Period of Streamcipher *Edon80*

Jin Hong

National Security Research Institute
161 Gajeong-dong, Yuseong-gu, Daejeon, 305-350, Korea
jinhong@etri.re.kr

Abstract. The period of a recent streamcipher proposal *Edon80* is analyzed. Even though the average period may be quite large, we show that for a randomly chosen key and IV pair, there exists a non-dismissible probability that the produced keystream will be of very short period.

Keywords: *Edon80*, streamcipher, period

1 Introduction

Edon80 [3] is one of the streamciphers submitted to eSTREAM, the ECRYPT streamcipher project [1]. It was one of the ciphers chosen for presentation at the Symmetric Key Encryption Workshop (SKEW, Århus, Denmark, May, 2005) and rests on ideas previously presented at FSE 2005 [2]. The core of the cipher consists of what is called a *quasigroup string e-transformation,* and using related theory previously developed, the designers present some provable results supporting its security.

In this short paper, we study *Edon80*, focusing on its period. The designers had projected a period of 2^{103}, and even though this may be true on average, we show that there exists a non-negligible probability that the keystream will fall into a much shorter period. For example, with random use of key and IV, keystreams of period as short as 2^{55} may occur with probability 2^{-71} and the existence of at least one key-IV pair producing a period-2^{11} keystream can be expected.

2 *Edon80*

The streamcipher *Edon80* [3] will be described briefly in this section. It is a hardware oriented streamcipher with intended security level corresponding to 80 bits. Keys of 80-bit size and IVs of 64-bit size are used.

Quasigroup The first ingredient of *Edon80* is four quasigroups of order 4. The quasigroup operators are given explicitly in Table 1. If you are not familiar with quasigroups, you can simply think of these as four different collections of (possibly non-commutative and non-associative) multiplication rules $\bullet_i$, defined on sets of four elements. Notice that each of the four operators are indexed by a 2-bit number.

$\bullet_0$	0	1	2	3
0	0	2	1	3
1	2	1	3	0
2	1	3	0	2
3	3	0	2	1

$\bullet_1$	0	1	2	3
0	1	3	0	2
1	0	1	2	3
2	2	0	3	1
3	3	2	1	0

$\bullet_2$	0	1	2	3
0	2	1	0	3
1	1	2	3	0
2	3	0	2	1
3	0	3	1	2

$\bullet_3$	0	1	2	3
0	3	2	1	0
1	1	0	3	2
2	0	3	2	1
3	2	1	0	3

Table 1. Quasigroups

KeySetup The 80-bit key K is used to select 80 sequential quasigroups that are to be used for keystream production. The key is first divided into 40-many 2-bit subkeys.

$$K = K_0||K_1||\cdots||K_{39}.$$

Then each of the working quasigroup operators $*_i$ $(i = 0, 1, \ldots, 79)$ are assigned to be one of $\bullet_j$ $(j = 0, \ldots, 3)$. Explicitly, recalling that each quasigroup operator is indexed by a 2-bit number, we set

$$*_i \leftarrow \begin{cases} \bullet_{K_i} & 0 \le i < 40, \\ \bullet_{K_{i-40}} & 40 \le i < 80. \end{cases}$$

Hence the key K determines the working quasigroups completely, and any consecutive 40 operators $*_i$ determine the key completely.

IVSetup For any fixed key, *IVSetup* sends a 64-bit IV to a finite sequence $(a_0, \ldots, a_{79})$, where each a_i is a 2-bit value, in a key-dependent manner. For discussions of this paper, we will not need to know its actual inner workings, but we shall assume that it is well-designed, meaning that some sort of randomness can be expected of the process.

