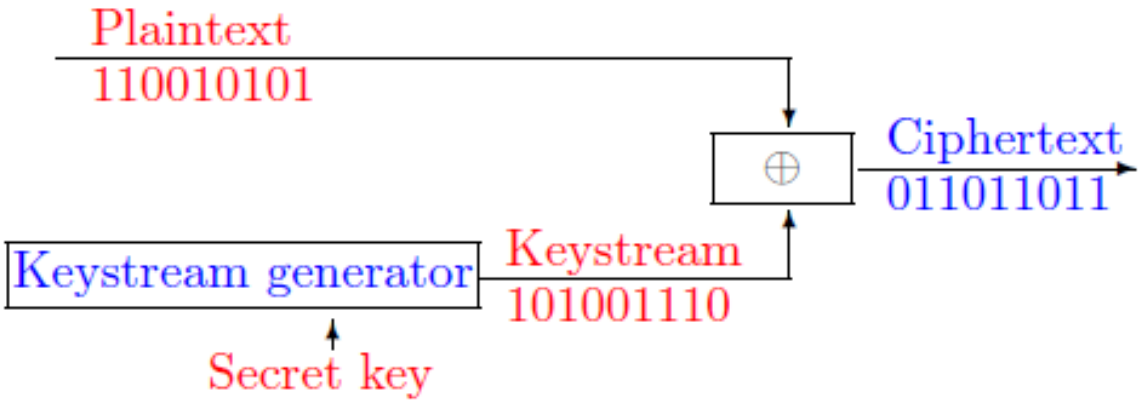
STREAM CIPHER SYSTEMS



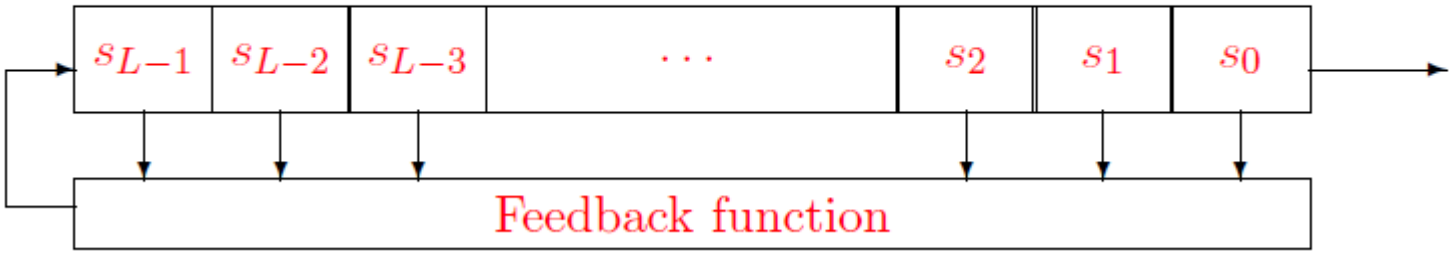
Thus we have $c_i = m_i \oplus k_i$ where

- $m_0, m_1, \ldots$ are the plaintext bits,
- $k_0, k_1, \ldots$ are the keystream bits,
- $c_0, c_1, \ldots$ are the ciphertext bits.

This means

$$m_i = c_i \oplus k_i$$

FIGURE 1. Feedback shift register



$$s_{L-1}(t+1) = f(s_0(t), s_1(t), \cdots, s_{L-1}(t))$$

The output is periodic

Linear feedback shift registers

Linear feedback shift registers (LFSRs) are used in many of the keystream generators that have been proposed in the literature. There are several reasons for this:

1. LFSRs are well-suited to hardware implementation;
2. they can produce sequences of large period (Fact 6.12);
3. they can produce sequences with good statistical properties (Fact 6.14); and
4. because of their structure, they can be readily analyzed using algebraic techniques.

