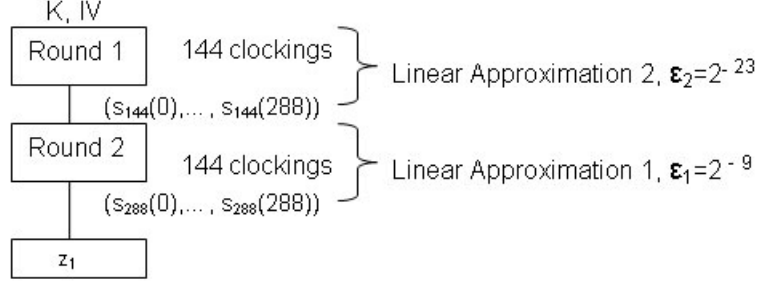
with bias  $1/2 + 2^{-9}$ .

Since the aim is to obtain an approximation based on key, IV and output bits, the linear approximation given above is rewritten in terms of  $s_0(i)$ ,  $i = 1, 2, \dots, 80$  and  $i = 94, \dots, 173$  values, the remaining terms are omitted, since they are assigned to constants during initialization. Then, the equation given in Appendix A is obtained. The equation has 24 linear, 59 quadratic terms, 20 terms with degree 3. The linear approximation for the function is found as

$$\begin{aligned} z_1 = & 1 + s_0(3) + s_0(6) + s_0(15) + s_0(21) + s_0(27) + s_0(30) + s_0(39) + \\ & s_0(54) + s_0(57) + s_0(67) + s_0(68) + s_0(69) + s_0(72) + s_0(96) + \quad (3) \\ & s_0(99) + s_0(114) + s_0(117) + s_0(123) + s_0(126) + s_0(132) + s_0(138) + \\ & s_0(144) + s_0(165) + s_0(171) \end{aligned}$$

with bias  $2^{78} \cdot (0.25)^{59} \cdot (0.375)^{20} = 2^{-68.30}$ , assuming all nonlinear terms are independent. We increase the amount of the bias by assigning zero string to certain IV and key bits.

**Chosen IVs** For IVs in the form  $iv_{25} = iv_{26} = iv_{31} = iv_{32} = iv_{49} = iv_{50} = iv_{54} = iv_{55} = iv_{70} = iv_{71} = 0$ , the bias of the equation increases to  $2^{-44}$ . This bias is still very low and cannot be used to break 2-round Trivium. To improve bias further, also some of the key bits are fixed. Then, the bias of the second linear approximation increases to  $2^{-23}$  for keys satisfying  $k_{14} = k_{19} = k_{20} = k_{38} = k_{39} = k_{45} = k_{63} = k_{64} = k_{65} = k_{77} = 0$ .



**Fig. 2.** Linear Approximations for 2-round Trivium

Combining two linear approximations (2) and (3), the total bias of the following approximation,

$$z_1 = 1 + k_3 + k_6 + k_{15} + k_{21} + k_{27} + k_{30} + k_{39} + k_{57} + k_{67} + k_{68} + k_{69} + k_{72} + iv_3 + iv_6 + iv_{21} + iv_{24} + iv_{30} + iv_{33} + iv_{39} + iv_{45} + iv_{51} + iv_{72} + iv_{78},$$

is obtained as  $2 \cdot 2^{-9} \cdot 2^{-23} = 2^{-31}$  by piling-up lemma. The upper bound on the number of resynchronizations is  $2^{70}$ , since 10 bits of IV bits are fixed to zero. To identify a key with specified bits, we need  $2^{62}$  chosen IV.

## 5 Proposal for Initialization

In this section, we propose a new method for initialization which is very similar to the original. The only difference is related to the initial assignment of state bits. Only 22 of the internal state variables are set to constants and this change does not increase the cost significantly. This obviously increases the number of variables while searching for linear approximations and therefore it gets harder to find linear approximations. As a result, it may be possible to decrease the number of initial clockings.

