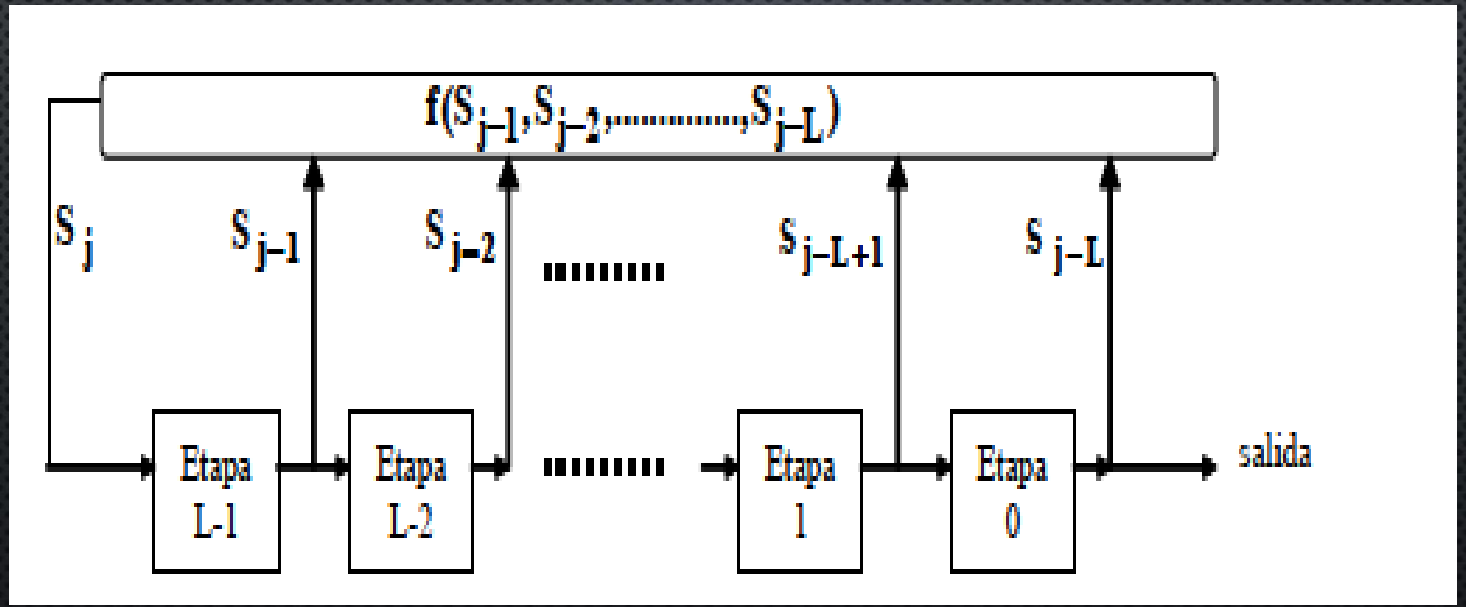
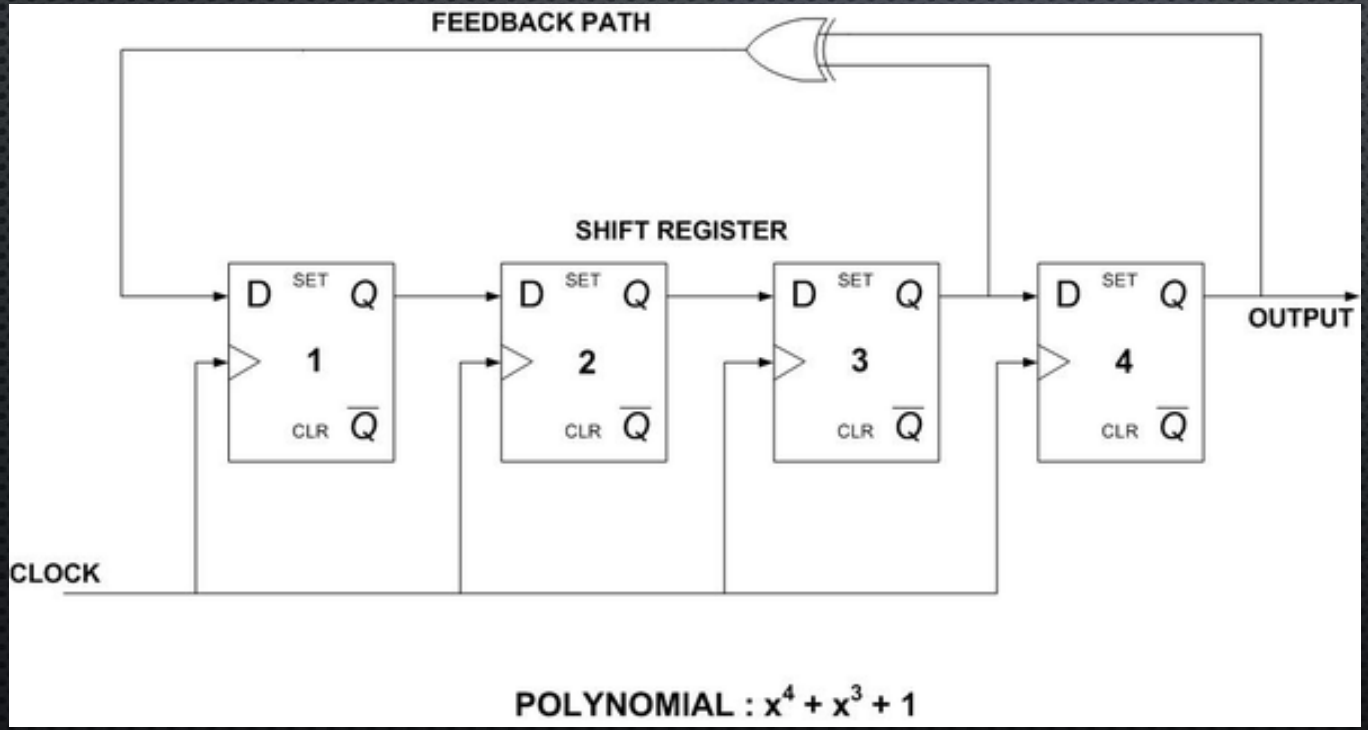
ANALYSIS OF LFSR STRUCTURE FOR N<13 BITS

