Prime Fields : $\mathbb{Z}_p$

In [2]:
```
p:7
```
Out[2]: 7

In [3]:
```
n:1
```
Out[3]: 1

In [4]:
```
gf_set_data(p,n)
```
Out[4]: Structure [GF-DATA]

In [5]:
```
gf_info()
```
```
characteristic = 7
reduction polynomial = x
primitive element = 3
nr of elements = 7
nr of units = 6
nr of primitive elements = 2
```
Out[5]: **false**

In [6]:
```
gf_add_table()
```
Out[6]:
$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 2 & 3 & 4 & 5 & 6 & 0 \\ 2 & 3 & 4 & 5 & 6 & 0 & 1 \\ 3 & 4 & 5 & 6 & 0 & 1 & 2 \\ 4 & 5 & 6 & 0 & 1 & 2 & 3 \\ 5 & 6 & 0 & 1 & 2 & 3 & 4 \\ 6 & 0 & 1 & 2 & 3 & 4 & 5 \end{pmatrix}$$

In [7]:
```
gf_mult_table()
```
Out[7]:
$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 4 & 6 & 1 & 3 & 5 \\ 3 & 6 & 2 & 5 & 1 & 4 \\ 4 & 1 & 5 & 2 & 6 & 3 \\ 5 & 3 & 1 & 6 & 4 & 2 \\ 6 & 5 & 4 & 3 & 2 & 1 \end{pmatrix}$$

If an element α generates with its powers all the elements of the group $\mathbb{Z}_p^*$

then α is a generator of the cyclic multiplicative group and is called **primitive**.

In $\mathbb{Z}_7^*$, 3 and 5 are the **primitive** elements.

In [27]:
```
for i:1 thru p-1 do print(i,gf_order(i))
```
```
1 1
2 3
3 6
4 3
5 6
6 2
```
Out[27]: **done**

Fermat's little theorem

$$p \mid (a^p - a)$$

or, said in another way

$$a^p - a \equiv 0 \mod p$$

In [29]:
```
for a:1 thru p-1 do print(a,mod(a^p-a,p))
```

```
1 0
2 0
3 0
4 0
5 0
6 0
```

Out[29]: **done**

if $a \neq 0$ then we can multiply by a^{-1}

$a^{p-1} - 1 \equiv 0 \mod p$ or $a^{p-1} \equiv 1 \mod p$

In [30]:
```
for a:1 thru p-1 do print(a,mod(a^(p-1),p))
```

```
1 1
2 1
3 1
4 1
5 1
6 1
```

Out[30]: **done**

Inverse computed with Fermat's little theorem

let's multiply again by a^{-1}

$a^{p-2} \equiv a^{-1} \mod p$

In [11]:
```
for a:2 thru p-1 do print(a,"^-1=",mod(a^(p-2),p))
```

```
2 ^-1= 4
3 ^-1= 5
4 ^-1= 2
5 ^-1= 3
6 ^-1= 6
```

Out[11]: **done**

In [12]:
```
gf_primitive()
```

Out[12]: 3

In [16]:
```
gf_make_logs()
```

Out[16]: ["{Lisp Array: \#(1 3 2 6 4 5 1)}" , "{Lisp Array: \#(NIL 0 2 1 4 5 3)}" , "{Lisp Array: \#(2 4 1

In [24]:
```
for j:0 thru p-2 do print(gf_primitive(),"^",j,"=",gf_powers[j])
```

```
3 ^ 0 = 1
3 ^ 1 = 3
3 ^ 2 = 2
3 ^ 3 = 6
3 ^ 4 = 4
3 ^ 5 = 5
```

Out[24]: **done**

In [ ]:

Prime Power Fields : $\mathbb{F}_q = \mathbb{Z}_p[x]/m(x) = GF(p^n)$

In [ ]:
```
p:3
```

In [ ]:
```
n:2
```

In [3]:
```
gf_set_data(p,n)
```

Out[3]: Structure [GF-DATA]

In [4]:
```
gf_info()
```

```
characteristic = 3
reduction polynomial = x^2+1
primitive element = x+1
nr of elements = 9
nr of units = 8
nr of primitive elements = 4
```

Out[4]: **false**

In [18]:
```
atable:gf_add_table()
```

Out[18]:

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 2 & 0 & 4 & 5 & 3 & 7 & 8 & 6 \\ 2 & 0 & 1 & 5 & 3 & 4 & 8 & 6 & 7 \\ 3 & 4 & 5 & 6 & 7 & 8 & 0 & 1 & 2 \\ 4 & 5 & 3 & 7 & 8 & 6 & 1 & 2 & 0 \\ 5 & 3 & 4 & 8 & 6 & 7 & 2 & 0 & 1 \\ 6 & 7 & 8 & 0 & 1 & 2 & 3 & 4 & 5 \\ 7 & 8 & 6 & 1 & 2 & 0 & 4 & 5 & 3 \\ 8 & 6 & 7 & 2 & 0 & 1 & 5 & 3 & 4 \end{pmatrix}$$

In [19]:
```
for i:1 thru p^n do for j:1 thru p^n do atable[i,j]:gf_n2p(atable[i,j])
```