Keystream generation Consider the array of quasigroup elements given in Table 2. It consists of 81 rows, each row is a sequence of quasigroup elements

$*_i$		0	1	2	3	0	1	2	...
$*_0$	a_0	$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$a_{0,4}$	$a_{0,5}$	$a_{0,6}$	...
$*_1$	a_1	$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$	$a_{1,6}$	...
$*_2$	a_2	$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$	$a_{2,6}$	...
$*_3$	a_3	$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$a_{3,4}$	$a_{3,5}$	$a_{3,6}$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$					
$*_{78}$	a_{78}	$a_{78,0}$	$a_{78,1}$	$a_{78,2}$	$a_{78,3}$	$a_{78,4}$	$a_{78,5}$	$a_{78,6}$	...
$*_{79}$	a_{79}	$a_{79,0}$	$a_{79,1}$	$a_{79,2}$	$a_{79,3}$	$a_{79,4}$	$a_{79,5}$	$a_{79,6}$	...

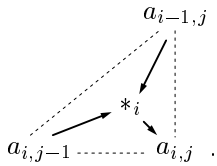
Table 2. Keystream generation

extending infinitely to the right, and the top row is a fixed repeating pattern of

period-4. The first column contains the 80 quasigroup operations $*_i$ previously determined from key. The next column contains the finite sequence obtained from the *IVSetup* process. The rest of the elements $a_{i,j}$ are obtained sequentially starting from the top left corner through quasigroup operations. More explicitly, we set

$$a_{i,j} = *_i(a_{i,j-1}, a_{i-1,j}),$$

where we take $a_{-1,j} = j \pmod 4$ to be the top row and where $a_{i,-1} = a_i$ is the second column. Pictorially, we can view this as



Finally, the keystream itself is given as every other element of the bottom row.

$$\text{keystream} = (a_{79,1}, a_{79,3}, a_{79,5}, \dots).$$

Our discussion will center mostly on the key determining the quasigroup operators $*_i$ $(i = 0, \dots 79)$ and the initial state $(a_0, \dots, a_{79})$ obtained right after the *IVSetup* operation. These two will be referred to together as *key-state* pair from now on.

Period Calculation of each new row in Table 2 is said to be a quasigroup string e-transformation. The designers of *Edon80* assert that each e-transformation increases the string period by a factor of 2.48 on average. Following along arguments of the designers, the final keystream should have period

$$4 \times (2.48)^{80} \times \frac{1}{2} \approx 2^{105.8}$$

on average. The term 4 comes from the period of the initiating sequence at the top, and the term $\frac{1}{2}$ is multiplied because only every other term of last row is used as keystream. But there seems to have been a slight miscalculation by the designers and they project a period of 2^{103} instead of 2^{106}.

No explicit restriction on the length of keystream usage is given by the designers. Hence readers are led to believe that keystreams of length up to 2^{103} bits may be used.

3 Undesirable key-state pairs

We shall instantiate Table 2 in such a way that the bottom row is a sequence of period 4. The corresponding key-state pair will result in producing a keystream of period 2.

$*_i$		0	1	2	3	0	1	2	3	0	...
$\bullet_2$	1	1	2	2	1	1	2	2	1	1	...
$\bullet_0$	2	3	2	0	2	3	2	0	2	3	...
$\bullet_3$	1	2	2	0	1	2	2	0	1	2	...
$\bullet_0$	1	3	2	1	1	3	2	1	1	3	...
$\bullet_2$	0	3	1	2	0	3	1	2	0	3	...

Table 3. Partial key-state pair of period 4

3.1 Partial key-state pairs

Consider the series of five quasigroup string e-transformations given in Table 3. Notice that the period of each row is 4. Actually, we found $166 \approx 2^{7.38}$ such 5-row key-state pairs of period 4. Through an exhaustive searching program, we counted all d-row key-state pairs of period p, for small values of d and p. The results are gathered in Table 4. The actual numbers written down in the table

d	$p = 4$	$p = 8$	$p = 16$
5	7.38	11.49	13.30
6	9.36	13.58	15.68
7	11.04	15.63	18.01
8	12.97	17.71	20.30
9	14.75	19.76	22.55
10	16.63	21.81	24.77
11	18.44	23.85	26.96
12	20.30	25.88	29.13
13	22.13	27.91	31.29
14	23.97	29.94	33.44
15	25.81	31.96	35.57
16	27.65	33.98	
17	29.49		
18	31.33		

Table 4. d-row period-p key-state pair count

are logarithms of the counts. So, for example, the first entry states that there are approximately $2^{7.38}$ key-state pairs of period 4, consisting of 5 rows.

