جامـــــعة المـــــستـقبـل
AL MUSTAQBAL UNIVERSITY

College of science
Department of Cyber Security

Lecture 5:

Nonlinear Shift Register
Nonlinear combination generators

Stage :2

م.م مصطفى امير صبري

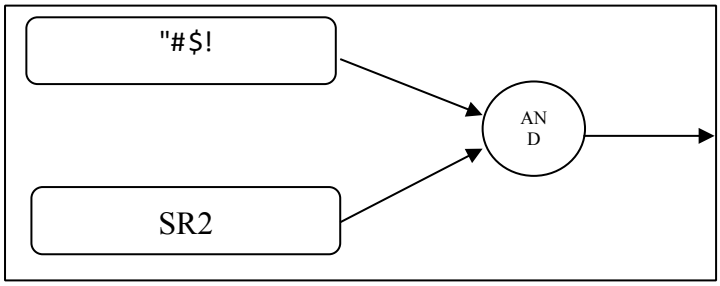
6.1 Nonlinear Shfit Register

LFSR are not always useful ,since they are not resistence to a given know-plaintext attack.as an alternative ,nonlinear feedback shfit register with nonlinear feed back funtion are often used.

Linear feedback shfit register are unsafe because they have relative small linear complxity ,and hence a relatively small fragment of the key stream(LFSR sequnce) can be used to obtain the entire sequnce by solving as set of linear equation .to increase the linear complxity of LFSR ,one or more output sequnce of LFSR's are combined with some nonlinear function to produce relative higher linear complxity ,for example shfit register ***SR1*** genertates sequnce(***S1***) with sequnce length of (2^n-1),and shfit register ***SR2*** generates sequnce (***S2***) with sequnce length (2^m-1),then output sequnce

(**S3**) will be :

S3=S1*S2 with period (sequnce length)= $(2^n-1)*(2^m-1)$



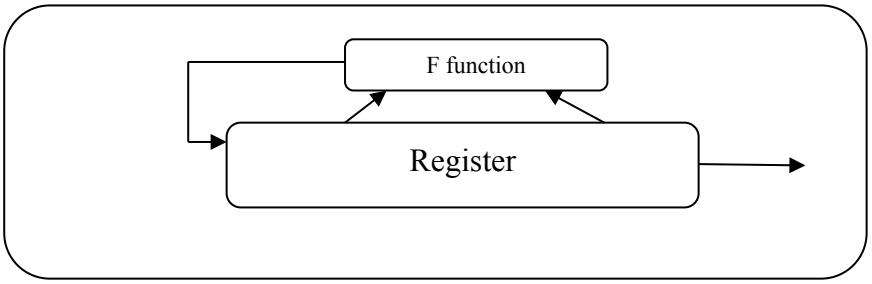
- **Nonlinear Feedback Shift Register**

In which the key stream generator is a shift register with non-linear feedback function .as illustrated in following figure.

In this type one LFSR is used with n-stages and non-linear feedback function. The simplest nonlinear function is **"AND"** functions, for example:

$F=1+X_1X_2+X_2X_3+X_2X_3X_4$

Where X1X2 are (**X_1 and X_2**)



- **Non Linearity Filtered LFSR System:**

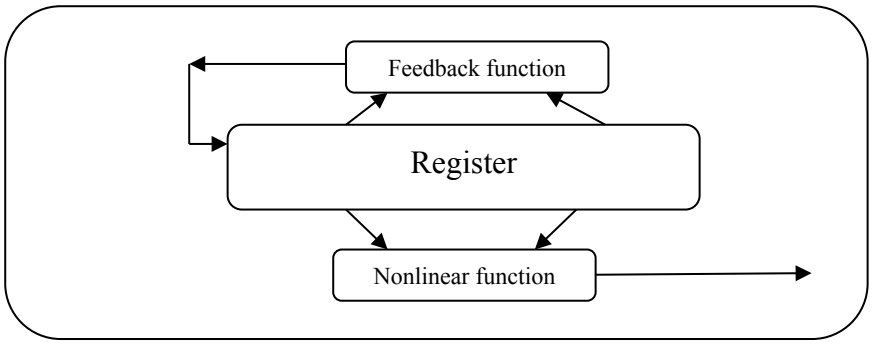
In which a nonlinear logical function is applied to the contented of the LFSR. **Gorth** generator is an example of this type ,as illustrated in following figure **Gorth** sequence consists of:

- Linear feedback function given by:

F1=$S_1+S_2+S_3$ ❖ Non

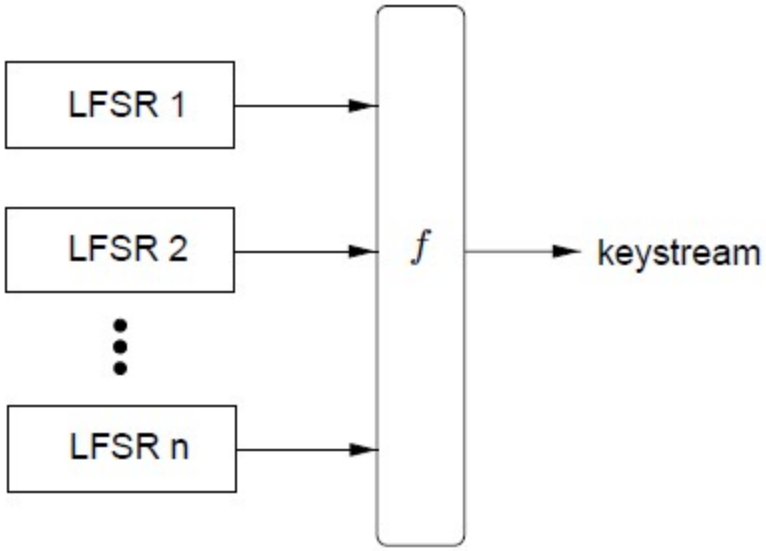
linear filter given by:

F2=$S_0S3+S_1S_5+S_2S_4$



6.2 Nonlinear combination generators

One general technique for destroying the linearity inherent in LFSRs is to use several LFSRs in parallel. The key stream is generated as a nonlinear function **f** of the outputs of the component LFSRs; this construction is illustrated in the following Figure. Such key stream generators are called ***nonlinear combination*** *generators*, and **f** is called the *combining function*. The remainder of this subsection demonstrates that the function **f** must satisfy several criteria in order to withstand certain particular cryptographic attacks.



A product of m distinct variables is called an m_{th} order product of the variables. Every Boolean function f(x1, x2,……, xn) can be written as a modulo 2 sum of distinct m_{th} order products of its variables, 0 <= m <= n; this expression is called the algebraic normal form of ***f***. The nonlinear order of ***f*** is the maximum of the order of the terms appearing in its algebraic normal form.

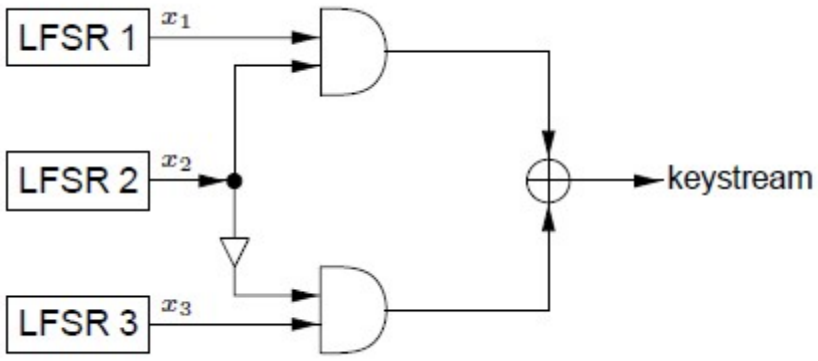
For example, the Boolean function **f(x1, x2, x3, x4, x5)** = X1 **Xor** x2 **Xor** x3 **Xor** x4x5 **Xor** x1x3x4x5 has nonlinear order **4**. Note that the maximum possible nonlinear order of a Boolean function in n variables is n. demonstrates that the output sequence of a nonlinear combination generator has high linear complexity, provided that a combining function f of high nonlinear order is employed.