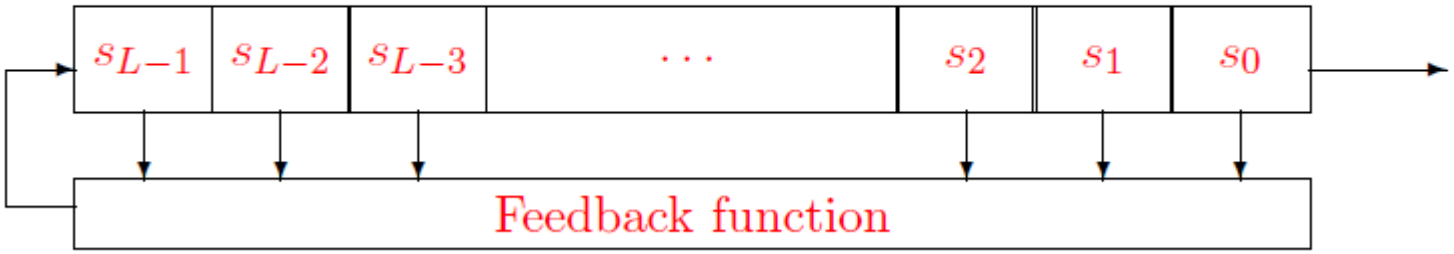
Definition A *linear feedback shift register* (LFSR) of length L consists of L *stages* (or *delay elements*) numbered $0, 1, \ldots, L-1$, each capable of storing one bit and having one input and one output; and a clock which controls the movement of data. During each unit of time the following operations are performed:

(i) the content of stage 0 is output and forms part of the *output sequence*;

(ii) the content of stage i is moved to stage $i-1$ for each i, $1 \leq i \leq L-1$; and

(iii) the new content of stage $L-1$ is the *feedback bit* s_j which is calculated by adding together modulo 2 the previous contents of a fixed subset of stages $0, 1, \ldots, L-1$.

FIGURE 1. Feedback shift register



Linear Feedback Shift Register(*LFSR*)

$$s_{L-1}(t+1) = \sum_{i=1}^{L} c_i s_{L-i}(t) \qquad s_{L+t} = \sum_{i=1}^{L} c_i s_{L-i+t} \quad t = 0,1,2,\cdots$$

If we write

$$M = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ c_L & c_{L-1} & c_{L-2} & \dots & c_1 \end{pmatrix}$$

$$\underline{S}(t+1) = M\,\underline{S}(t)$$

$$\underline{S}(t) = \begin{bmatrix} s_0(t) \\ s_1(t) \\ \vdots \\ s_{L-1}(t) \end{bmatrix}$$

$$C(X) = C_L + C_{L-1}X + \cdots + C_1 X^{L-1} + C_0 X^L$$

$$C(X) = 1 + C_{L-1}X + \cdots + C_1 X^{L-1} + X^L$$

$$LFSR \quad is \quad denoted:$$

$$< L, C(x) > \quad or \quad < L, C(D) >$$

In some text books the connection polynomial is written in reverse, i.e. they use

$$G(X) = X^L C(1/X)$$

$$C(X) = 1 + c_1 X + c_2 X^2 + \cdots + c_L X^L \in \mathbb{F}_2[X],$$

FIGURE 2. Linear feedback shift register: $X^3 + X + 1$

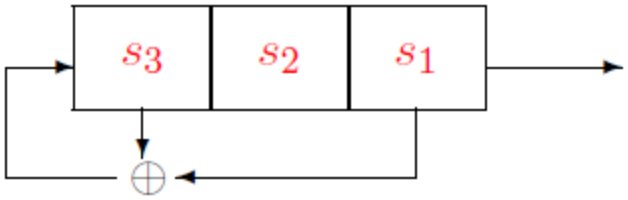


FIGURE 3. Linear feedback shift register: $X^{32} + X^3 + 1$

Fact Every output sequence (i.e., for all possible initial states) of an LFSR $\langle L, C(D)\rangle$ is periodic if and only if the connection polynomial $C(D)$ has degree L.