- Laura Guijarro Iguacel
- Daniel Olivares Garcés

FSR (FEEDBACK SHIFT REGISTER)



LFSR (LINEAR FEEDBACK SHIFT REGISTER)

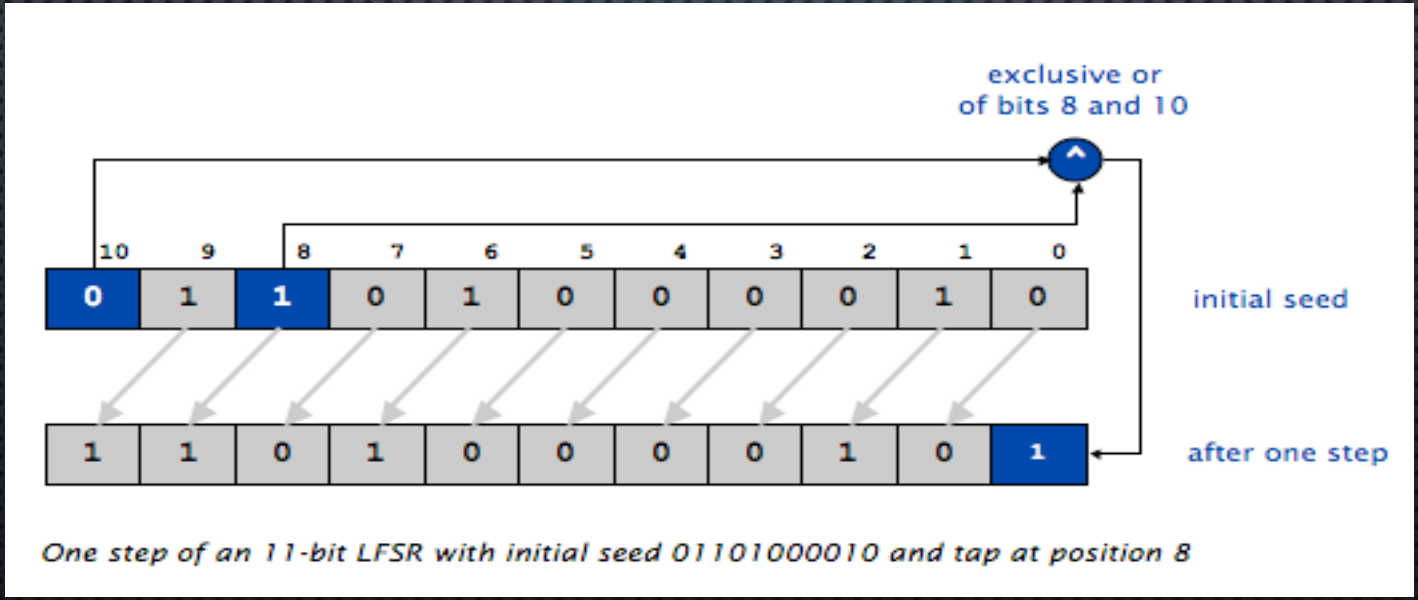


- Length N
- States of the register
- Maximun length sequence L

  L = 2^N -1

LFSR (LINEAR FEEDBACK SHIFT REGISTER)

Example



One step of an 11-bit LFSR with initial seed 01101000010 and tap at position 8