- (***Geffe generator***) The Geffe generator, as depicted in following Figure. is defined by three maximum-length LFSRs whose lengths ***L1, L2, L3*** are pairwise relatively prime, with nonlinear combining function

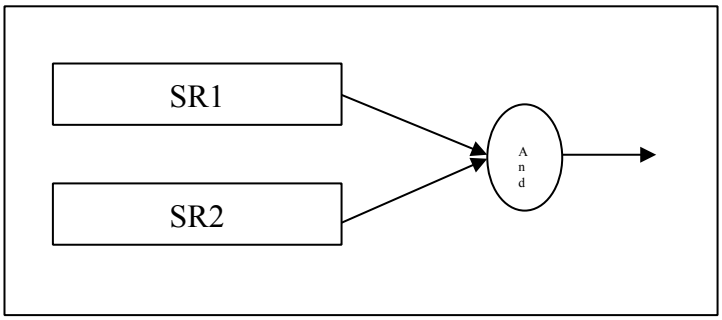
  $F(X_1, X_2, X_3)=(X_1$ and $X_2)$ Xor (not $(X_2)$and $X_3)$

  And the maximal length of Geffe is

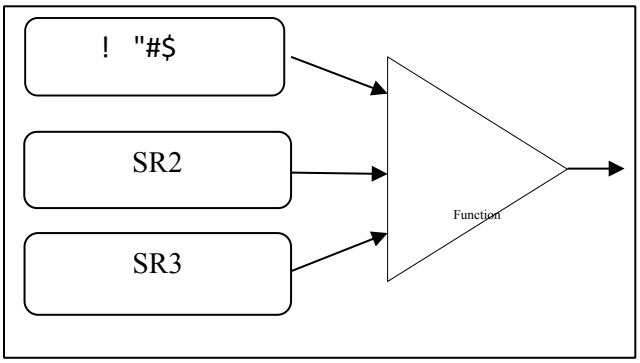
  Max period = $(2^{L1}-1)^* (2^{L2}-1)^* (2^{L3}-1)$



- **(Hardmard Generator)** :Nonlinear generator consists of two LFSR of(**X1**,**X2**) with nonlinear function
  F(x) = (S1 and S2)
  And the maximal
  Max period =$(2^{L1}-1)*(2^{L2}-1)$



- **(Threshold Generator)**: nonlinear generator consist of three LFSR with nonlinear function
  F(x) = (S1 and S2) Xor (S1 and S3) Xor (S2 and S3)
  And the maximal
  Max period = $(2^{L1}-1)*(2^{L2}-1)*(2^{L3}-1)$



LFSR/FCSR Summation/Parity Cascade

The theory is that addition with carry destroys the algebraic properties of LFSRs, and that XOR destroys the algebraic properties of FCSRs. This generator combines those ideas, as used in the LFSR/FCSR Summation Generator and the LFSR/FCSR Parity Generator just listed, with the Gollmann cascade.

The generator is a series of arrays of registers, with the clock of each array controlled by the output of the previous array. In the following figure is one stage of this generator. The first array of LFSRs is clocked and the results are combined using addition with carry. If the output of this combining function is 1, then the next array (of FCSRs) is clocked and the output of those FCSRs is combined with the output of the previous combining function using XOR. If the output of the first combining function is 0, then the array of FCSRs is not clocked and the output is simply added to the carry from the previous round. If the output of this second combining function is 1, then the third array of LFSRs is clocked, and so on.