If an LFSR $\langle L, C(D)\rangle$ is *singular* (i.e., $C(D)$ has degree less than L), then not all output sequences are periodic. However, the output sequences are *ultimately periodic*; that is, the sequences obtained by ignoring a certain finite number of terms at the beginning are periodic. For the remainder of this chapter, it will be assumed that all LFSRs are non-singular. Fact 6.12 determines the periods of the output sequences of some special types of non-singular LFSRs.

$We\,call\ N\ \exp onent\ if\ N\ is\,the\,smallest\,value\,N$

$such\ that\ \ C(x)/x^{N}+1$

$1-\ N\le 2^{L}-1$

$2-\ P/N \qquad (P=\ period\ of\ \ the\ output)$

$3-If\ \ c(x)\,is\,an\,irreducible\ \ polynomial\ then\ \ P=N$

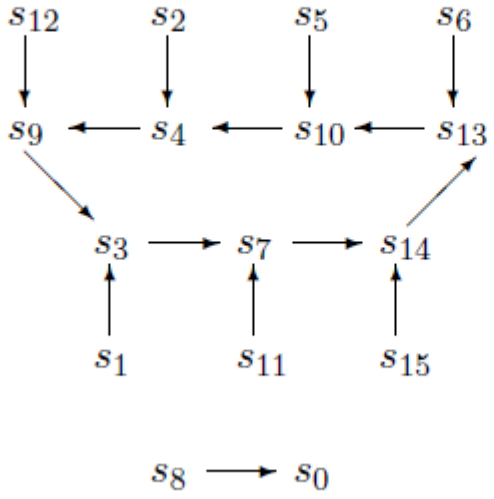
$4-If\ \ c(x)\,is\,a\ primitive\ \ polynomial\ then\ \ P=2^{L}-1$

Example 1 : In this example we use an LFSR with connection polynomial $C(X) = X^3 + X + 1$. We therefore see that $\deg(C) \neq L$, and so the sequence will be singular. The matrix M generating the sequence is given by

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

$s_0 = (0, 0, 0, 0)$ and $s_5 = (0, 1, 0, 1)$

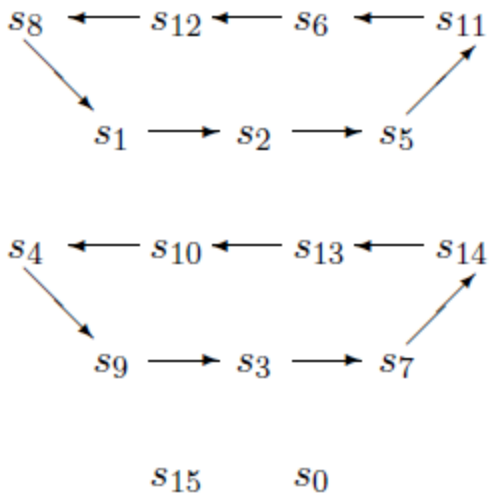
FIGURE 4. Transitions of the four bit LFSR with connection polynomial $X^3 + X + 1$



Example 2 : Now let the connection polynomial $C(X) = X^4+X^3+X^2+1 = (X+1)(X^3+X+1)$, which corresponds to the matrix

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

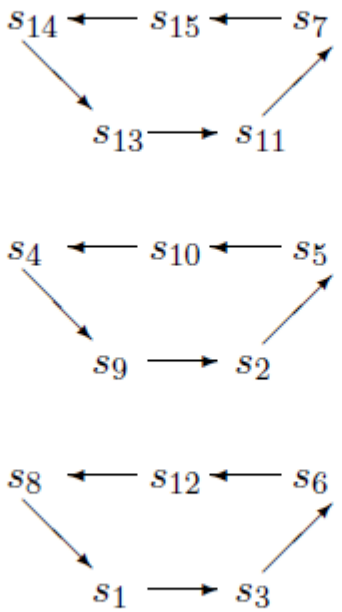
FIGURE 5. Transitions of the four bit LFSR with connection polynomial $X^4+X^3+X^2+1$



