جامعة المستقبل
AL MUSTAQBAL UNIVERSITY

College of science
Department of Cyber Security

Lecture 4:

Introduction SR
key stream generator
Feedback Function
TAP Sequence
Maximal Length Tap Sequences
Linear Equivalence

Stage :2

م.م مصطفى امير صبري

5.1 Introduction SR

In digital circuits, a shift register is a cascade of flip flops, sharing the same clock, which has the output of anyone but the last flip-flop connected to the "data" input of the next one in the chain, resulting in a circuit that shifts by one position the **one-dimensional** "bit array" stored in it, shifting in the data present at its input and shifting out the last bit in the array, when enabled to do so by a transition of the clock input. More generally, a shift register may be **multidimensional**; such that its "data in" input and stage outputs are themselves bit arrays: this is implemented simply by running several shift registers of the same bit-length in parallel.

Shift registers can have both **parallel** and **serial** inputs and outputs. These are often configured as serial-in, parallel-out (**SIPO**) or as parallel-in, serialout (**PISO**). There are also types that have both serial and parallel input and types with serial and parallel output. There are also bi-directional shift registers which allow shifting in both directions: L→R or R→L. The serial input and last output of a shift register can also be connected together to create a circular shift register.

5.2 key stream generator

The basic element in **stream ciphers** is the key stream generators, which will generator the key stream (sequence) to be combining with plain text stream and produce the cipher text and often called *feedback shift register* (**FSR**).

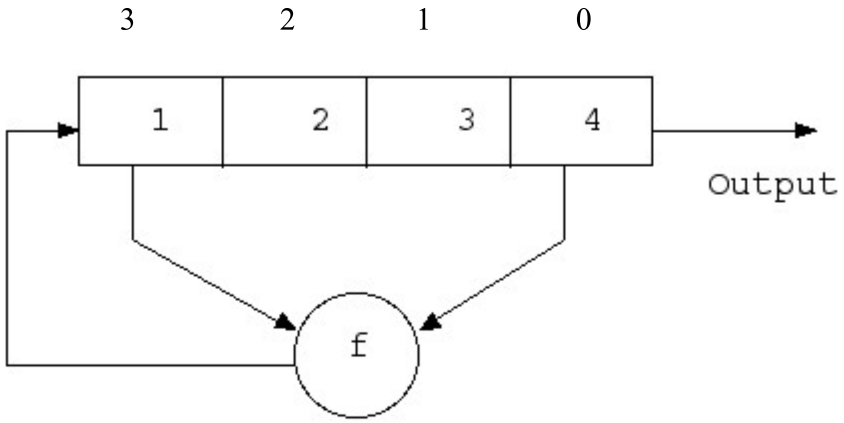
The shift register controlled by a block input. On every plus of the block the bits are shift one stage to right the bits that are generated at stage 0 from the output of the shift register .As the shift to the right, it is necessary to supply new bits to stage m-1, these bits can be obtained from feedback loop, according to a feedback function based on the values, which the feedback from the outputs of the ether stages.

For any m-stage register with feedback constants $c_{0,}c_1,c_2,\ldots..,c_{m-1}$,the characteristic polynomial f(x) is define

$F(x)=c_0x^0+c_1x^1+\ldots..+x^{m-1}$

$F(x)=1+ c_1x^1+\ldots..+x^{m-1}$

The period of the sequence generator by the circit depend on whether polynomial E(x) is primitive and irreducible .maximal length sequence with period (2_m-1) are generator only if the case when the characteristic generating polynomial is primate and irreducible.