LFSR (LINEAR FEEDBACK SHIFT REGISTER)

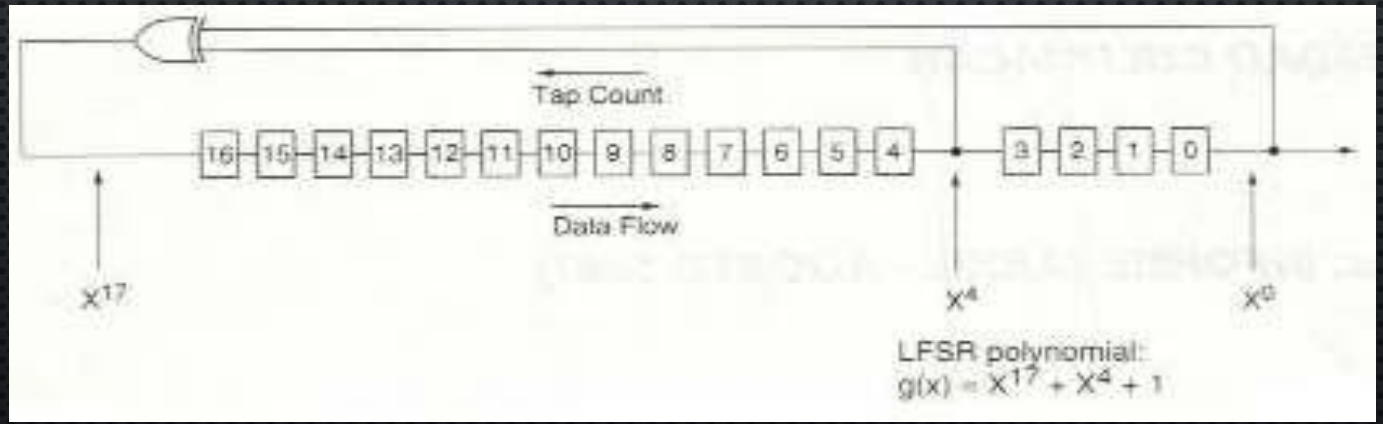
IMPLEMENTATION OF GALOIS

- DATA FLOW FROM LEFT TO RIGHT
- FEEDBACK FROM RIGHT TO LEFT
- INCREMENT OF THE POLYNOMIAL FROM LEFT TO RIGHT, X^0 AS FIRST TERM

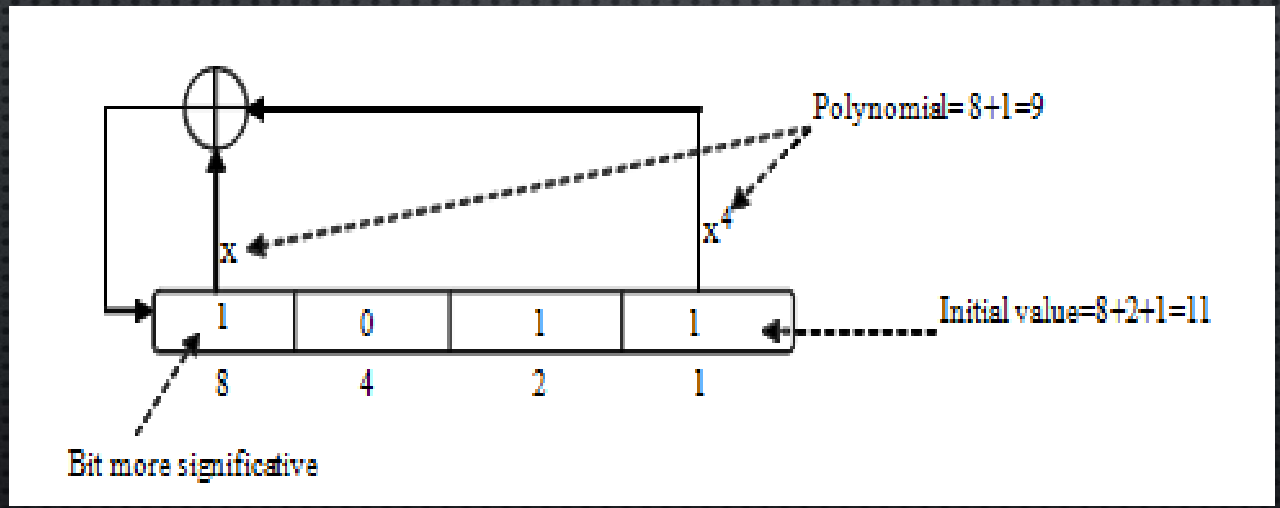
LFSR (LINEAR FEEDBACK SHIFT REGISTER)