Example 3 : Now take the connection polynomial $C(X) = X^4 + X^3 + X^2 + X + 1$, which is irreducible, but not primitive. The matrix is now given by

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

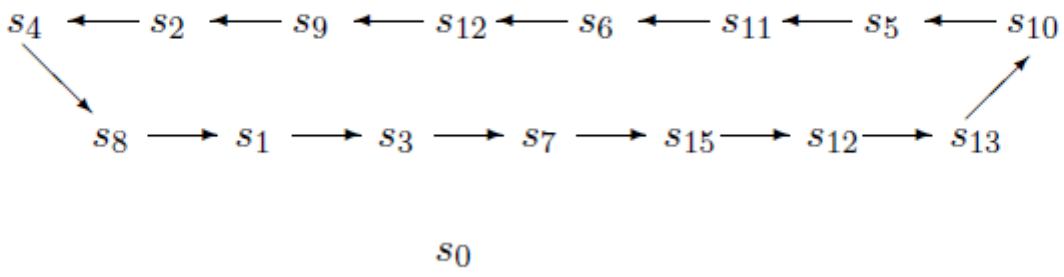
FIGURE 6. Transitions of the four bit LFSR with connection polynomial $X^4 + X^3 + X^2 + X + 1$



Example 4 : As our final example we take the connection polynomial $C(X) = X^4 + X + 1$, which is irreducible and primitive. The matrix M is now

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

FIGURE 7. Transitions of the four bit LFSR with connection polynomial $X^4 + X + 1$



Whilst there are algorithms to generate primitive polynomials for use in applications we shall not describe them here. We give some samples in the following list, where we give polynomials with a small number of taps for efficiency.

$$
\begin{array}{lll}
x^{31}+x^{3}+1 & x^{31}+x^{6}+1 & x^{31}+x^{7}+1 \\
x^{39}+x^{4}+1 & x^{60}+x+1 & x^{63}+x+1 \\
x^{71}+x^{6}+1 & x^{93}+x^{2}+1 & x^{137}+x^{21}+1 \\
x^{145}+x^{52}+1 & x^{161}+x^{18}+1 & x^{521}+x^{32}+1
\end{array}
$$

We shall now show that if we know an LFSR has L internal registers, and we can determine $2L$ consecutive bits of the stream then we can determine the whole stream. First notice we need to determine L unknowns, the L values of the 'taps' c_i, since the L values of the initial state $s_0, \ldots, s_{L-1}$ are given to us. This type of data could be available in a known plaintext attack, where we obtain the ciphertext corresponding to a known piece of plaintext, since the encryption operation is simply exclusive-or we can determine as many bits of the keystream as we require.

Using the equation

$$s_j = \sum_{i=1}^{L} c_i \cdot s_{j-i} \pmod 2,$$

we obtain $2L$ linear equations, which we then solve via matrix techniques. We write our matrix equation as

$$\begin{pmatrix} s_{L-1} & s_{L-2} & \cdots & s_1 & s_0 \\ s_L & s_{L-1} & \cdots & s_2 & s_1 \\ \vdots & \vdots & & \vdots & \vdots \\ s_{2L-3} & s_{2L-4} & \cdots & s_{L-1} & s_{L-2} \\ s_{2L-2} & s_{2L-3} & \cdots & s_L & s_{L-1} \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{L-1} \\ c_L \end{pmatrix} = \begin{pmatrix} s_L \\ s_{L+1} \\ \vdots \\ s_{2L-2} \\ s_{2L-1} \end{pmatrix}.$$

Stream ciphers based on LFSRs

Note (*use of LFSRs in keystream generators*) Since a well-designed system should be secure against known-plaintext attacks, an LFSR should never be used by itself as a keystream generator. Nevertheless, LFSRs are desirable because of their very low implementation costs. Three general methodologies for destroying the linearity properties of LFSRs are discussed in this section:

(i) using a nonlinear combining function on the outputs of several LFSRs (§6.3.1);
(ii) using a nonlinear filtering function on the contents of a single LFSR (§6.3.2); and
(iii) using the output of one (or more) LFSRs to control the clock of one (or more) other LFSRs (§6.3.3).