Out[19]: **done**

In [20]:
```
print(atable)$
```

```
        [    0    ]         [    1    ]         [    2    ]
        [         ]         [         ]         [         ]
        [    1    ]         [    2    ]         [    0    ]
        [         ]         [         ]         [         ]
        [    2    ]         [    0    ]         [    1    ]
        [         ]         [         ]         [         ]
        [    x    ]         [  x + 1  ]         [  x + 2  ]
        [         ]         [         ]         [         ]
Col 1 = [  x + 1  ] Col 2 = [  x + 2  ] Col 3 = [    x    ]
        [         ]         [         ]         [         ]
        [  x + 2  ]         [    x    ]         [  x + 1  ]
        [         ]         [         ]         [         ]
        [   2 x   ]         [ 2 x + 1 ]         [ 2 x + 2 ]
        [         ]         [         ]         [         ]
        [ 2 x + 1 ]         [ 2 x + 2 ]         [   2 x   ]
        [         ]         [         ]         [         ]
        [ 2 x + 2 ]         [   2 x   ]         [ 2 x + 1 ]
        [    x    ]         [  x + 1  ]         [  x + 2  ]
        [         ]         [         ]         [         ]
        [  x + 1  ]         [  x + 2  ]         [    x    ]
        [         ]         [         ]         [         ]
        [  x + 2  ]         [    x    ]         [  x + 1  ]
        [         ]         [         ]         [         ]
        [   2 x   ]         [ 2 x + 1 ]         [ 2 x + 2 ]
        [         ]         [         ]         [         ]
Col 4 = [ 2 x + 1 ] Col 5 = [ 2 x + 2 ] Col 6 = [   2 x   ]
        [         ]         [         ]         [         ]
        [ 2 x + 2 ]         [   2 x   ]         [ 2 x + 1 ]
        [         ]         [         ]         [         ]
        [    0    ]         [    1    ]         [    2    ]
        [         ]         [         ]         [         ]
        [    1    ]         [    2    ]         [    0    ]
        [         ]         [         ]         [         ]
        [    2    ]         [    0    ]         [    1    ]
        [   2 x   ]         [ 2 x + 1 ]         [ 2 x + 2 ]
        [         ]         [         ]         [         ]
        [ 2 x + 1 ]         [ 2 x + 2 ]         [   2 x   ]
        [         ]         [         ]         [         ]
        [ 2 x + 2 ]         [   2 x   ]         [ 2 x + 1 ]
        [         ]         [         ]         [         ]
        [    0    ]         [    1    ]         [    2    ]
        [         ]         [         ]         [         ]
Col 7 = [    1    ] Col 8 = [    2    ] Col 9 = [    0    ]
        [         ]         [         ]         [         ]
        [    2    ]         [    0    ]         [    1    ]
        [         ]         [         ]         [         ]
        [    x    ]         [  x + 1  ]         [  x + 2  ]
        [         ]         [         ]         [         ]
        [  x + 1  ]         [  x + 2  ]         [    x    ]
        [         ]         [         ]         [         ]
        [  x + 2  ]         [    x    ]         [  x + 1  ]
```

Out[20]:

$$\begin{pmatrix} 0 & 1 & 2 & x & x+1 & x+2 & 2\,x & 2\,x+1 & 2\,x+2 \\ 1 & 2 & 0 & x+1 & x+2 & x & 2\,x+1 & 2\,x+2 & 2\,x \\ 2 & 0 & 1 & x+2 & x & x+1 & 2\,x+2 & 2\,x & 2\,x+1 \\ x & x+1 & x+2 & 2\,x & 2\,x+1 & 2\,x+2 & 0 & 1 & 2 \\ x+1 & x+2 & x & 2\,x+1 & 2\,x+2 & 2\,x & 1 & 2 & 0 \\ x+2 & x & x+1 & 2\,x+2 & 2\,x & 2\,x+1 & 2 & 0 & 1 \\ 2\,x & 2\,x+1 & 2\,x+2 & 0 & 1 & 2 & x & x+1 & x+2 \\ 2\,x+1 & 2\,x+2 & 2\,x & 1 & 2 & 0 & x+1 & x+2 & x \\ 2\,x+2 & 2\,x & 2\,x+1 & 2 & 0 & 1 & x+2 & x & x+1 \end{pmatrix}$$

In [21]:
```
mtable:gf_mult_table()$
```

Out[21]:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 1 & 6 & 8 & 7 & 3 & 5 & 4 \\ 3 & 6 & 2 & 5 & 8 & 1 & 4 & 7 \\ 4 & 8 & 5 & 6 & 1 & 7 & 2 & 3 \\ 5 & 7 & 8 & 1 & 3 & 4 & 6 & 2 \\ 6 & 3 & 1 & 7 & 4 & 2 & 8 & 5 \\ 7 & 5 & 4 & 2 & 6 & 8 & 3 & 1 \\ 8 & 4 & 7 & 3 & 2 & 5 & 1 & 6 \end{pmatrix}$$

In [23]:
```
for i:1 thru p^n-1 do for j:1 thru p^n-1 do mtable[i,j]:gf_n2p(mtable[i,j])
```

Out[23]: **done**

In [24]:
```
print(mtable)
```

```
[    1        2        x      x + 1    x + 2     2 x    2 x + 1  2 x + 2 ]
[                                                                        ]
[    2        1       2 x    2 x + 2  2 x + 1     x      x + 2    x + 1  ]
[                                                                        ]
[    x       2 x       2      x + 2   2 x + 2     1      x + 1   2 x + 1 ]
[                                                                        ]
[  x + 1   2 x + 2   x + 2     2 x       1     2 x + 1     2        x    ]
[                                                                        ]
[  x + 2   2 x + 1  2 x + 2     1        x      x + 1     2 x       2    ]
[                                                                        ]
[   2 x       x        1     2 x + 1   x + 1      2     2 x + 2   x + 2  ]
[                                                                        ]
[ 2 x + 1   x + 2    x + 1      2       2 x    2 x + 2     x        1    ]
[                                                                        ]
[ 2 x + 2   x + 1   2 x + 1     x        2      x + 2      1       2 x   ]
```

Out[24]:

$$\begin{pmatrix} 1 & 2 & x & x+1 & x+2 & 2x & 2x+1 & 2x+2 \\ 2 & 1 & 2x & 2x+2 & 2x+1 & x & x+2 & x+1 \\ x & 2x & 2 & x+2 & 2x+2 & 1 & x+1 & 2x+1 \\ x+1 & 2x+2 & x+2 & 2x & 1 & 2x+1 & 2 & x \\ x+2 & 2x+1 & 2x+2 & 1 & x & x+1 & 2x & 2 \\ 2x & x & 1 & 2x+1 & x+1 & 2 & 2x+2 & x+2 \\ 2x+1 & x+2 & x+1 & 2 & 2x & 2x+2 & x & 1 \\ 2x+2 & x+1 & 2x+1 & x & 2 & x+2 & 1 & 2x \end{pmatrix}$$

