

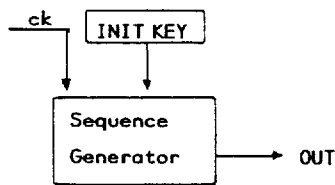
THE STOP-AND-GO-GENERATOR

T. Beth

F.C. Piper

1. Introduction

The usual method for generating binary sequences of acceptable properties with respect to period-complexity and statistics is based on a deterministic finite boolean automaton



which after having been initialized by the key on every clock impulse at time t outputs a bit u_t , $t \in \mathbb{N}_0$.

The cryptographic value of such a sequence generator depends obviously on the complexity of this machine. Several concepts for its design are known.

The best-understood-though not too desirable - finite state machine is a linear feed back shift register (cf. Selmer, Golomb, Jennings).

In most practical applications so-called non-linear feedback machines are being used while their complexity, the so-called *linear equivalent* is described via the shortest linear recursion generating the same output sequence. Another concept of measuring complexity has recently been proposed by Micali et al..

The art of designing finite boolean automata of high complexity has naturally become one of the central topics of modern cryptography - especially in the light of readily accessible VLSI-implementations. Examples of these have been described by Beker/Piper, Jennings, Beth.

A rather new concept of this kind seems to originate from the idea of a variable clock

While the usual concept is based on a clock with timing diagram



diagram 1

a variable clock has a timing diagram like

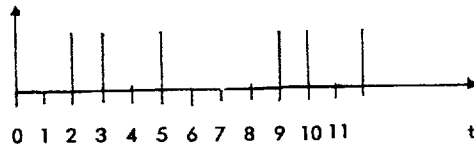


diagram 2

which could be produced from a usual clock (e.g. in diagram 1) AND-gated with a 0-1-sequence (as in diagram 3)

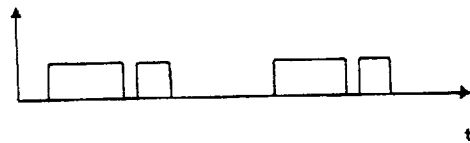
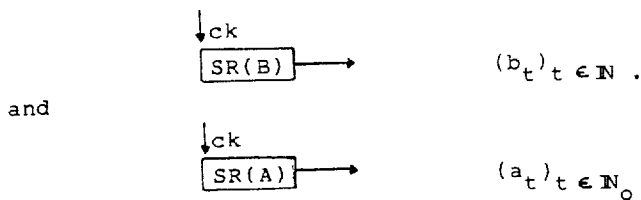


diagram 3

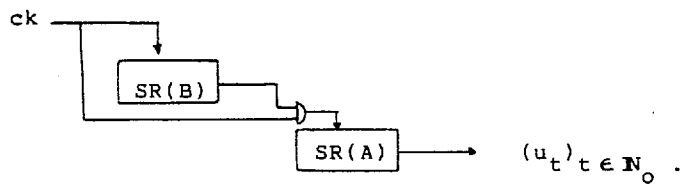
In a research project which has been initiated through a grant by the British Science and Engineering Research Council awarded to the two authors in the year 1983, the theory and realisations of these so-called *Stop-and-Go-Generators* are being investigated.

2. The Stop-and-Go-Generator

The general Stop-and-Go-Generator is built from two feedback shift registers (FSR)



where the outputs of SR(B) are driving the clock of SR(A)



The output-sequence $(u_t)_t$ is the Stop-and-Go-Sequence of $(a_t)_t$ by $(b_t)_t$.

2.1 Observation

$$\text{With } s(t) = \text{wgt}(b_0, \dots, b_t) = \sum_{j=0}^t b_j$$

the bits are given by

$$u_t = a_{s(t)} .$$

From this observation we immediately conclude the following

2.2 Proposition

Let Π_1 resp. Π_2 be the period of the sequence $(a_t)_t$ resp. $(b_t)_t$.

$$w = \text{wgt}(b_1, \dots, b_{\Pi_2})$$

be the number of 1's in the full period of b_t .