Desirable properties of LFSR-based keystream generators

For essentially all possible secret keys, the output sequence of an LFSR-based keystream generator should have the following properties:

1. large period;
2. large linear complexity; and
3. good statistical properties (e.g., as described in Fact 6.14).

It is emphasized that these properties are only *necessary* conditions for a keystream generator to be considered cryptographically secure. Since mathematical proofs of security of such generators are not known, such generators can only be deemed *computationally secure* (§1.13.3(iv)) after having withstood sufficient public scrutiny.

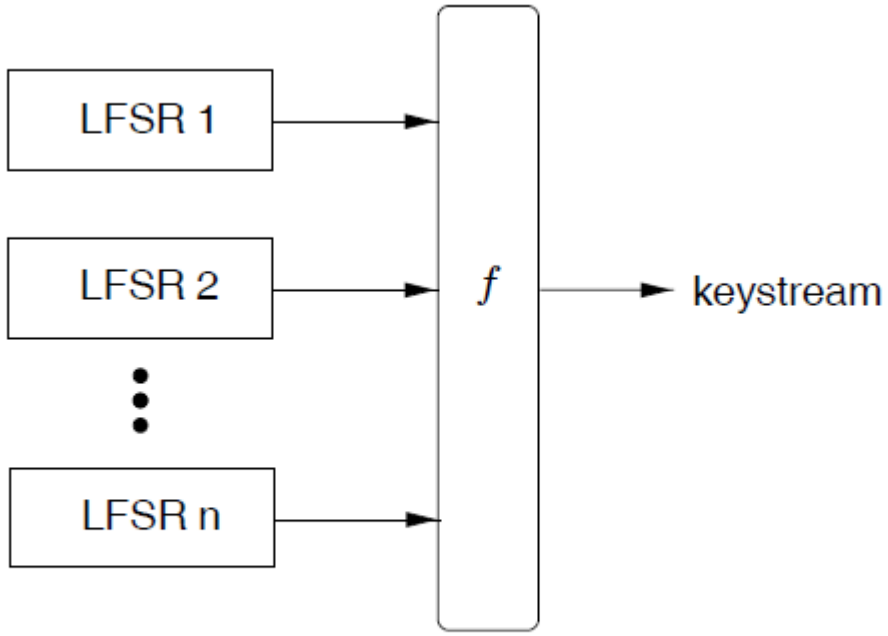
Desirable properties of LFSR-based keystream generators

For essentially all possible secret keys, the output sequence of an LFSR-based keystream generator should have the following properties:

1. large period;
2. large linear complexity; and
3. good statistical properties (e.g., as described in Fact 6.14).

It is emphasized that these properties are only *necessary* conditions for a keystream generator to be considered cryptographically secure. Since mathematical proofs of security of such generators are not known, such generators can only be deemed *computationally secure* (§1.13.3(iv)) after having withstood sufficient public scrutiny.

Nonlinear combination generators



A nonlinear combination generator. f is a nonlinear combining function.

Example (*Geffe generator*) The Geffe generator, as depicted in Figure 6.9, is defined by three maximum-length LFSRs whose lengths L_1, L_2, L_3 are pairwise relatively prime, with nonlinear combining function

$$f(x_1, x_2, x_3) \;=\; x_1x_2 \oplus (1 + x_2)x_3 \;=\; x_1x_2 \oplus x_2x_3 \oplus x_3.$$

The keystream generated has period $(2^{L_1} - 1) \cdot (2^{L_2} - 1) \cdot (2^{L_3} - 1)$ and linear complexity $L = L_1L_2 + L_2L_3 + L_3$.