Going down any column, one can see that the numbers increase at almost a constant rate. This is easier to see in Figure 1, which is the graph version of Table 4. Extrapolating, we obtain the values given in Table 5 for the number of 40-row key-state pairs. We can expect these values to be at least approximately true and our future discussion will not depend too much on their exact value.

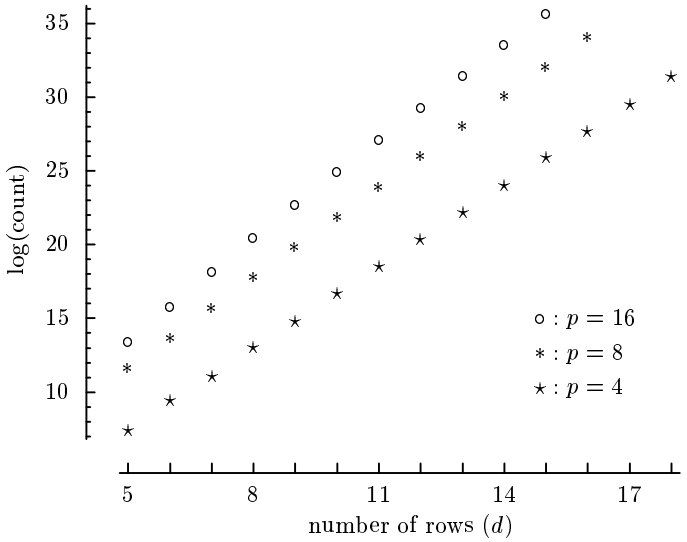


Fig. 1. Key-state pair count

	$p = 4$	$p = 8$	$p = 16$
$d = 40$	71.81	82.46	88.57

Table 5. Expected number of 40-row short period key-states

3.2 Full key-state pairs producing a period-2 keystream

Take any one of the 2^{72}-many 40-row key-state pairs of period 4. Its bottom row, in particular, is a sequence of period 4. So there is a $(1/4)^4$ probability of it being equal to the top initiating sequence $(0, 1, 2, 3, 0, \dots)$. This has been experimentally verified to hold roughly true at smaller row counts.[1] Hence we can expect there to be approximately 2^{64}-many 40-row key-state pairs having bottom row identical to the initiating sequence. In the appendix, we have written out one such partial key-state pair as an explicit example.

Now, fix any such 40-row key-state and attach another copy of the same partial key-state to its bottom. This gives an explicit instantiation for Table 2. This attaching is made possible by the fact that the 40-th row is identical to the top of the copy. We also remark that, in this argument, we did not overlook the fact that the top 40 rows will determine the key completely and hence also the quasigroup operators for the lower 40 rows. Since $*_i = *_{i+40}$ $(i = 0, \dots, 39)$, our choice of using a copy on the bottom 40 rows does not conflict with this structure of the cipher.

Recall that the actual keystream is every other quasigroup element from the bottom row sequence. Hence, we have shown the existence of at least 2^{64}-many

[1] We have reasons to believe that the slightly bigger value 1/240 reflects the actual situation better than $(1/4)^4$, but this does not affect the big picture.

(full) key-state pairs that all produce the identical keystream $(1, 3, 1, 3, \dots)$ of period 2. We make no claims as to whether these key-state pairs may be reached through normal *IVSetup* process.

4 Undesirable key-IV pairs

In this section we shall instantiate Table 2 in such a way that the bottom row is a sequence of relatively short period. There will be so many of these that a meaningful number of them will be reachable through normal state initialization process. In a way, we can see this as giving a class of *weak* key-IV pairs.

The instantiation will be done in two stages. First, the top 40 rows are filled so that the 40-th row is of period 4, 8, or 16. Then, the rest of rows are filled randomly subject to the restraints caused by the top 40 rows.