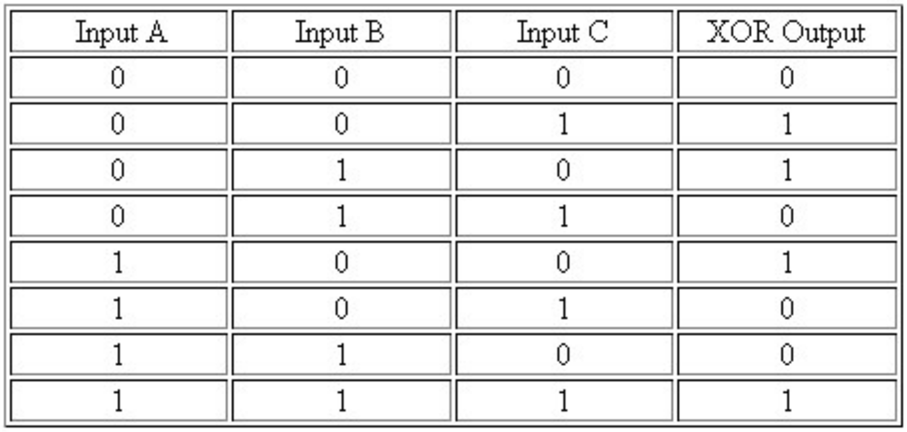


- A linear feedback shift register (**LFSR**) is a shift register whose input bit is a linear function of its previous state.

5.3 Feedback Function

In an LFSR, the bits contained in selected positions in the shift register are combined in some sort of function and the result is feed back into the register's input bit. By definition, the selected bit values are collected before the register is clocked and the result of the feedback function is inserted into the shift register during the shift, filling the position that is emptied as a result of the shift.

The feedback function in an LFSR has several names: **XOR**, **odd parity, sum modulo 2**. Whatever the name, the function is simple: 1) Add the selected bit values, 2) If the sum is odd, the output of the function is one; otherwise the output is zero. In following Table shows the output for a 3 input XOR function.



Input A	Input B	Input C	XOR Output
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

5.4 TAP Sequence

An LFSR is one of a class of devices known as state machines. The contents of the register, the bits tapped for the feedback function, and the output of the feedback function together describe the state of the LFSR. With each shift, the LFSR moves to a new state. (**There is one exception to this - when the contents of the register are all zeroes, the LFSR will never change state.**) For any given state, there can be only one succeeding state. The reverse is also true: any given state can have only one preceding state. For the rest of this discussion, only the contents of the register will be used to describe the state of the LFSR.

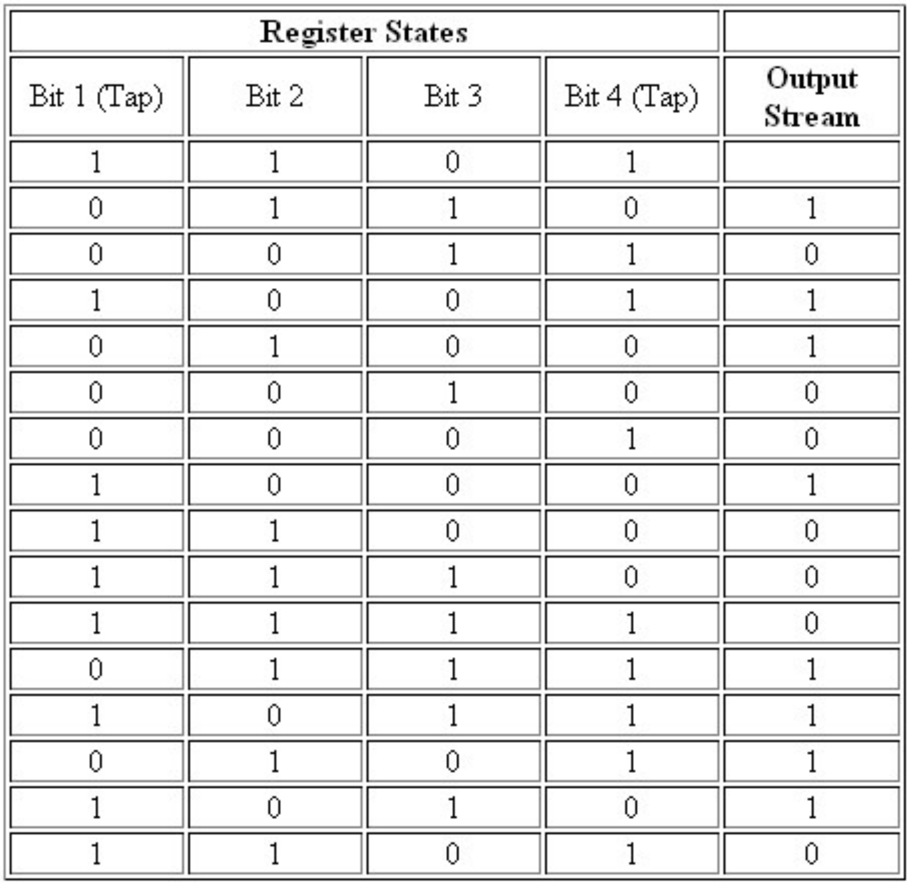
A state space of an **LFSR** is the list of all the states the LFSR can be in for a particular tap sequence and a particular starting value. Any tap sequence will yield at least two state spaces for an LFSR. (One of these spaces will be the one that contains only one state -- the all zero one.)