In [25]:
```
gf_make_logs()
```

Out[25]: ["{Lisp Array: \#(1 4 6 7 2 8 3 5 1)}" , "{Lisp Array: \#(NIL 0 4 6 1 7 2 3 5)}" , "{Lisp Array: \#
]

In [31]:
```
for i:1 thru p^n-1 do print("(",gf_primitive(),")^",i,"=",gf_n2p(gf_powers[i]))
```

```
( x + 1 )^ 1 = x + 1
( x + 1 )^ 2 = 2 x
( x + 1 )^ 3 = 2 x + 1
( x + 1 )^ 4 = 2
( x + 1 )^ 5 = 2 x + 2
( x + 1 )^ 6 = x
( x + 1 )^ 7 = x + 2
( x + 1 )^ 8 = 1
```

Out[31]: **done**

In [32]:
```
for i:1 thru p^n-1 do print(gf_n2p(i)," has order ",gf_order(gf_n2p(i)))
```

```
1  has order  1
2  has order  2
x  has order  4
x + 1  has order  8
x + 2  has order  8
2 x  has order  4
2 x + 1  has order  8
2 x + 2  has order  8
```

Out[32]: **done**

In [35]:
```
gf_factor(x^(p^n)-x,3)
```

Out[35]: $x\ (x+1)\ (x+2)\ \left(x^2+1\right)\ \left(x^2+x+2\right)\ \left(x^2+2\,x+2\right)$

Universal Polynomial : $x^{p^n}-x$

factors in all irreducible monic polynomials of degree $k \mid n$ In this case $n=2$ and therefore it factors in all irreducibles of degree 1 or 2

In [64]:
```
gf_factor(x^p^n-x)
```

Out[64]: $x\ (x+1)\ (x+2)\ \left(x^2+1\right)\ \left(x^2+x+2\right)\ \left(x^2+2\,x+2\right)$

In [64]:
```
gf_factor(x^2+1)
```

Out[64]: x^2+1

In [64]:
```
for i:0 thru p^n-1 do ( print(gf_add(x,-gf_n2p(i))))
```

```
x
x + 2
x + 1
0
2
1
2 x
2 x + 2
2 x + 1
```

Out[64]: **done**

In [ ]:

Primitive polynomials

$primitive\ polynomials \subset irreducible\ polynomials \subset polynomials$

Is $f(x) = x^3 + x + 1$ a *primitive polynomial* over $\mathbb{Z}_2$?

In [1]:
```
f:x^3+x+1;
```
Out[1]: $x^3 + x + 1$

In [2]:
```
n:hipow(f,x);
```
Out[2]: 3

In [3]:
```
p:modulus:2;
```
Out[3]: 2

To be primitive $f(x)$ should divide $x^{p^n-1} - 1$ and no other $x^e - 1$ for $e < p^n - 1$

In [4]:
```
divide(x^(p^n-1)-1,f);
```
Out[4]: $[x^4 + x^2 + x + 1, 0]$

In [5]:
```
for e:1 thru p^n-1 do print("e=",e,",",divide(x^e-1,f));
```
```
e= 1 , [0, x + 1]
               2
e= 2 , [0, x  + 1]
e= 3 , [1, x]
               2
e= 4 , [x, x  + x + 1]
          2       2
e= 5 , [x  + 1, x  + x]
          3           2
e= 6 , [x  + x + 1, x ]
          4    2
e= 7 , [x  + x  + x + 1, 0]
```
Out[5]: **done**

In [6]:
```
gf_primitive_poly_p(f,p);
```
Out[6]: **true**

In [7]:
```
factor(x^(p^n-1)-1);
```
Out[7]: $(x+1)\left(x^3+x+1\right)\left(x^3+x^2+1\right)$

Universal polynomial : $U(x) = x^{p^n} - x$

The universal polynomial is the product of all irreducible polynomials of degree d for $\forall d : d \mid n$

In [8]:
```
factor(x^p^n -x);
```
Out[8]: $x\ (x+1)\ \left(x^3+x+1\right)\ \left(x^3+x^2+1\right)$

If $f(x)$ is a primitive polynomial of degree n over $\mathbb{Z}_p$ then x is a generator of $\mathbb{F}_{p^n} = \mathbb{Z}_p[x]/f(x)$

In [9]:
```
gf_set_data(p,n);
```
Out[9]: Structure [GF-DATA]

In [10]:
```
for i:1 thru p^n-1 do print("x^",i,"=",gf_exp(x,i));
```
```
x^ 1 = x
        2
x^ 2 = x
x^ 3 = x + 1
        2
x^ 4 = x  + x
        2
x^ 5 = x  + x + 1
        2
x^ 6 = x  + 1
x^ 7 = 1
```
Out[10]: **done**

The above should give you a hint of why the LFSR works. Multiplying by x is shifting left.