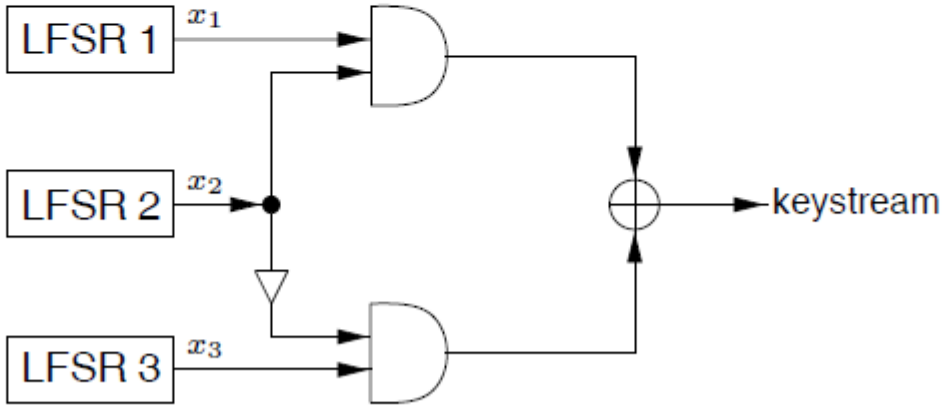


Figure 6.9: *The Geffe generator.*

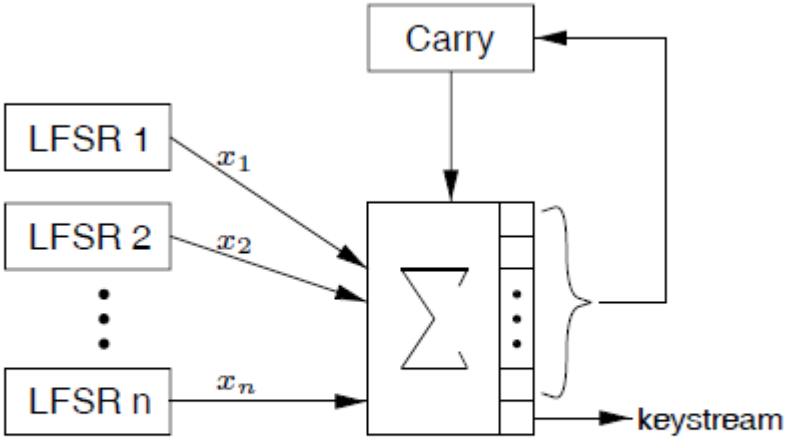
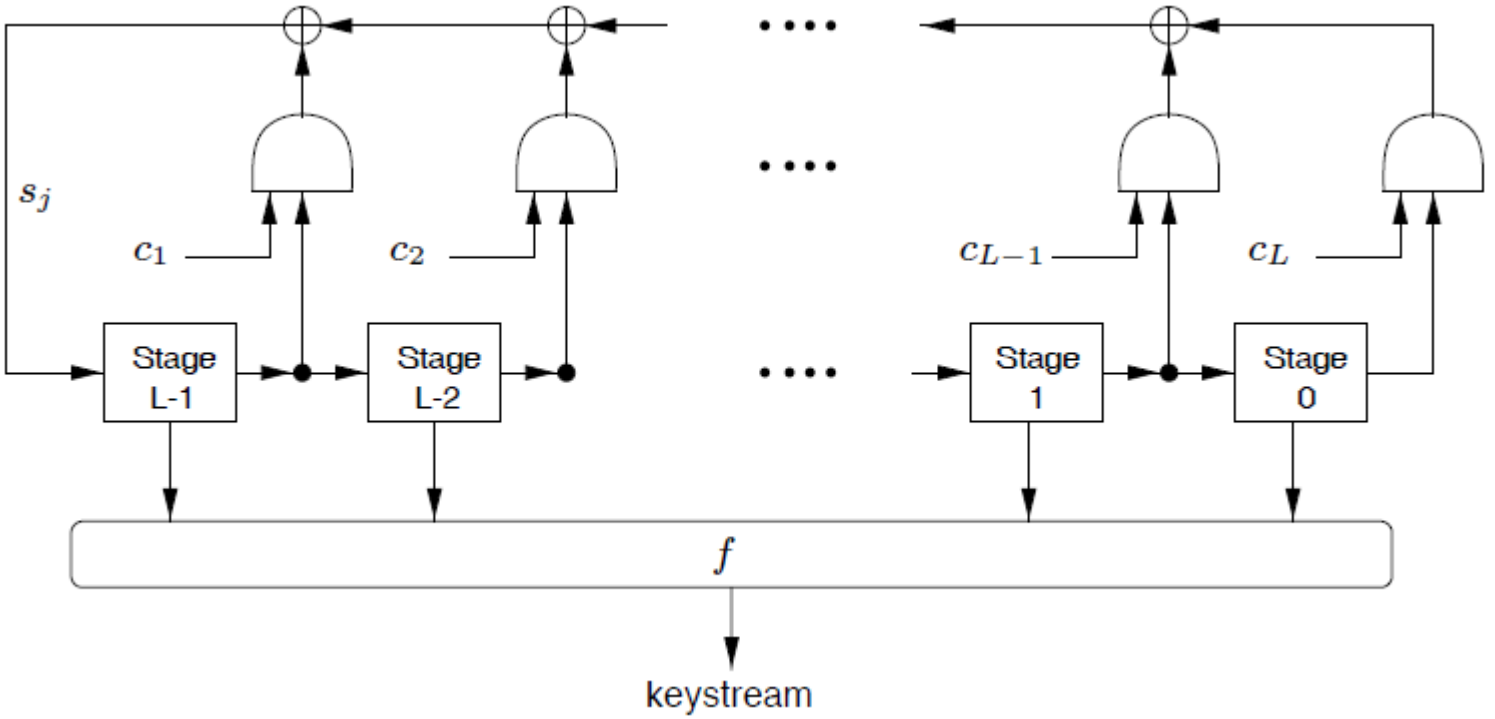


Figure 6.10: *The summation generator.*

The summation generator, as depicted in Figure 6.10, is defined by n maximum-length LFSRs whose lengths $L_1, L_2, \ldots, L_n$ are pairwise relatively prime. The secret key consists of the initial states of the LFSRs, and an initial (integer) carry C_0. The keystream is generated as follows. At time j ($j \geq 1$), the LFSRs are stepped producing output bits $x_1, x_2, \ldots, x_n$, and the *integer* sum $S_j = \sum_{i=1}^{n} x_i + C_{j-1}$ is computed. The keystream bit is $S_j \bmod 2$ (the least significant bit of S_j), while the new carry is computed as $C_j = \lfloor S_j/2 \rfloor$ (the remaining bits of S_j). The period of the keystream is $\prod_{i=1}^{n}(2^{L_i} - 1)$, while its linear complexity is close to this number.