4.1 Key-state pairs producing relatively short period keystreams

Fix any 40-row key-state pair of period 4. In particular, the bottom row is a sequence of period 4, which may or may not be equal to the top initiating sequence. These 40-rows determine the key completely, and hence also the quasigroup operators $*_i$ $(i = 40, \dots, 79)$ for the remaining lower 40 rows. Let us fix these lower row operators accordingly and fill in the remaining 40 initial states $(a_{40}, \dots, a_{79})$ with arbitrary quasigroup elements.

Following along the arguments of the cipher designers, we can expect a period of $4 \times (2.48)^{40} \approx 2^{54.41}$ at the bottom 80-th row. This leads to a keystream of period $2^{53.41}$ which is much smaller than the value 2^{103} projected by the designers of *Edon80* and also smaller than even the intended security level 2^{80}.

Since the 40-row key-state pairs were approximately 2^{72} in number (Table 5), and since we have 80-bit freedom coming from the choice of quasigroup elements filling the bottom 40 rows, we can expect the existence of at least

- 2^{72+80} key-state pairs producing keystreams of period 2^{53}.

A more exact statement would be that there exists a group of 2^{72+80} key-state pairs whose average period is 2^{53}. But we shall be a bit sloppy and express this as in the above.

If we start with 40-row key-state pairs of period 8, or 16, we obtain the existence of at least

- 2^{82+80} key-state pairs producing period-2^{54} keystreams and
- 2^{89+80} key-state pairs producing period-2^{55} keystreams,

respectively.

4.2 Key-IV pairs producing relatively short period keystreams

It remains to see if any of the discussed key-state pairs producing keystreams of short period are reachable by normal *IVSetup* operation.

Recall that the *IVSetup* process is a 64-bit to 160-bit mapping for any fixed key. Hence, under the assumption that the *IVSetup* is well-designed, given any key-state pair, under random use of key and IV, the probability of it being reachable by *IVSetup* is 2^{-96}.

Thus we have the existence of at least

- 2^{56} key-IV pairs producing period-2^{53} keystreams,
- 2^{66} key-IV pairs producing period-2^{54} keystreams, and
- 2^{73} key-IV pairs producing period-2^{55} keystreams.

Since there are 2^{80+64} key-IV pairs, for a randomly chosen key-IV pair, the probability of it producing a keystream

- of period 2^{53} is at least 2^{-88},
- of period 2^{54} is at least 2^{-78}, and
- of period 2^{55} is at least 2^{-71}.

The latter two probabilities are larger than 2^{-80} and hence constitutes a valid, although certificatory, attack on *Edon80*, if keystreams of such length were allowed. Of course, if users were just given the value 2^{103} projected as period by the cipher designers, as it is for the moment, keystreams of such length would certainly be allowed.

5 Distribution of keystream periods

In this section, we show that if we look for keystreams of period slightly longer than was considered in the previous section, then they are easier to encounter during random key and IV use. The existence of key-IV pairs producing extremely short period keystreams is also shown.

5.1 Probability / period tradeoffs

We do not have to divide the 80 rows appearing in Table 2 into just top 40 and bottom 40 rows. Extrapolating Table 4, one can come up with Table 6 that gives expected number of, say, 34-row key-state pairs of short period.

	$p = 4$	$p = 8$	$p = 16$
$d = 34$	60.77	70.34	75.85

Table 6. Expected number of 34-row short period key-states

One could start with any of these 34-row key-state pairs, fill the six rows from the 35-th to 40-th with random key-state values, fill lower 40 row quasigroup

operators as defined by the upper 40 rows, and finally fill the rest of the states with random values. This would gives us $4 \cdot 6 + 80 = 104$ bits of freedom.

Since $4 \times (2.48)^{46} \times \frac{1}{2} \approx 2^{61.28}$, and since we loose 96-bit freedom from relating key-state pairs to key-IV pairs, we have the existence of following key-IV pairs.