If x is a generator then the order of x should be $p^n - 1$

In [11]:
```
for i:1 thru p^n-2 do print(gf_exp(x,i)," has order ",gf_order(gf_exp(x,i)))
;
```
```
x  has order  7
 2
x   has order  7
x + 1  has order  7
 2
x  + x  has order  7
 2
x  + x + 1  has order  7
 2
x  + 1  has order  7
```
Out[11]: **done**

If we know the factorization of $p^n - 1$ then we can check that $\forall\ primes\ q|(p^n-1), f \nmid x^{\frac{p^n-1}{q}} - 1$

In [12]:
```
modulus:3;
```
Out[12]: 3

In [13]:
```
p:3;
```
Out[13]: 3

In [14]:
```
n:4;
```
Out[14]: 4

In [15]:
```
f:gf_primitive_poly(p,n);
```
Out[15]: $x^4 + x + 2$

In [16]:
```
ifactors(p^n-1);
```
Out[16]: $[[2,4],[5,1]]$

Prime factors of 80 are $2, 5$

In [17]:
```
q:2;
```
Out[17]: 2

In [18]:
```
divide(x^((p^n-1)/q)-1,f);
```
Out[18]: $\left[x^{36} - x^{33} + x^{32} + x^{30} + x^{29} + x^{28} - x^{27} - x^{24} - x^{23} + x^{21} - x^{19} - x^{18} + x^{17} + x^{16} - x^6 + x^4 - x^3 - x^2 - x - 1, 1\right]$

In [19]:
```
q:5;
```
Out[19]: 5

In [20]:
```
divide(x^((p^n-1)/q)-1,f);
```
Out[20]: $\left[x^{12} - x^9 + x^8 + x^6 + x^5 + x^4 - x^3 - 1, -x^3 + x + 1\right]$

In [ ]:

In [ ]:

LFSR Linear Feedback Shift Register

An LFSR has maximal period if its associated/connection polynomial is primitive.

In that case if n is the length of the LFSR, the period will be $2^n - 1$

In [1]:
```
p:2
```
Out[1]: 2

In [2]:
```
n:4
```
Out[2]: 4

A poly of degree m over $\mathbb{Z}_p$ is primitive if its order is $p^m - 1$

Here $p = 2, n = 4$, therefore $p^m - 1 = 2^4 - 1 = 15$

In [3]:
```
f:gf_primitive_poly(p,n)
```
Out[3]: $x^4 + x + 1$

In [4]:
```
modulus:2
```
Out[4]: 2

In [5]:
```
gf_set_data(p,n)
```
Out[5]: $\text{Structure [GF-DATA]}$

In [6]:
```
gf_order(f(x))
```
Out[6]: 15

f(x) is a primitive polynomial so we can expect a period of $2^4 - 1 = 15$ from its LFSR

In [7]:
```
modulus:2
```
Out[7]: 2

In [8]:
```
seed:[0,1,0,1]
```
Out[8]: $[0, 1, 0, 1]$

This matrix does exactly what a LFSR does: shifts right and replaces first bit with the xor of the taps

In [9]:
```
mlfsr:matrix([1,1,0,0],
             [0,0,1,0],
             [0,0,0,1],
             [1,0,0,0])
```
Out[9]:

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

In [10]:
```
expand(charpoly(mlfsr,lambda))
```
Out[10]: $\lambda^4 - \lambda^3 - 1$

In [11]:
```
gf_primitive_poly_p(%,p)
```
Out[11]: **true**

In [14]:
```
for i:1 thru p^n-1 do ( seed:seed . mlfsr, seed:mod(seed,2),print(i,seed) )
```
```
1 [ 0  1  1  0 ]
2 [ 0  0  1  1 ]
3 [ 1  0  0  1 ]
4 [ 0  1  0  0 ]
5 [ 0  0  1  0 ]
6 [ 0  0  0  1 ]
7 [ 1  0  0  0 ]
8 [ 1  1  0  0 ]
9 [ 1  1  1  0 ]
10 [ 1  1  1  1 ]
11 [ 0  1  1  1 ]
12 [ 1  0  1  1 ]
13 [ 0  1  0  1 ]
14 [ 1  0  1  0 ]
15 [ 1  1  0  1 ]
```
Out[14]: **done**

In [64]:
```
p:3
```
Out[64]: 3

In [64]:
```
n:3
```
Out[64]: 3

In [64]:
```
gf_primitive_poly(p,n)
```
Out[64]: $x^3 + 2\,x + 1$

In [64]:
```
seed:[0,2,1]
```
Out[64]: $[0, 2, 1]$

In [64]:
```
mlfsr:matrix([2,1,0],
             [0,0,1],
             [1,0,0])
```
Out[64]: $\begin{pmatrix} 2 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$

In [64]:
```
expand(charpoly(mlfsr,lambda))
```
Out[64]: $-\lambda^3 + 2\,\lambda^2 + 1$

In [64]:
```
gf_primitive_poly_p(%,p)
```
Out[64]: **false**

In [64]:
```
for i:1 thru p^n do ( seed:seed . mlfsr, seed:mod(seed,p),print(i,seed) )
```

```
1 [ 1  0  2 ]
2 [ 1  1  0 ]
3 [ 2  1  1 ]
4 [ 2  2  1 ]
5 [ 2  2  2 ]
6 [ 0  2  2 ]
7 [ 2  0  2 ]
8 [ 0  2  0 ]
9 [ 0  0  2 ]
10 [ 2  0  0 ]
11 [ 1  2  0 ]
12 [ 2  1  2 ]
13 [ 0  2  1 ]
14 [ 1  0  2 ]
15 [ 1  1  0 ]
16 [ 2  1  1 ]
17 [ 2  2  1 ]
18 [ 2  2  2 ]
19 [ 0  2  2 ]
20 [ 2  0  2 ]
21 [ 0  2  0 ]
22 [ 0  0  2 ]
23 [ 2  0  0 ]
24 [ 1  2  0 ]
25 [ 2  1  2 ]
26 [ 0  2  1 ]
27 [ 1  0  2 ]
```