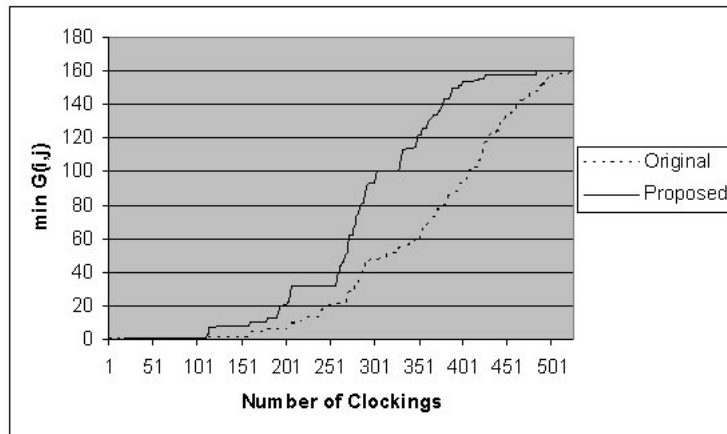
The proposed initial assignment is

$$\begin{aligned} (s_1, \dots, s_{93}) &\leftarrow (iv_1, \dots, iv_{13}, iv_{14} + k_1, \dots, iv_{80} + k_{67}, k_{68}, \dots, k_{80}), \\ (s_{94}, \dots, s_{177}) &\leftarrow (iv_1 + k_1, \dots, iv_{80} + k_{80}, 0, 0, 0, 0), \\ (s_{178}, \dots, s_{288}) &\leftarrow (k_1, \dots, k_{13}, k_{14} + iv_1, \dots, k_{80} + iv_{67}, iv_{68}, \dots, iv_{80}, 0, \dots, 0, 1, 1, 1). \end{aligned}$$

Let us note that we propose 13 shifts while loading key bits into the first register and 13 shifts while loading IV bits into the third register. The number

of shifts may be chosen something else. However, same key bits or same IV bits should not be XORed in the feedback functions of the registers during the first 80-90 clockings.

A comparison of the proposed and original method is done in terms of the completeness property. Let  $G(i, j)$  be the number of key and IV bits that affect the state bit  $i$  after  $j$  clockings. The comparison of both methods is done based on  $\min_i G(i, j)$  and as seen from Figure 3 the diffusion of key and IV bits are better in the proposed method. In the original method, completeness is satisfied after 525 clockings, whereas in the proposed method, 484 clockings are enough.



**Fig. 3.** Number of clockings vs.  $\min_i G(i, j)$

## 6 Conclusion and Further Study

In this study, we mainly concentrated on the initialization of Trivium which is one of the focus ciphers of eSTREAM project. We modeled the initialization phase of Trivium as an iterated cipher with 8 rounds. For frame based applications requiring frequent resynchronizations, we question the efficiency of the initialization phase and try to attack initialization with smaller rounds. For 2-round Trivium, we obtained a linear approximation of  $z_1$ , which is valid for a subset of key and IV's. It is an open question whether there exists linear approximations for  $r$ -round Trivium where  $r > 2$ .

As a list of future studies, the followings can be given:

- *Use of multiple approximations.* As an extension of linear cryptanalysis, in [13], the use of multiple linear approximations is proposed. As a future work, different approximations for the same output bits can be found and combined to make better approximations.

- *Use of nonlinear approximations.* As an alternative to linear approximations, nonlinear approximations may be applied to the initialization phase to find better approximations.
- *Different modelings of initialization.* The initialization phase can be remodeled differently, using different number of clockings in each round and better approximations may be found.

The most important problem is determining the security margin of Trivium. That is, what is the minimum number of rounds for the initialization of Trivium so as to supply 80-bit security? The same question can be given for the new proposal. Let the initialization function be complete in R-round. That is, each register bit is affected by all key and IV bits. Then, we conjecture that R-round Trivium is secure. Therefore, using the both initialization methods, a 4-round Trivium is expected to be secure.