- 2^{69} key-IV pairs producing period-2^{61} keystreams.
- 2^{78} key-IV pairs producing period-2^{62} keystreams.
- 2^{84} key-IV pairs producing period-2^{63} keystreams.

All of these periods are still small relative to both 2^{103} and 2^{80}. In terms of how often we may encounter these during random use of key and IV, we can expect higher than

- 2^{-75} probability of encountering period-2^{61} keystreams,
- 2^{-66} probability of encountering period-2^{62} keystreams, and
- 2^{-60} probability of encountering period-2^{63} keystreams.

These probabilities are much larger than the intended security level 2^{-80}.

This shows that there is a tradeoff between how short a period keystream we seek and how often we can encounter it at random. The tradeoff curve is given by Figure 2 for the segment that is most interesting, i.e., when probability is greater than 2^{-80} and period is smaller than 2^{80}. As an example, the left-most "$\circ$" of

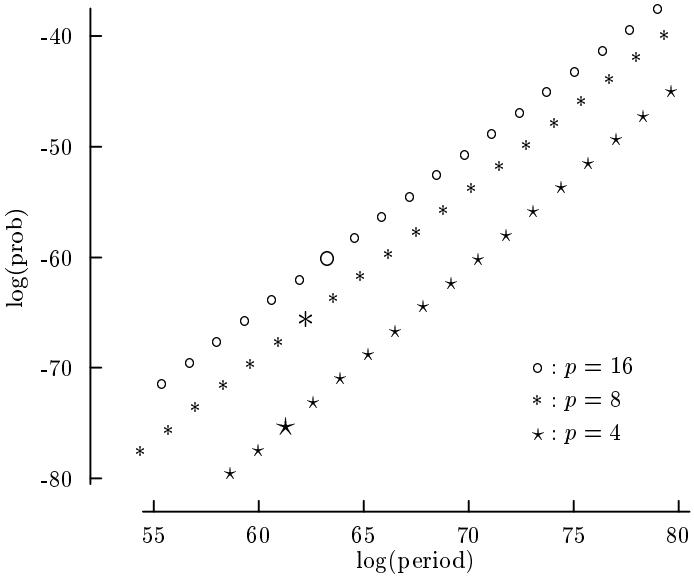


Fig. 2. Probability / period tradeoff

the graph tells us that if we use $p = 16$, we can show that with random use of key-IV pairs, one will encounter keystreams of period $2^{55.4}$ with probability no

less than $2^{-71.4}$. We have also marked the three points corresponding to $d = 34$ case, used in the above argument, with larger fonts, so that you can verify your understanding of the graph.

Notice that we are only providing a lower bound on the probability for a keystream of certain period to occur. We make no claims as to if these bounds are even close to what actually happens. For example, with p fixed, if the fact that given any divisor n of m, a period-n keystream is always a period-m sequence, is considered, one can immediately conclude that the above probability figures are much lower than what actually happens. On the orthogonal side, keystreams of same period may be obtained from multiple p values, so these can also be added and still be used as a lower bound. Furthermore, it is clear that with more computational power, we could work with $p = 32$ or $p = 64$ to obtain even better tradeoff curves.

We chose not to deal with these matters, as our rough lower bounds were already big enough to show that *Edon80* is under trouble with respect to period properties. The methods provided by this paper allows us to see the big picture, but we believe a totally different approach, for example, statistical modeling, is needed to understand the true extent of the period related problem, so as to be used on the constructive side.

5.2 Existence of key-IV pairs of very short period

We could also divide the 80 rows of Table 2 into two parts below the 40-th row.

Let us go back to Section 3.2 and first fill the top 40 rows with any one of the 2^{64}-many 40-row period-4 key-state pairs that has the $(0, 1, 2, 3, 0, \dots)$ initiating sequence at the bottom 40-th row. As before, add another copy below, but fill the quasigroup elements $a_{64}, \dots, a_{79}$ for the last 16 rows at random, so that we have an extra 32-bit degree of freedom.