Nonlinear filter generators



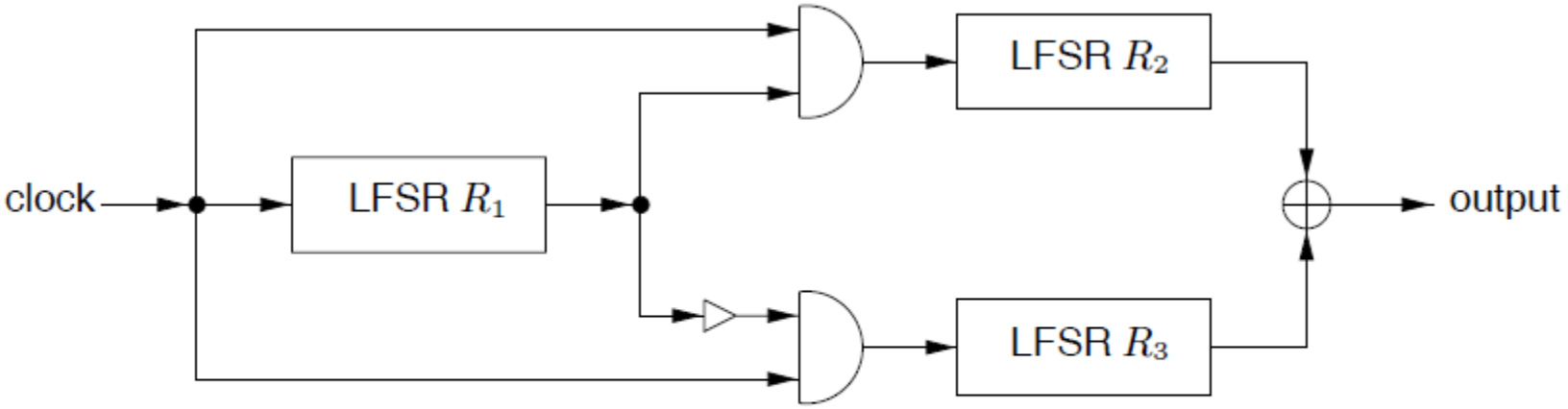


Figure 6.12: *The alternating step generator.*

(ii) The shrinking generator

Algorithm Shrinking generator

SUMMARY: a control LFSR R_1 is used to control the output of a second LFSR R_2. The following steps are repeated until a keystream of desired length is produced.

1. Registers R_1 and R_2 are clocked.
2. If the output of R_1 is 1, the output bit of R_2 forms part of the keystream.
3. If the output of R_1 is 0, the output bit of R_2 is discarded.

More formally, let the output sequences of LFSRs R_1 and R_2 be $a_0, a_1, a_2, \ldots$ and $b_0, b_1, b_2, \ldots$, respectively. Then the keystream produced by the shrinking generator is $x_0, x_1, x_2, \ldots$, where $x_j = b_{i_j}$, and, for each $j \geq 0$, i_j is the position of the j^{th} 1 in the sequence $a_0, a_1, a_2, \ldots$.

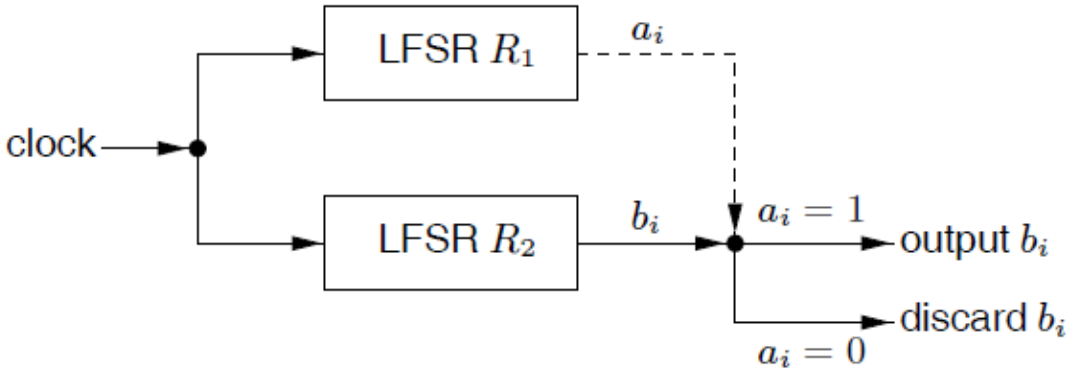


Figure 6.13: *The shrinking generator.*