Tap sequences that yield only two state spaces are referred to as maximal length tap sequences.

The state of an LFSR that is n bits long can be any one of 2_n different values. The largest state space possible for such an LFSR will be **2_n - 1** (all possible values minus the zero state). Because each state can have only once succeeding state, an LFSR with a maximal length tap sequence will pass through every non-zero state once and only once before repeating a state.

One corollary to this behavior is the output bit stream. The period of an LFSR is defined as the length of the stream before it repeats. The period, like the state space, is tied to the tap sequence and the starting value. As a matter of fact, the period is equal to the size of the state space. The longest period possible corresponds to the largest possible state space, which is produced by a maximal length tap sequence. (Hence "maximal length")

In the table is a listing of the internal states and the output bit stream of a 4-bit LFSR with tap sequence [4, 1]. (This is the LFSR shown first figure.)



Register States				
Bit 1 (Tap)	Bit 2	Bit 3	Bit 4 (Tap)	Output Stream
1	1	0	1	
0	1	1	0	1
0	0	1	1	0
1	0	0	1	1
0	1	0	0	1
0	0	1	0	0
0	0	0	1	0
1	0	0	0	1
1	1	0	0	0
1	1	1	0	0
1	1	1	1	0
0	1	1	1	1
1	0	1	1	1
0	1	0	1	1
1	0	1	0	1
1	1	0	1	0

5.5 Maximal Length Tap Sequences

LFSR's can have multiple maximal length tap sequences. A maximal length tap sequence also describes the exponents in what is known as a primitive polynomial mod 2. For example, a **tap** sequence of 4, 1 describes the primitive polynomial **$x_4 + x_1 + 1$**. Finding a primitive polynomial mod 2 of degree n (the largest exponent in the polynomial) will yield a maximal length tap sequence for an LFSR that is n bits long.

There is no quick way to determine if a tap sequence is maximal length. However, there are some ways to tell if one is not maximal length:

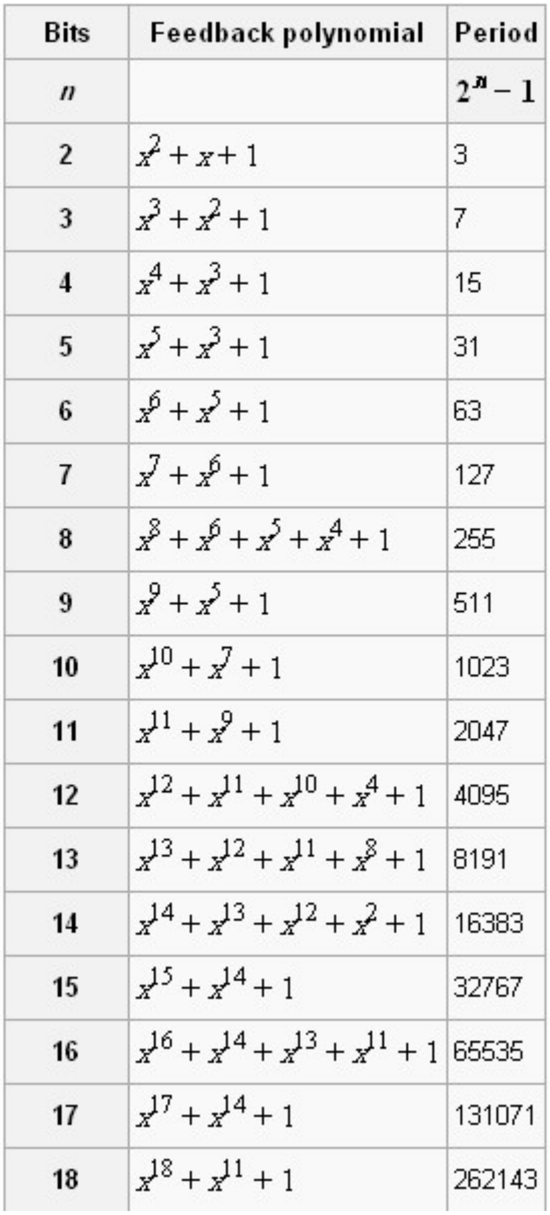
1) Maximal length tap sequences always have an ***even number of taps***.