Since $4 \times (2.48)^{16} \times \frac{1}{2} \approx 2^{19.97}$, we have the existence of 2^{64+32} key-IV pairs producing period-2^{20} keystreams. Recalling that probability of one of these being reachable by normal *IVSetup* process is 2^{-96}, one can expect the existence of at least one key-IV pair that leads to a keystream of extremely short period 2^{20}.

If we start with $p = 16$ key-state pairs, we can conclude that there exists at least one key-IV pair producing an even shorter period-2^{11} keystream. This is all shown in Figure 3. Although this single key-IV pair would be hard to reach at random through normal use of this cipher, it still does pose a threat, as the corresponding keystream is of extremely short period.

Once again, the counts we provide are only lower bounds. There may be ways to produce low-period sequences different from the explicit method we have considered.

Before ending this section, we remark that taking note of the top two "$\circ$" from Figure 3 can be interesting. For example, the top point tells us that there are 2^{67} key-IV pairs producing period-2^{54} keystreams. The probability of encountering one of these key-IV pairs at random is 2^{-77}, which is greater than the intended security level 2^{-80}. It is clear that working with larger p values will give additional meaningful tradeoff points.

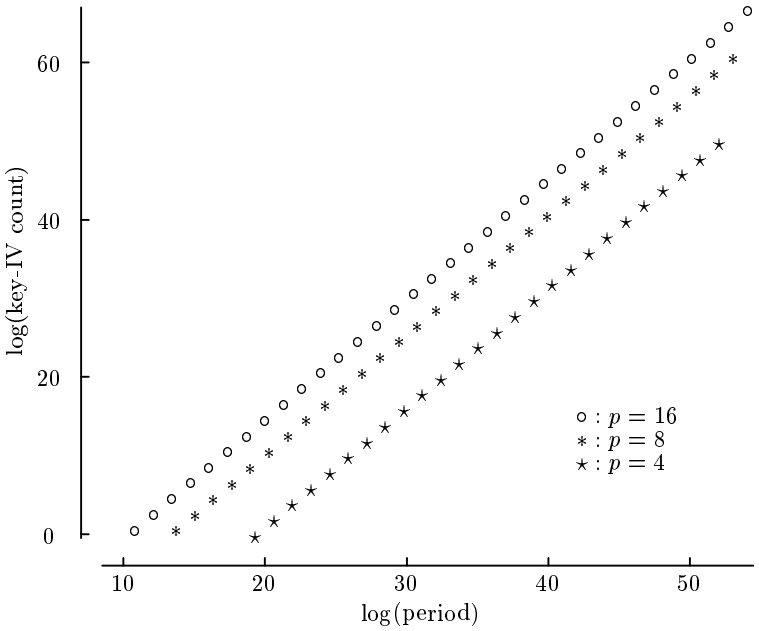


Fig. 3. Key-IV count / period tradeoff

6 Conclusion

The period property of streamcipher *Edon80* has been studied. We have shown that there are quite a large number of key-state pairs that produce identical sequence of period 2. We have also shown that there is a probability of no less than 2^{-71} for a random key-IV pair to produce a keystream of period 2^{55}. The tradeoff between period and occurrence probability was studied and a period-2^{63} keystream can be expected from a random key-IV pair with probability at least 2^{-60}. Finally, we can expect the existence of at least one key-IV pair producing the extremely short period-2^{11} keystream.

These short period keystreams occur with probability greater than 2^{-80} and the periods are very small relative to 2^{103}, which is the value designers had projected as cipher period. These numbers are smaller than even 2^{80}, which many would take for granted from an 80-bit security cipher, unless explicitly stated otherwise. Also, these key-IV pairs of bad characteristics, or *weak* key-IV pairs, are hard to categorize at the moment and hence avoiding them does not seem to be easy.

These results show that while the average period of *Edon80* may still be 2^{103} as projected, the range of keystream period is very wide with a non-dismissible portion of key-IV pairs produce keystreams of periods shorter than one would be comfortable with. Furthermore, one should keep in mind that we have only given a (rough) lower bound on the probability of short period keystream occurrences. Recent supplementary results [4] on the period of *Edon80*, written by the

designers in response to an earlier version of the current paper, seem to indicate that the actual situation is even worse than what we have pointed out so far.