Out[64]: **done**

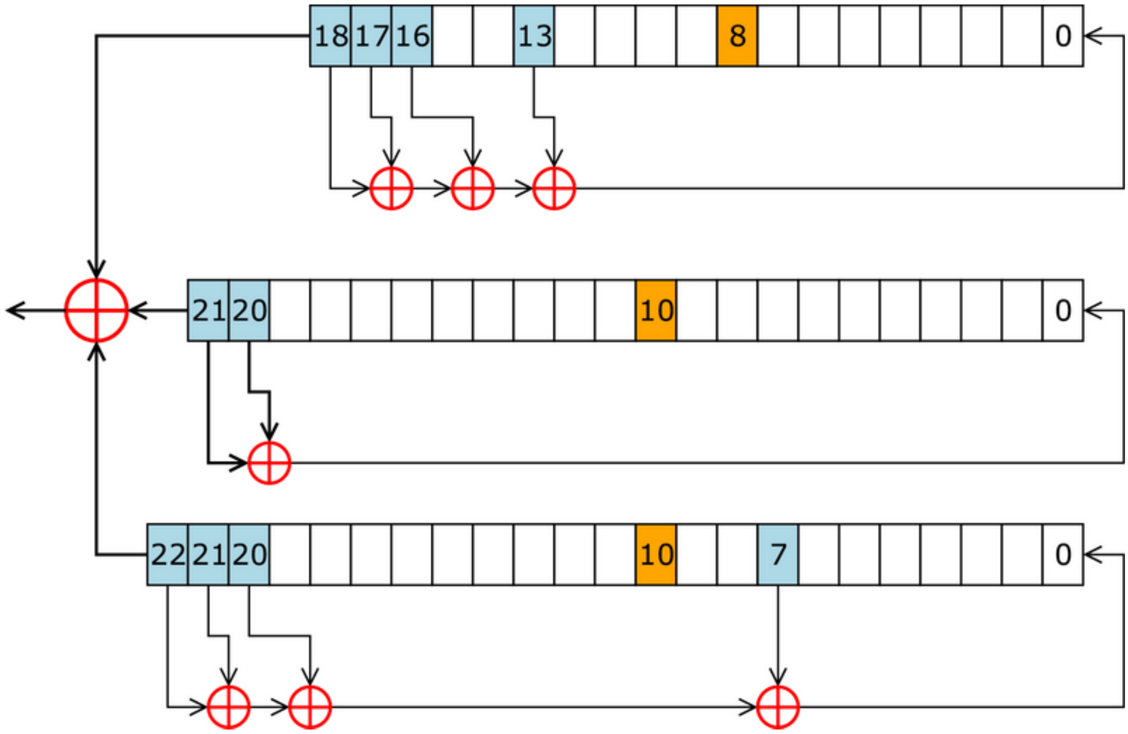
In [ ]:

GSM A5/1 cipher

3 different LFSR are initialized with 64 bits

and their output is xored to produce 114 bits at a time

these bits are xored with 114 bits of the data stream and sent every 4.615 msec



pic from Matt Crypto - Wikipedia

In [1]:
```
lfsr1:x^19+x^18+x^17+x^14+1
```
Out[1]: $x^{19}+x^{18}+x^{17}+x^{14}+1$

In [2]:
```
gf_primitive_poly_p(lfsr1,2)
```
Out[2]: **true**

In [3]:
```
lfsr2:x^22+x^21+1
```
Out[3]: $x^{22}+x^{21}+1$

In [4]:
```
gf_primitive_poly_p(lfsr2,2)
```
Out[4]: **true**

In [5]:
```
lfsr3:x^23+x^22+x^21+x^8+1
```
Out[5]: $x^{23}+x^{22}+x^{21}+x^{8}+1$

In [6]:
```
gf_primitive_poly_p(lfsr3,2)
```
Out[6]: **true**

In [ ]:

GFSR - Lewis , Payne 1973

Uses the same LFSR across all bits of the words : if the degree of the LFSR is n then the period is $2^n - 1$.

The initialization is problematic and slow especially in the original rng.

If we see the state as made by n vertical words then fill each row of bits one after the other with the LFSR skipping some outcomes after any row :

$$[\mathbf{w_0}, \mathbf{w_1}, \ldots] = \begin{bmatrix} w_{0,0} & \ldots & w_{0,n-1} \\ w_{1,0} & \ldots & w_{1,n-1} \\ w_{2,0} & \ldots & w_{2,n-1} \end{bmatrix}$$

In [2]:
```
p:2
```
Out[2]: 2

In [3]:
```
n:3
```
Out[3]: 3

In [4]:
```
gf_primitive_poly(p,3)
```
Out[4]: $x^3 + x + 1$

Therefore a $LFSR(3, 1 + x + x^3)$ will have maximal period $p^n - 1 = 2^3 - 1 = 8 - 1 = 7$

In [5]:
```
seed:[1,0,1]
```
Out[5]: $[1, 0, 1]$

In [6]:
```
lfsr:matrix([1,1,0],
            [0,0,1],
            [1,0,0])
```
Out[6]:
$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

In [7]:
```
for i:1 thru p^n  do ( seed:mod(seed . lfsr,p), print(i,seed))
```
```
1 [ 0  1  0 ]
2 [ 0  0  1 ]
3 [ 1  0  0 ]
4 [ 1  1  0 ]
5 [ 1  1  1 ]
6 [ 0  1  1 ]
7 [ 1  0  1 ]
8 [ 0  1  0 ]
```
Out[7]: **done**

Let's now build a GFSR with the same polynomial using numbers $0 \le x \le m : 2^3$