IMPLEMENTATION OF FIBONACCI

- DATA FLOW FROM LEFT TO RIGHT
- FEEDBACK FROM RIGHT TO LEFT
- DECREASES OF THE POLYNOMIAL FROM LEFT TO RIGHT, X^0 AS LAST TERM



APLICATTION 1. LFSR OF 4 BITS

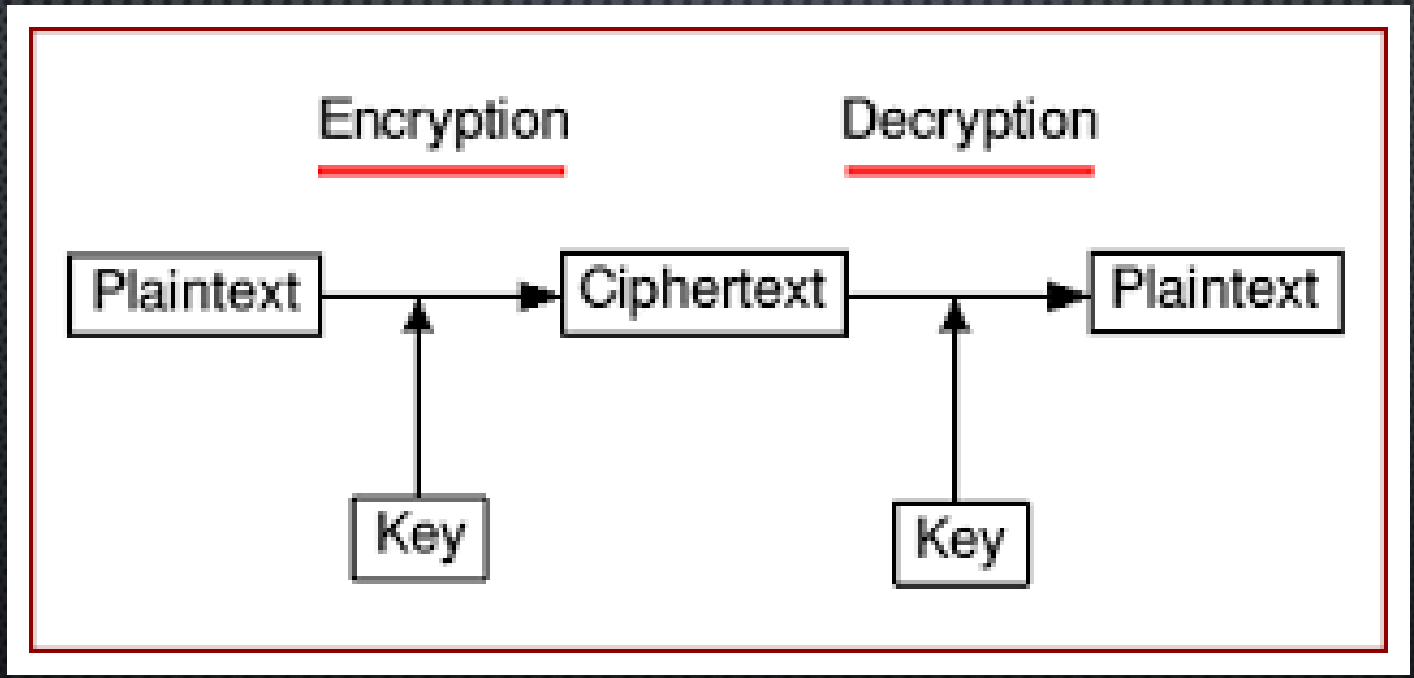


CODE

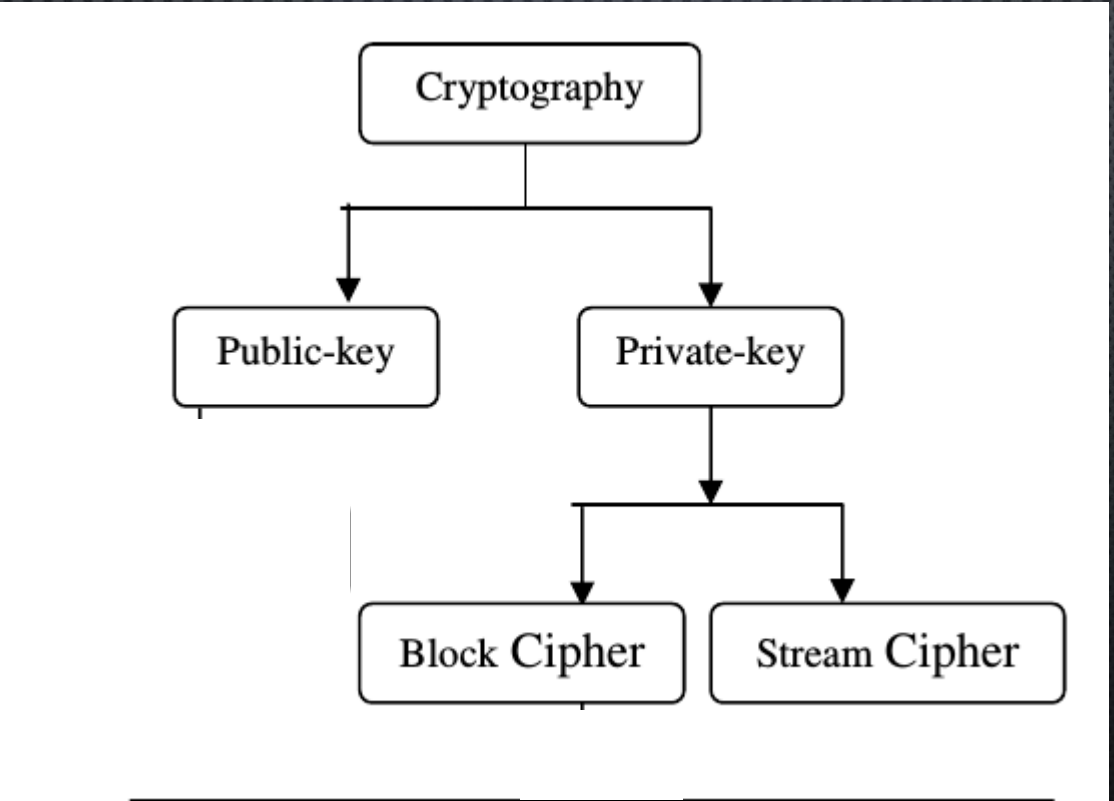
```
process (clk_500)
    begin
        if rising_edge(clk_500) then
            if BTN0 = '0' then
                dec <= "0010";
            else

            decl(3 downto 1) <= dec(2 downto 0);
            dec(0) <= dec(2) XOR dec(1) ;
            dec(3 downto 1) <= decl(3 downto 1);

                    if display= "00"  then
                        if dec(3) = '0' then
                            hex <= "0000";
                            position <= "0111";
                            display <=  "01" ;
                        elsif dec(3) = '1' then
                            hex <= "0001";
                            position <= "0111";
                            display <= "01" ;
                         end if ;
                    end if ;
                    if display= "01"  then
                        if dec(2) = '0' then
                            hex <= "0000";
                            position <= "1011";
                            display <= "10" ;
                        elsif dec(2) = '1' then
                            hex <= "0001";
                            position <= "1011";
                            display <= "10" ;
                         end if ;
                    end if ;
```

```
                    if display= "10"  then
                        if dec(1) = '0' then
                            hex <= "0000";
                            position <= "1101";
                            display <= "11" ;
                        elsif dec(1) = '1' then
                            hex <= "0001";
                            position <= "1101";
                            display <= "11" ;
                         end if ;
                    end if ;
                    if display= "11" then
                        if dec(0) = '0' then
                            hex <= "0000";
                            position <= "1110";
                            display <= "00" ;
                        elsif dec(0) = '1' then
                            hex <= "0001";
                            position <= "1110";
                            display <= "00" ;

                         end if ;
                    end if ;
```