2) The tap values in a maximal length tap sequence are all ***relatively prime***. A tap sequence like **12, 9, 6, 3** will not be maximal length because the tap values are all divisible by 3.

Discovering one maximal length tap sequence leads automatically to another. If a maximal length tap sequence is described by [**n, A, B, C**], another maximal length tap sequence will be described by [**n, n-C, n-B, n-A**]. Thus, if [**32, 3, 2, 1**] is a maximal length tap sequence, [**32, 31, 30, 29**] will also be a maximal length tap sequence. An interesting behavior of two such tap sequences is that the output bit streams are mirror images in time.

Some polynomials for maximal LFSRs

The following table lists maximal-length polynomials for shift-register lengths up to 18. Note that more than one maximal-length polynomial may exist for any given shift-register length.



Bits	Feedback polynomial	Period
n		$2^n - 1$
2	$x^2 + x + 1$	3
3	$x^3 + x^2 + 1$	7
4	$x^4 + x^3 + 1$	15
5	$x^5 + x^3 + 1$	31
6	$x^6 + x^5 + 1$	63
7	$x^7 + x^6 + 1$	127
8	$x^8 + x^6 + x^5 + x^4 + 1$	255
9	$x^9 + x^5 + 1$	511
10	$x^{10} + x^7 + 1$	1023
11	$x^{11} + x^9 + 1$	2047
12	$x^{12} + x^{11} + x^{10} + x^4 + 1$	4095
13	$x^{13} + x^{12} + x^{11} + x^8 + 1$	8191
14	$x^{14} + x^{13} + x^{12} + x^2 + 1$	16383
15	$x^{15} + x^{14} + 1$	32767
16	$x^{16} + x^{14} + x^{13} + x^{11} + 1$	65535
17	$x^{17} + x^{14} + 1$	131071
18	$x^{18} + x^{11} + 1$	262143

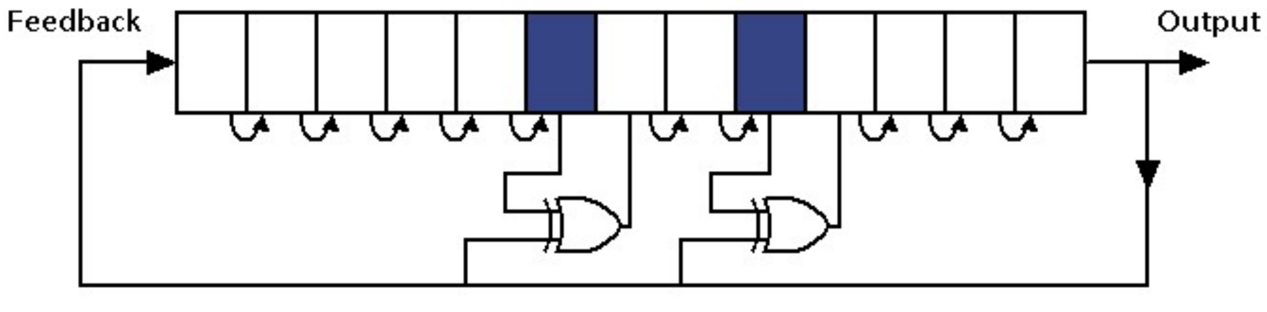
Example: Galois LFSRs

A Galois LFSR, or a LFSR in Galois configuration, is an alternate structure that can generate the same sequences as a conventional LFSR.