Even though our results do not give any information on how to recover keys or states, it does show that the period of *Edon80* is far from being well understood. Before *Edon80* can be used in practice, the distribution of keystream periods with respect to randomly chosen key-IV pairs should be fully understood and measures should be taken to prevent use of the shorter ones, if at all possible.

As the designers of *Edon80* put no explicit restriction on the length of keystream usable, and since probabilities for encountering these short keystreams are greater than what is expected from the intended security level, our observation is technically a valid attack on streamcipher *Edon80*.

References

1. ECRYPT, eSTREAM - ECRYPT Stream Cipher Project. Information available from http://www.ecrypt.eu.org/stream/
2. S. Markovski, D. Gligoroski, and L. Kocarev, Unbiased random sequences from quasigroup string transformations. *FSE 2005*, LNCS 3557, pp. 163–180, Springer, 2005.
3. D. Gligoroski, S. Markovski, L. Kocarev, and M. Gušev, *Edon80* - Hardware synchoronous stream cipher. eSTREAM, ECRYPT Stream Cipher Project Report 2005/007, 2005. Presented at Symmetric Key Encryption Workshop, Århus, Denmark, May, 2005. Available from [1].
4. D. Gligoroski, S. Markovski, L. Kocarev, and M. Gušev, Understanding periods in Edon80. eSTREAM, ECRYPT Stream Cipher Project Report 2005/054, 2005. Available from [1].

A 40-row key-state pair

Here is an explicit key-state pair consisting of 40 rows that contains the initial sequence $(0, 1, 2, 3, 0, \ldots)$ at the bottom row. This is not a concatenation of smaller such partial key-state pairs.

	$*_i$		0	1	2	3
0:	$\bullet_1$	2	2	0	0	2
1:	$\bullet_1$	0	0	1	0	0
2:	$\bullet_2$	2	3	3	0	2
3:	$\bullet_0$	0	3	1	2	0
4:	$\bullet_0$	3	1	1	3	3
5:	$\bullet_0$	0	2	3	1	0
6:	$\bullet_0$	3	2	2	3	3
7:	$\bullet_0$	0	1	3	1	0
8:	$\bullet_0$	1	1	0	2	1
9:	$\bullet_0$	0	2	1	3	0
10:	$\bullet_0$	0	1	1	0	0
11:	$\bullet_0$	1	1	1	2	1
12:	$\bullet_1$	3	2	0	0	3
13:	$\bullet_0$	0	1	2	1	0
14:	$\bullet_1$	0	3	1	1	0
15:	$\bullet_1$	1	3	2	0	1
16:	$\bullet_0$	2	2	0	0	2
17:	$\bullet_1$	0	0	1	0	0
18:	$\bullet_2$	2	3	3	0	2
19:	$\bullet_0$	0	3	1	2	0
20:	$\bullet_0$	3	1	1	3	3
21:	$\bullet_0$	0	2	3	1	0
22:	$\bullet_0$	3	2	2	3	3
23:	$\bullet_0$	0	1	3	1	0
24:	$\bullet_0$	2	3	1	1	2
25:	$\bullet_1$	2	1	1	1	2
26:	$\bullet_1$	0	3	2	0	0
27:	$\bullet_1$	2	1	2	2	2
28:	$\bullet_1$	1	1	2	3	1
29:	$\bullet_0$	0	2	0	3	0
30:	$\bullet_3$	1	3	2	1	1
31:	$\bullet_1$	1	3	1	1	1
32:	$\bullet_0$	2	2	3	0	2
33:	$\bullet_0$	2	0	3	3	2
34:	$\bullet_1$	3	3	0	2	3
35:	$\bullet_1$	2	1	0	0	2
36:	$\bullet_3$	0	2	0	3	0
37:	$\bullet_3$	1	3	2	1	1
38:	$\bullet_3$	1	2	2	3	1
39:	$\bullet_3$	3	0	1	2	3