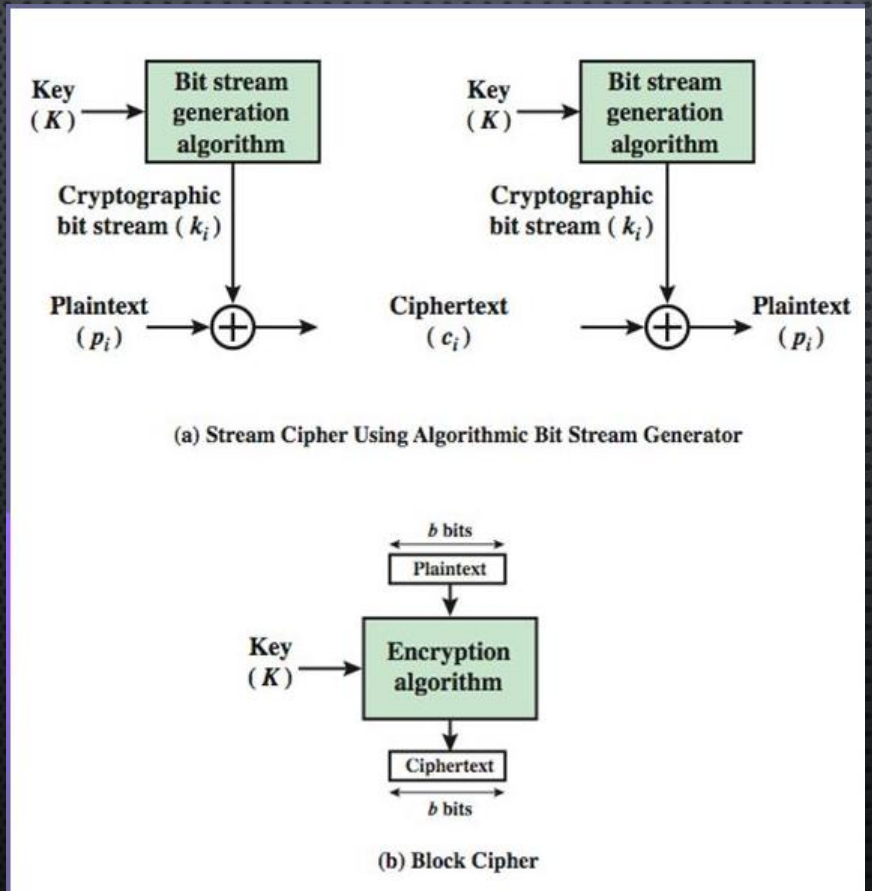
CRYPTOGRAPHY



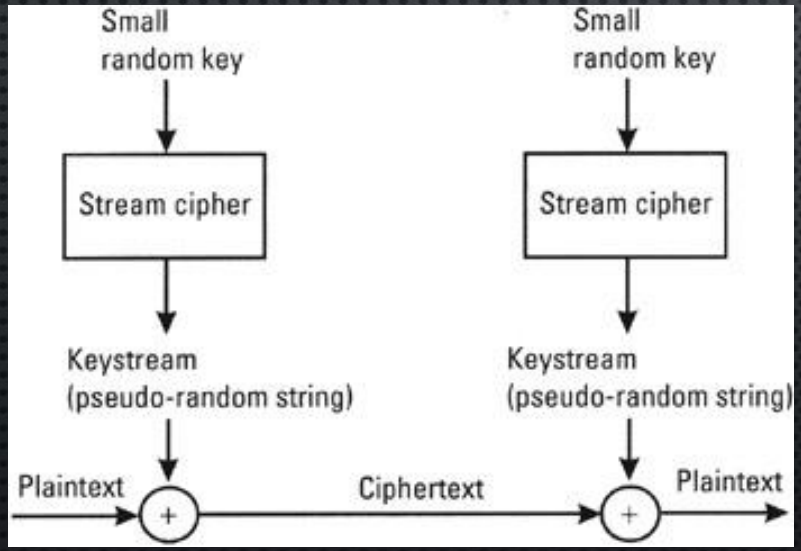
CRYPTOGRAPHY



BLOCK CIPHER VS STREAM CIPHER



STREAM CIPHER



The stream cipher can be performed by means of pseudorandom sequence generators, which use a function *f* and a key.