If $(w, \Pi_1) = 1$ then the period of $(u_t)_t$ is

$$\Pi = \Pi_1 \cdot \Pi_2$$

The condition of Prop. 2.2 is necessary as the following example shows.

2.3 Example

Let $\underline{a} = (a_0, a_1, a_2, a_0, a_1, a_2, \dots)$ be any sequence of period 3 .

Let \underline{b} the sequence with period (10101) .

Then the Stop-and-Go-Sequence of \underline{a} by \underline{b} is

$$\underline{u} = (a_0 a_0 a_1 a_1 a_2 a_0 a_0 a_1 a_1 a_2 \dots)$$

of period 5 and not 15 as we may expect.
 After determining the period we have to determine the linear equivalent.
 As we know from coding theory weight function wgt is non-trivial analytically, we try another approach to describe the sequence u_t .

2.4 Example:

Obviously the following Boolean equations hold

$$\begin{aligned} u_0 &= a_0 && \text{(by definition)} \\ u_1 &= b_1 a_1 + (1-b_1) a_0 \\ u_2 &= b_2 b_1 a_2 + (b_1(1-b_2) + b_2(1-b_1)) a_1 \\ &\quad + (1-b_1)(1-b_2) a_0 \end{aligned}$$

In general we have

2.5 Lemma

For $n \in \mathbb{N}_0$ $u_n \in \text{BoolPol}[b_1, \dots, b_n; a_0, \dots, a_n]$

is a Boolean Polynomial in

b_1, \dots, b_n and a_0, \dots, a_n with degree $\delta_{\underline{b}}(u_n) = n$
 wrt. b_1, \dots, b_n .

From this we derive

2.6 Lemma

Let R_n denote the ring of Boolean polynomials

$$R_n = \text{BoolPol}[a_0, \dots, a_n].$$

Suppose the linear equivalent of $(b_t)_t$ is $L(B)$. Then u_n is the R -linear combination of all $2^{L(B)-1}$ monomials in b_0, \dots, b_{L-1} .

With some special assumptions from this we can for instance derive the

2.7 Theorem

If $(a_t)_t$ and $(b_t)_t$ are binary sequences which belong to linearly disjoint field extensions then $(u_t)_t$ has the linear equivalent

$$L(U) = (2^{L(B)} - 1) L(A)$$

Of course the situation assumed in the theorem is the "nicest" general case. Other special cases are studied by Vogel, who considers the case of equal field extensions (cf. Vogel) and by Gollmann, who investigates cascaded shift registers of *equal prime period*.

3. Concluding remarks

Under the correct assumptions cascading of primitive shift registers leads to interesting results. But from Gollmann's work it is clear that general results on cascaded arbitrary shift registers cannot be expected.

In order to guarantee a good statistical behaviour of the Stop-and-Go-Sequence it is suggested that the output sequence u_t is finally XOR-gated with another PN-sequence.

The statistical behaviour of $(u_t)_t$ itself - though theoretically quite good in special cases - is so that a cryptoanalytic attack would be promising in spite of the extremely high linear equivalent of the sequence.

4. References

- Beker/Piper: Cipher Systems, Northwood 1982
 Beth: Stream Ciphers, in: Secure Digital Communications, G. Longo ed., Springer 1983
 Gollmann: Doctoral Dissertation, University of Linz, Austria 1983
 Golomb: Shift register sequences, Holden-Day 1967
 Jennings: Multiplexed Sequences, in: Cryptography, T. Beth ed., Springer LNCS 149, 1983
 Selmer: Linear Recurrence Relations over Finite Fields, manuscript. Dept. of Math., University of Bergen, Norway 1960
 Vogel: On the linear complexity of cascaded sequences, preprint, SEL Pforzheim, CP/ERMF, Germany 1984