In [17]:
```
m:2^3
```
Out[17]: 8

In [20]:
```
mseed:matrix([7,3,2])
```

Out[20]: $\begin{pmatrix} 7 & 3 & 2 \end{pmatrix}$

In [21]:
```
for i:1 thru p^m-1 do ( mseed:mod(mseed . lfsr,m), print(i,mseed))
```

```
1 [ 1  7  3 ]
2 [ 4  1  7 ]
3 [ 3  4  1 ]
4 [ 4  3  4 ]
5 [ 0  4  3 ]
6 [ 3  0  4 ]
7 [ 7  3  0 ]
8 [ 7  7  3 ]
9 [ 2  7  7 ]
10 [ 1  2  7 ]
11 [ 0  1  2 ]
12 [ 2  0  1 ]
13 [ 3  2  0 ]
14 [ 3  3  2 ]
15 [ 5  3  3 ]
16 [ 0  5  3 ]
17 [ 3  0  5 ]
18 [ 0  3  0 ]
19 [ 0  0  3 ]
20 [ 3  0  0 ]
21 [ 3  3  0 ]
22 [ 3  3  3 ]
23 [ 6  3  3 ]
24 [ 1  6  3 ]
25 [ 4  1  6 ]
26 [ 2  4  1 ]
27 [ 3  2  4 ]
28 [ 7  3  2 ]
29 [ 1  7  3 ]
30 [ 4  1  7 ]
31 [ 3  4  1 ]
32 [ 4  3  4 ]
33 [ 0  4  3 ]
34 [ 3  0  4 ]
35 [ 7  3  0 ]
36 [ 7  7  3 ]
37 [ 2  7  7 ]
38 [ 1  2  7 ]
39 [ 0  1  2 ]
40 [ 2  0  1 ]
41 [ 3  2  0 ]
42 [ 3  3  2 ]
43 [ 5  3  3 ]
44 [ 0  5  3 ]
45 [ 3  0  5 ]
46 [ 0  3  0 ]
47 [ 0  0  3 ]
48 [ 3  0  0 ]
49 [ 3  3  0 ]
50 [ 3  3  3 ]
51 [ 6  3  3 ]
52 [ 1  6  3 ]
53 [ 4  1  6 ]
54 [ 2  4  1 ]
55 [ 3  2  4 ]
56 [ 7  3  2 ]
57 [ 1  7  3 ]
58 [ 4  1  7 ]
59 [ 3  4  1 ]
60 [ 4  3  4 ]
61 [ 0  4  3 ]
62 [ 3  0  4 ]
63 [ 7  3  0 ]
64 [ 7  7  3 ]
65 [ 2  7  7 ]
66 [ 1  2  7 ]
67 [ 0  1  2 ]
68 [ 2  0  1 ]
69 [ 3  2  0 ]
70 [ 3  3  2 ]
71 [ 5  3  3 ]
72 [ 0  5  3 ]
73 [ 3  0  5 ]
74 [ 0  3  0 ]
75 [ 0  0  3 ]
76 [ 3  0  0 ]
77 [ 3  3  0 ]
78 [ 3  3  3 ]
79 [ 6  3  3 ]
```

Out[21]: **done**

In [ ]:

T400 Twisted GFSR

based on $GFSR(25, x^{25} + x^{11} + 1)$

n=25 words x w=16 bits = 400 bits, twisting vector $\mathbf{a} = \mathbf{0xA875}$ (a 16 bits vector)

twisting matrix $A = \begin{bmatrix} 0_{15x1} & I_{15x15} \\ \multicolumn{2}{c}{\mathbf{a_{1x16}}} \end{bmatrix}$ (a 16x16 bits array)

if we let $x = [x_0 \dots x_{w-2} x_{w-1}]$ then the block multiplication is

$$[x_0 \dots x_{w-2} \mid x_{w-1}] \, . \, \left[\begin{array}{c|c} 0_{15x1} & I_{15x15} \\ \hline a_0 & a_1 \dots a_{15} \end{array} \right] = [\; x_{w-1} . a_0 \quad x_0 + x_{w-1} * a_1 \dots x_{w-2} + x_{w-1} * a_{16} \;]$$
$$= [\, x_{w-1} . \mathbf{a}_{1x16} \oplus shiftright(\mathbf{x}) \,]$$

The form of A is dictated by the necessity to make it simple to multiply by it :

$$\mathbf{x} . \mathbf{A} \; = \; if \; (x_{w-1} \; = \; 0) \; then \; shiftright(\mathbf{x}) \; else \; shiftright(\mathbf{x}) \oplus \mathbf{a}$$

(Matsumoto, Kurita, 1992) Theorem : if $\varphi_A(x)$ is the characteristic polynomial of the $w \times w$ bits matrix A and $\varphi_A(t^n + t^m)$ is of degree nw and is primitive then the period of :

$$x_{l+n} = x_{l+m} \oplus x_l . A$$

is $2^{nw} - 1$.

This generator returns the random floats $\frac{x}{2^{16}}$

In [26]:
```
a_15_ident:diagmatrix(15,1)$
```

Out[26]:

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

In [27]:
```
a_15_zero_row:[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
```

Out[27]: $[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$

In [28]:
```
a_15_zero_col:transpose(a_15_zero_row)
```
Out[28]:
$$\begin{pmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{pmatrix}$$

In [29]:
```
a_16_zero_row:addcol(matrix(a_15_zero_row),matrix([0]))
```
Out[29]:
$$\begin{pmatrix}0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{pmatrix}$$

In [ ]:

In [30]:
```
a_16_vector_a:[1,0,1,0,1,0,0,0,0,1,1,1,0,1,0,1]
```
Out[30]:
$$[1,0,1,0,1,0,0,0,0,1,1,1,0,1,0,1]$$

In [31]:
```
a_15x16:addcol(a_15_zero_col,a_15_ident)$
```
Out[31]:
$$\begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}$$

In [32]:
```
shiftright_16x16:addrow(a_15x16,a_16_zero_row)
```

Out[32]:

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

In [33]:
```
a_16x16:addrow(a_15x16,a_16_vector_a)
```

Out[33]:

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

In [34]:
```
phi:charpoly(a_16x16,t^25+t^11)
```

Out[34]:

$$\left(-t^{25}-t^{11}\right)^{2}$$

$$\Bigg(\left(-t^{25}-t^{11}\right)^{2}\ \Bigg(\left(-t^{25}-t^{11}\right)^{5}\ \Bigg(\left(-t^{25}-t^{11}\right)\ \Bigg(\left(-t^{25}-t^{11}\right)\ \Bigg(\Bigg(\left(-t^{25}-t^{11}\right)^{2}\ \left(-t^{25}-t^{11}+1\right)+1\Bigg)$$

$$-1$$

In [35]:
```
phi:expand(phi)
```

Out[35]:

$$t^{400}+16\,t^{386}-t^{375}+120\,t^{372}-15\,t^{361}+560\,t^{358}-105\,t^{347}+1820\,t^{344}-455\,t^{333}+4368\,t^{330}-t^{325}$$
$$-13\,t^{311}-3003\,t^{305}+11440\,t^{302}-78\,t^{297}-5005\,t^{291}+12870\,t^{288}-286\,t^{283}-6435\,t^{277}-t^{275}+$$
$$-6435\,t^{263}-11\,t^{261}+8008\,t^{260}-1287\,t^{255}-t^{250}-5005\,t^{249}-55\,t^{247}+4368\,t^{246}-1716\,t^{241}-10$$
$$t^{233}+1820\,t^{232}-1716\,t^{227}-t^{225}-45\,t^{222}-1365\,t^{221}-330\,t^{219}+560\,t^{218}-1287\,t^{213}-9\,t^{211}-12$$
$$t^{205}+120\,t^{204}-715\,t^{199}-36\,t^{197}-210\,t^{194}-105\,t^{193}-462\,t^{191}+16\,t^{190}-286\,t^{185}-84\,t^{183}-252$$
$$+t^{176}-78\,t^{171}-126\,t^{169}-210\,t^{166}-t^{165}-165\,t^{163}-13\,t^{157}-126\,t^{155}-120\,t^{152}-55\,t^{149}-t^{143}-$$
$$t^{135}-36\,t^{127}-10\,t^{124}-t^{121}-9\,t^{113}-t^{110}-t^{100}-t^{99}-4\,t^{86}-6\,t^{72}-4\,t^{58}-t^{50}-t^{44}-$$

In [36]:
```
gf_primitive_poly_p(phi,2)
```
Out[36]: **true**

In [37]:
```
hipow(phi,t)
```
Out[37]: 400

In [46]:
```
x:matrix([1,0,1,0,1,1,1,1,0,1,0,1,0,0,0,1])
```
Out[46]: $\begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$

In [47]:
```
a_16_vector_a_m:matrix(a_16_vector_a)
```
Out[47]: $\begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$

In [48]:
```
mod(x . a_16x16,2)
```
Out[48]: $\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix}$

In [60]:
```
if (x[1][16] = 0) then (x . shiftright_16x16) else mod(x . shiftright_16x16 + a_16_vector_a_
m,2)
```
Out[60]: $\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix}$

In [ ]:
```
```

T800 TGFSR A twisted GFSR

25 words x 32 bits = 800 bits

based on $LGFSR(25, x^{25} + x^{18} + 1)$ twisting vector $\mathbf{a} = \mathbf{0x8ebfd028}$

In [13]:
```
f:x^25+x^18+1
```
Out[13]: $x^{25} + x^{18} + 1$

In [14]:
```
gf_primitive_poly_p(f,2)
```
Out[14]: **true**

In [15]:
```
a:diagmatrix(31,1)$
```
Out[15]:

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

In [16]:
```
acol1:transpose([0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0])$
```

Out[16]:

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

In [20]:
```
arow32:[1,0,0,0,1,1,1,0,1,0,1,1,1,1,1,1,1,1,0,1,0,0,0,0,0,0,1,0,1,0,0,0]$
```

Out[20]: $[1,0,0,0,1,1,1,0,1,0,1,1,1,1,1,1,1,1,0,1,0,0,0,0,0,0,1,0,1,0,0,0]$

In [18]:
```
a:addcol(acol1,a)$
```

Out[18]:

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

In [21]:
```
a:addrow(a,arow32)
```

Out[21]:

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

In [ ]:
```
d:charpoly(a,t^25+t^18)
```

In [ ]:

In [1]:
```
ident_4x4:ident(4);
```
Out[1]:
$$\begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{pmatrix}$$

In [2]:
```
zero_4x1:transpose([0,0,0,0]);
```
Out[2]:
$$\begin{pmatrix}0\\ 0\\ 0\\ 0\end{pmatrix}$$

In [3]:
```
vec_a:[1,1,0,1,1];
```
Out[3]: $[1,1,0,1,1]$

In [4]:
```
m:ident_4x4;
```
Out[4]:
$$\begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{pmatrix}$$

In [5]:
```
m:addcol(zero_4x1,m);
```
Out[5]:
$$\begin{pmatrix}0 & 1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 1\end{pmatrix}$$

In [6]:
```
m:addrow(m,vec_a);
```
Out[6]:
$$\begin{pmatrix}0 & 1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 1\\ 1 & 1 & 0 & 1 & 1\end{pmatrix}$$

In [9]:
```
gf_primitive_poly(2,5);
```
Out[9]: $x^5 + x^2 + 1$

In [10]:
```
m;
```
Out[10]:
$$\begin{pmatrix}0 & 1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 1\\ 1 & 1 & 0 & 1 & 1\end{pmatrix}$$

In [14]:
```
phi:charpoly(m,t^5+t^3);
```

Out[14]: $\left(-t^5-t^3\right)\left(\left(\left(-t^5-t^3\right)\left(-t^5-t^3+1\right)-1\right)\left(-t^5-t^3\right)^2-1\right)+1$

In [15]:
```
phi2:expand(phi);
```

Out[15]: $-t^{25}-5\,t^{23}-10\,t^{21}+t^{20}-10\,t^{19}+4\,t^{18}-5\,t^{17}+6\,t^{16}+4\,t^{14}+3\,t^{13}+t^{12}+3\,t^{11}$ -

In [16]:
```
gf_primitive_poly_p(phi,2);
```

Out[16]: **true**

In [17]:
```
ident_5x5:ident(5);
```

Out[17]: $\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$

In [18]:
```
for i:1 thru 100 do if mod(m^^i,2) = ident_5x5 then print("Wow ",i);
```

```
Wow  31
Wow  62
Wow  93
```

Out[18]: **done**

In [ ]:

In [ ]:

Tempering

The new very long period Twisted GFSR can have not so good statistical properties.
Matsumoto and Kurita developed a shuffling of the bits that can help with this.