In Galois configuration, when the system is clocked, bits that are not taps are shifted as normal. The taps, on the other hand, are XOR'd with the new output, which also becomes the new input. To generate the same sequence, the order of the taps is the reverse of the order for the conventional LFSR.

Galois LFSRs do not concatenate every tap to produce the new input (the XOR'ing is done within the LFSR and no XOR's are run in serial, therefore the propagation times are reduced to that of one XOR rather than a whole chain), thus it is possible for each tap to be computed in parallel, increasing the speed of execution.

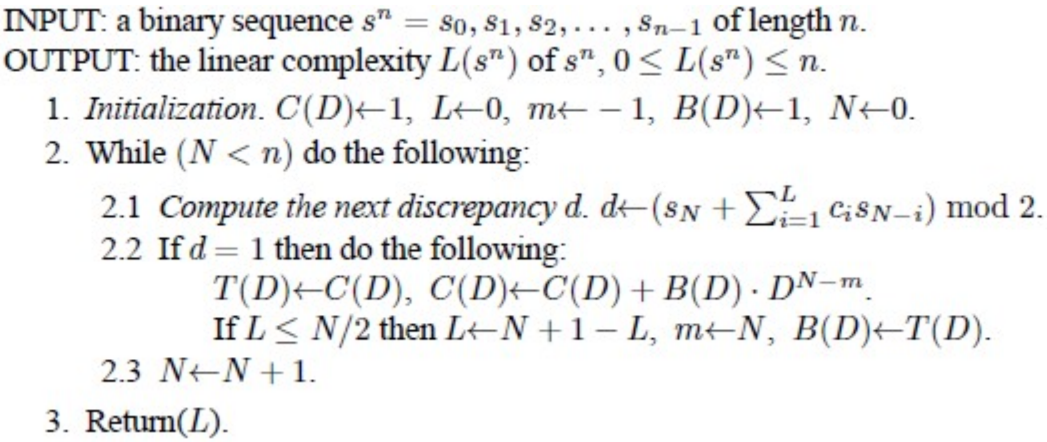
In a software implementation of an LFSR, the Galois form is more efficient **as the XOR operations can be implemented a word at a time: only the output bit must be examined individually.**



5.6 Linear Equivalence

Linear equivalence is define as the length of the smallest linear shift register which can be used to generate the same sequence .the primitive polynomial for the linear equivalence is called minimal polynomial .linear equivalence determines the complexity of the generate sequence .for the sequence with ***m*** linear equivalence ,it needs only ***2m*** of the generate sequence to deduce the whole sequence .hence for good cipher secrecy ,it need to have a generator with large linear equivalence .BerleKamp-Massey algorithm can be used to determine the linear equivalence and the minimum polynomial as shown in the following algorithm:

- **Berlekamp-Massey algorithm**



```
INPUT: a binary sequence $s^n = s_0, s_1, s_2, \ldots, s_{n-1}$ of length $n$.
OUTPUT: the linear complexity $L(s^n)$ of $s^n$, $0 \le L(s^n) \le n$.
  1. Initialization. $C(D) \leftarrow 1,\ L \leftarrow 0,\ m \leftarrow -1,\ B(D) \leftarrow 1,\ N \leftarrow 0$.
  2. While $(N < n)$ do the following:
       2.1 Compute the next discrepancy d. $d \leftarrow (s_N + \sum_{i=1}^{L} c_i s_{N-i}) \bmod 2$.
       2.2 If $d = 1$ then do the following:
             $T(D) \leftarrow C(D),\ C(D) \leftarrow C(D) + B(D) \cdot D^{N-m}$.
             If $L \le N/2$ then $L \leftarrow N + 1 - L,\ m \leftarrow N,\ B(D) \leftarrow T(D)$.
       2.3 $N \leftarrow N + 1$.
  3. Return($L$).
```