This generator uses a lot of registers: *n*m*, where *n* is the number of stages and *m* is the number of registers per stage. recommend ***n* = 10 and *m* = 5.**

Example:

- ***Algorithm Alternating step generator***

SUMMARY: a control LFSR R1 is used to selectively step two other LFSRs, R2 and R3.

OUTPUT: a sequence which is the bitwise XOR of the output sequences of R2 and R3.

The following steps are repeated until a keystream of desired length is produced.

1. Register R1 is clocked.

2. If the output of R1 is 1 then:

- *R2 is clocked; R3 is not clocked but its previous output bit is repeated.*

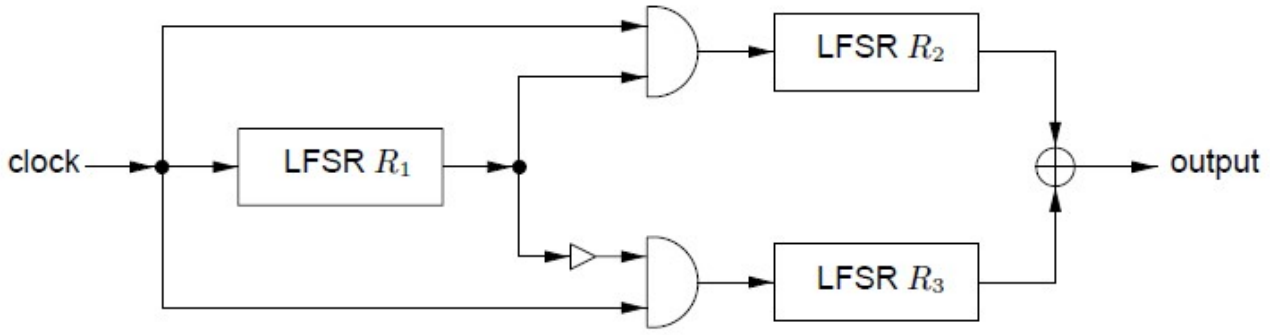
(For the first clock cycle, the "previous output bit" of R3 is taken to be 0.)

3. If the output of R1 is 0 then:

- *R3 is clocked; R2 is not clocked but its previous output bit is repeated.*

(For the first clock cycle, the "previous output bit" of R2 is taken to be 0.)

4. The output bits of R2 and R3 are XORed; the resulting bit is part of the key stream

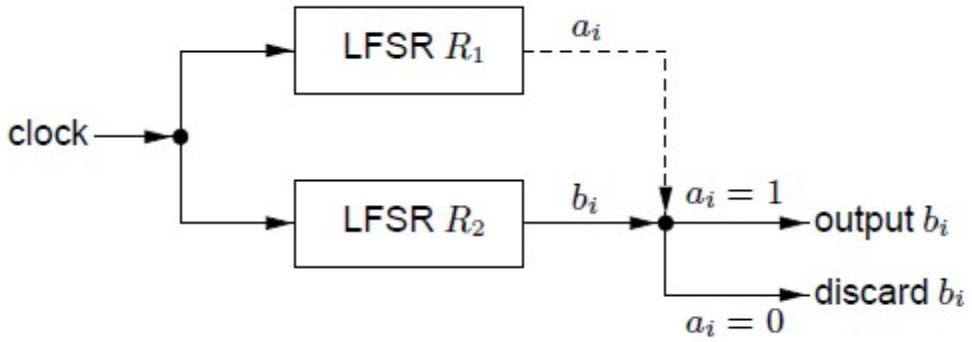


❖ *Shrinking Generators*

SUMMARY: a control LFSR R1 is used to control the output of a second LFSR R2.

The following steps are repeated until a keystream of desired length is produced.

1. Registers R1 and R2 are clocked.
2. If the output of R1 is 1, the output bit of R2 forms part of the keystream.
3. If the output of R1 is 0, the output bit of R2 is discarded..


LFSR R_1
a_i
clock
LFSR R_2
b_i
$a_i = 1$
output b_i
discard b_i
$a_i = 0$