## A $F_1$ for 2-round Trivium

$$\begin{aligned}
z_1 = & 1 + s_0(3) + s_0(6) + s_0(15) + s_0(21) + s_0(27) + s_0(30) + s_0(39) + s_0(54) + \\
& s_0(57) + s_0(67) + s_0(68) + s_0(69) + s_0(72) + s_0(96) + s_0(99) + s_0(114) + s_0(117) + \\
& s_0(123) + s_0(126) + s_0(132) + s_0(138) + s_0(144) + s_0(165) + s_0(171) + s_0(4).s_0(5) + \\
& s_0(13).s_0(14) + s_0(13).s_0(41) + s_0(13).s_0(119) + s_0(14).s_0(40) + s_0(14).s_0(118) + \\
& s_0(16).s_0(17) + s_0(19).s_0(20) + s_0(19).s_0(47) + s_0(19).s_0(125) + s_0(20).s_0(46) + \\
& s_0(20).s_0(124) + s_0(22).s_0(23) + s_0(25).s_0(26) + s_0(28).s_0(39) + s_0(34).s_0(35) + \\
& s_0(37).s_0(38) + s_0(37).s_0(65) + s_0(37).s_0(143) + s_0(38).s_0(64) + s_0(39).s_0(40) + \\
& s_0(38).s_0(142) + s_0(40).s_0(119) + s_0(41).s_0(118) + s_0(43).s_0(44) + s_0(45).s_0(46) + \\
& s_0(46).s_0(125) + s_0(47).s_0(124) + s_0(49).s_0(50) + s_0(52).s_0(53) + s_0(58).s_0(59) + \\
& s_0(58).s_0(164) + s_0(59).s_0(163) + s_0(61).s_0(62) + s_0(63).s_0(64) + s_0(64).s_0(65) + \\
& s_0(64).s_0(143) + s_0(64).s_0(170) + s_0(65).s_0(169) + s_0(65).s_0(142) + s_0(67).s_0(68) + \\
& s_0(70).s_0(71) + s_0(76).s_0(77) + s_0(79).s_0(77) + s_0(103).s_0(104) + s_0(106).s_0(107) + \\
& s_0(118).s_0(119) + s_0(124).s_0(125) + s_0(127).s_0(128) + s_0(130).s_0(131) + \\
& s_0(133).s_0(149) + s_0(134).s_0(148) + s_0(142).s_0(143) + s_0(147).s_0(148) + \\
& s_0(151).s_0(152) + s_0(154).s_0(155) + s_0(160).s_0(161) + s_0(163).s_0(164) + \\
& s_0(166).s_0(167) + s_0(13).s_0(39).s_0(40) + s_0(14).s_0(38).s_0(39) + \\
& s_0(19).s_0(45).s_0(46) + s_0(20).s_0(44).s_0(45) + s_0(37).s_0(63).s_0(64) + \\
& s_0(38).s_0(39).s_0(40) + s_0(38).s_0(39).s_0(41) + s_0(38).s_0(39).s_0(119) + \\
& s_0(38).s_0(62).s_0(63) + s_0(39).s_0(40).s_0(118) + s_0(44).s_0(45).s_0(46) + \\
& s_0(44).s_0(45).s_0(47) + s_0(44).s_0(45).s_0(125) + s_0(45).s_0(46).s_0(124) + \\
& s_0(62).s_0(63).s_0(64) + s_0(62).s_0(63).s_0(65) + s_0(62).s_0(63).s_0(143) + \\
& s_0(63).s_0(64).s_0(142) + s_0(133).s_0(147).s_0(148) + s_0(134).s_0(146).s_0(147)
\end{aligned}$$

with bias  $2^{-9}$ .

## References

1. M. Matsui. Linear cryptanalysis method for DES cipher. In *EUROCRYPT*, pages 386–397, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.
2. T. Siegenthaler. Decrypting a class of stream ciphers using ciphertext only. *IEEE Trans. Computers*, 34(1):81–85, 1985.
3. W. Meier and O. Staffelbach. Fast correlation attacks on certain stream ciphers. *Journal of Cryptology*, 1(3):159–176, 1989.
4. V. V. Chepyzhov, T. Johansson, and B. Smeets. A simple algorithm for fast correlation attacks on stream ciphers. In *Fast Software Encryption*, pages 181–195, London, UK, 2001. Springer-Verlag.
5. J. Dj. Golic. Linear cryptanalysis of stream ciphers. In *Fast Software Encryption*, pages 154–169, 1994.
6. F. Muller and T. Peyrin. Linear cryptanalysis of the TSC family of stream ciphers. In *ASIACRYPT*, pages 373–394, 2005.
7. M. Hell and T. Johansson. On the Problem of Finding Linear Approximations and Cryptanalysis of Pomaranch Version 2. In *SAC*, 2006.
8. C. De Cannière and B. Preneel. Trivium - a stream cipher construction inspired by block cipher design principles. eSTREAM, ECRYPT Stream Cipher Project, Report 2005/030, 2005. <http://www.ecrypt.eu.org/stream>.
9. S. Khazaei, M. M. Hasanzadeh, and M. S. Kiaei. Linear Sequential Circuit Approximation of Grain and Trivium Stream Ciphers. eSTREAM, ECRYPT Stream Cipher Project, Report 2005/063, 2005.
10. H. Raddum. Cryptanalytic results on trivium. eSTREAM, ECRYPT Stream Cipher Project, Report 2006/039, 2006.
11. M. S. Turan, A. Doğanaksoy, and Ç. Çalk. Statistical analysis of synchronous stream ciphers. *SASC 2006: Stream Ciphers Revisited*, 2006.
12. 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. eSTREAM, ECRYPT Stream Cipher Project, Report 2006/015, 2006.
13. Jr. B. S. Kaliski and M. J. B. Robshaw. Linear cryptanalysis using multiple approximations. In *CRYPTO*, pages 26–39, London, UK, 1994. Springer-Verlag.