For T800 they resubmitted a TT800 version (Tempered Twisted GFSR).
The Marsenne Twister MT19337 is instead already equipped with a similar pattern.

T800 twisting :

```
#include <stdio.h>
#include <unistd.h>



int main(int argc, char* argv[])
{
unsigned int x,y,z;
int s = 7, t=15;
unsigned int a = 0x8ebfd028 ,b = 0x2b5b2500 ,c = 0xdb8b0000;

  while (1) {
    if (read(0,&x,4) != 4) _exit(1);
    y = x ^ ((x<<s) & b);
    z = y ^ ((y<<t)& c);
    write(1,&z,4);
  }
}
```

MT19337 twisting :

```
#include <stdio.h>
#include <unistd.h>



int main(int argc, char* argv[])
{
unsigned int x,y,z;
int u = 11, s = 7, t=15, l = 18;
unsigned int a = 0x9908b0df ,b = 0x9d2c5680 ,c = 0xefc60000;

  while (1) {
    if (read(0,&x,4) != 4) _exit(1);
    y = x ^ (x>>u) ;
    y = y ^ ((y<<s)& b);
    y = y ^ ((y<<t)& c);
    z = y ^ (y>>l) ;
    write(1,&z,4);
  }
}
```

```
In [ ]:
```

Design an MT

In [1]:
```
load("bitwise");
```
Out[1]: /usr/local/share/maxima/5.41.0/share/contrib/bitwise/bitwise.lisp

In [2]:
```
seed:matrix([27,17,21,5,30,14,16]);
```
Out[2]: $\begin{pmatrix} 27 & 17 & 21 & 5 & 30 & 14 & 16 \end{pmatrix}$

In [4]:
```
n:matrix_size(seed)[2];
```
Out[4]: 7

In [5]:
```
ident_4x4:ident(4);
```
Out[5]: $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

In [6]:
```
zero_4x1:transpose([0,0,0,0]);
```
Out[6]: $\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$

In [7]:
```
matrix_A:ident_4x4;
```
Out[7]: $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

In [8]:
```
matrix_A:addcol(zero_4x1,matrix_A)$
```
Out[8]: $\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$

In [9]:
```
vec_a:[1,0,1,1,0]$
```
Out[9]: $[1,0,1,1,0]$

In [10]:
```
matrix_A:addrow(matrix_A,vec_a)$
```
Out[10]: $\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix}$

In [11]:
```
matrix_A[5]:vec_a;
```
Out[11]: $[1,0,1,1,0]$

In [12]:
```
m:2;
```
Out[12]: 2

In [17]:
```
bit_and(seed[1][m],16);
```
Out[17]: 16

In [18]:
```
bit_rsh(%,4);
```
Out[18]: 1

In [19]:
```
bottom_tap:[ bit_rsh(bit_and(seed[1][m],16),4),
             bit_rsh(bit_and(seed[1][m],8),3),
             bit_rsh(bit_and(seed[1][m],4),2),
             bit_rsh(bit_and(seed[1][m],2),1),
                     bit_and(seed[1][m],1)];
```
Out[19]: $[1,0,0,0,1]$

In [20]:
```
top_tap:[   bit_rsh(bit_and(seed[1][n],16),4),
            bit_rsh(bit_and(seed[1][n-1],8),3),
            bit_rsh(bit_and(seed[1][n-1],4),2),
            bit_rsh(bit_and(seed[1][n-1],2),1),
                    bit_and(seed[1][n-1],1)];
```
Out[20]: $[1,1,1,1,0]$

In [21]:
```
bottom_tap:matrix(bottom_tap);
```
Out[21]: $\begin{pmatrix}1 & 0 & 0 & 0 & 1\end{pmatrix}$

In [22]:
```
new_bits:mod(bottom_tap + top_tap  . matrix_A,2);
```
Out[22]: $\begin{pmatrix}1 & 1 & 1 & 1 & 0\end{pmatrix}$

In [23]:
```
new_numb:mod(new_bits[1][5]*16+
             new_bits[1][4]*8+
            new_bits[1][3]*4+
            new_bits[1][2]*2+
            new_bits[1][1],32);
```
Out[23]: 15

In [24]:
```
shr7:matrix([0,1,0,0,0,0,0],
             [0,0,1,0,0,0,0],
             [0,0,0,1,0,0,0],
             [0,0,0,0,1,0,0],
             [0,0,0,0,0,1,0],
             [0,0,0,0,0,0,1],
            [0,0,0,0,0,0,0]);
```
Out[24]:
$$\begin{pmatrix}0 & 1 & 0 & 0 & 0 & 0 & 0\\0 & 0 & 1 & 0 & 0 & 0 & 0\\0 & 0 & 0 & 1 & 0 & 0 & 0\\0 & 0 & 0 & 0 & 1 & 0 & 0\\0 & 0 & 0 & 0 & 0 & 1 & 0\\0 & 0 & 0 & 0 & 0 & 0 & 1\\0 & 0 & 0 & 0 & 0 & 0 & 0\end{pmatrix}$$

In [25]:
```
seed:seed . shr7;
```
Out[25]: $\begin{pmatrix}0 & 27 & 17 & 21 & 5 & 30 & 14\end{pmatrix}$

In [26]:
```
seed[1][1]:new_numb;
```
Out[26]: 15

In [27]:
```
printf(true,"~5,'0b",new_numb);
```

```
01111
```

Out[27]: **false**