Using the pseudorandom sequence every bit of the clear text is encrypted using a Boolean function, for example an XOR .

STREAM CIPHERS

MATLAB EXAMPLE

LFSR

```
%Implementation of LFRS

s=[0 0 1 0 ] %Initial value
t=[3 2] %function f(x)= 1+x^2+x^3

n=length(s);
c(1,:)=s;
m=length(t);
for k=1:2^n-2;
b(1)=xor(s(t(1)), s(t(2)));
if m>2;
    for i=1:m-2;
    b(i+1)=xor(s(t(i+2)), b(i));
    end
end
j=1:n-1;
s(n+1-j)=s(n-j);
s(1)=b(m-1);
c(k+1,:)=s;
end
seq=c(:,n)';
```

STREAM CYPHER

```
n=4; %length of the cypher

for i=1:2^n -1
    numero = i
    numero_binario =dec2bin(i,n)

    vector_numero_binario=num2str(numero_binario)-'0'

    codigo_cod=c(i,:)
    numero_codificado =xor (vector_numero_binario,c(i,:))
    numero_salida = xor (numero_codificado,c(i,:))

    matriz_mensaje(i,:)=(vector_numero_binario)
    matriz_codificada(i,:)=(numero_codificado)
    matriz_salida(i,:)=(numero_salida)
end
```

STREAM CIPHERS

MATLAB EXAMPLE

PSEUDORANDOM SEQUENCE

```
c =

     0     0     1     0
     1     0     0     1
     0     1     0     0
     1     0     1     0
     1     1     0     1
     1     1     1     0
     0     1     1     1
     0     0     1     1
     1     0     0     1
     0     1     0     0
     1     0     1     0
     1     1     0     1
     1     1     1     0
     0     1     1     1
     0     0     1     1
```

DATA MATRIX

```
matriz_mensaje =

     0     0     0     1
     0     0     1     0
     0     0     1     1
     0     1     0     0
     0     1     0     1
     0     1     1     0
     0     1     1     1
     1     0     0     0
     1     0     0     1
     1     0     1     0
     1     0     1     1
     1     1     0     0
     1     1     0     1
     1     1     1     0
     1     1     1     1
```

STREAM CIPHERS

MATLAB EXAMPLE

DATA MATRIX

```
matriz_mensaje =

     0     0     0     1
     0     0     1     0
     0     0     1     1
     0     1     0     0
     0     1     0     1
     0     1     1     0
     0     1     1     1
     1     0     0     0
     1     0     0     1
     1     0     1     0
     1     0     1     1
     1     1     0     0
     1     1     0     1
     1     1     1     0
     1     1     1     1
```

CODIFIED MATRIX

```
matriz_codificada =

     0     0     1     1
     1     0     1     1
     0     1     1     1
     1     1     1     0
     1     0     0     0
     1     0     0     0
     0     0     0     0
     1     0     1     1
     0     0     0     0
     1     1     1     0
     0     0     0     1
     0     0     0     1
     0     0     1     1
     1     0     0     1
     1     1     0     0
```

STREAM CIPHERS

MATLAB EXAMPLE

DATA MATRIX

```
matriz_mensaje =

     0     0     0     1
     0     0     1     0
     0     0     1     1
     0     1     0     0
     0     1     0     1
     0     1     1     0
     0     1     1     1
     1     0     0     0
     1     0     0     1
     1     0     1     0
     1     0     1     1
     1     1     0     0
     1     1     0     1
     1     1     1     0
     1     1     1     1
```

CODIFIED MATRIX

```
matriz_codificada =

     0     0     1     1
     1     0     1     1
     0     1     1     1
     1     1     1     0
     1     0     0     0
     1     0     0     0
     0     0     0     0
     1     0     1     1
     0     0     0     0
     1     1     1     0
     0     0     0     1
     0     0     0     1
     0     0     1     1
     1     0     0     1
     1     1     0     0
```

- OUTPUT MATRIX

```
matriz_salida =

     0     0     0     1
     0     0     1     0
     0     0     1     1
     0     1     0     0
     0     1     0     1
     0     1     1     0
     0     1     1     1
     1     0     0     0
     1     0     0     1
     1     0     1     0
     1     0     1     1
     1     1     0     0
     1     1     0     1
     1     1     1     0
     1     1     1     1
```