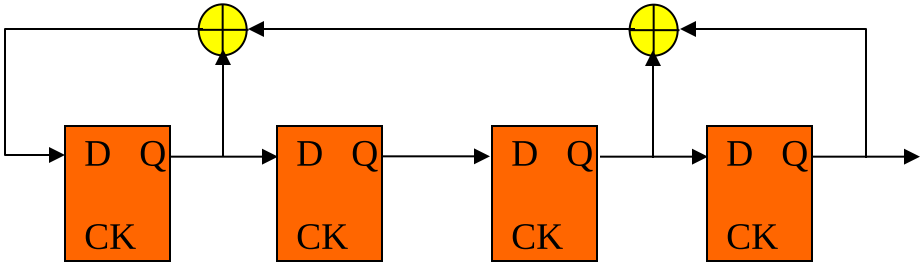
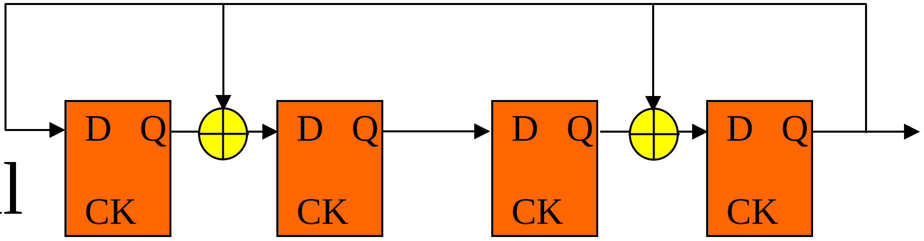
Linear Feedback Shift Registers (LFSRs)

- Efficient design for Test Pattern Generators & Output Response Analyzers (also used in CRC)
  - FFs plus a few XOR gates
  - better than counter
    - fewer gates
    - higher clock frequency
- Two types of LFSRs
  - External Feedback
  - Internal Feedback
    - higher clock frequency
- Characteristic polynomial
  - defined by XOR positions
  - $P(x) = x^4 + x^3 + x + 1$ in both examples

External Feedback LFSR



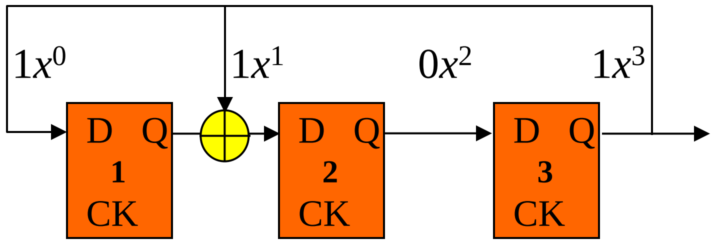
Internal Feedback LFSR



LFSRs (cont)

Characteristic polynomial of LFSR

- n = # of FFs = degree of polynomial
- XOR feedback connection to FF $i \Leftrightarrow$ coefficient of x^i
  - coefficient = 0 if no connection
  - coefficient = 1 if connection
  - coefficients always included in characteristic polynomial:
    - x^n (degree of polynomial & primary feedback)
    - $x^0 = 1$ (principle input to shift register)
- Note: state of the LFSR $\Leftrightarrow$ polynomial of degree n-1
- Example: $P(x) = x^3 + x + 1$

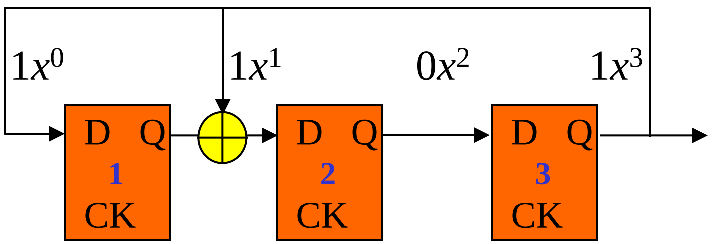


LSFRs (cont)

- An LFSR generates periodic sequence
  - must start in a non-zero state,
- The maximum-length of an LFSR sequence is 2^n -1
  - does not generate all 0s pattern (gets stuck in that state)
- The characteristic polynomial of an LFSR generating a maximum-length sequence is a ***primitive polynomial***
- A maximum-length sequence is ***pseudo-random***:
  - number of 1s = number of 0s + 1
  - same number of runs of consectuive 0s and 1s
  - 1/2 of the runs have length 1
  - 1/4 of the runs have length 2
  - … (as long as fractions result in integral numbers of runs)

LFSRs (cont)

- Example: Characteristic polynomial is $P(x) = x^3 + x + 1$
- Beginning at all 1s state
  - 7 clock cycles to repeat
  - maximal length = 2^n-1
  - polynomial is primitive
- Properties:
  - four 1s and three 0s
  - 4 runs:
    - 2 runs of length 1 (one 0 & one 1)
    - 1 run of length 2 (0s)
    - 1 run of length 3 (1s)
- Note: external & internal LFSRs with same primitive polynomial do not generate same sequence (only same length)



1	1	1	1
1	0	1	2
1	0	0	3
0	1	0	4
0	0	1	5
1	1	0	6
0	1	1	7
1	1	1	

LFSRs (cont)

- Reciprocal polynomial, $P^*(x)$
  - $P^*(x) = x^n P(1/x)$
    - example: $P(x) = x^3 + x + 1$
    - then: $P^*(x) = x^3 (x^{-3} + x^{-1} + 1) = 1 + x^2 + x^3 = x^3 + x^2 + 1$
  - if $P(x)$ is primitive, $P^*(x)$ is also primitive
    - same for non-primitive polynomials
- Polynomial arithmetic
  - modulo-2 ($x^n + x^n = x^n - x^n = 0$)

Addition/Subtraction

$(x^5 + x^2 + 1) + (x^4 + x^2)$

$$\begin{array}{lllll} x^5 & & x^2 & 1 \\ + & x^4 & x^2 & \\ \hline x^5 & x^4 & & 1 \end{array}$$

$= x^5 + x^4 + 1$

Multiplication

$(x^2 + x + 1) \times (x^2 + 1)$

$$\begin{array}{r} x^2 + x + 1 \\ \times \quad x^2 + 1 \\ \hline x^2 + x + 1 \\ x^4 + x^3 + x^2 \qquad\qquad \\ \hline \end{array}$$

$= x^4 + x^3 + x + 1$

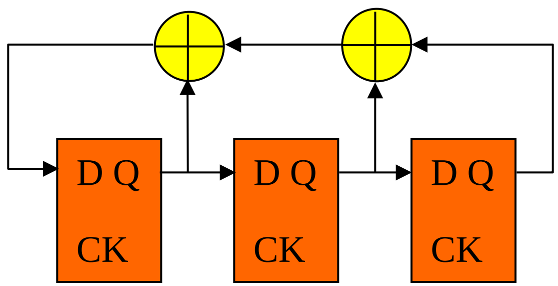
Division

$$\begin{array}{r|l} & x^2 + x + 1 \\ \cline{2-2} x^2 + 1 & x^4 + x^3 + x + 1 \\ & x^4 \quad + x^2 \\ \cline{2-2} & x^3 + x^2 + x + 1 \\ & x^3 \quad\quad + x \\ \cline{2-2} & x^2 + 1 \\ & x^2 + 1 \\ \cline{2-2} & 0 \end{array}$$

LFSRs (cont)

- Non-primitive polynomials produce sequences < 2^n-1
  - Typically primitive polys desired for TPGs & ORAs
- Example of non-primitive polynomial
  - $P(x) = x^3 + x^2 + x + 1$

External Feedback LFSR



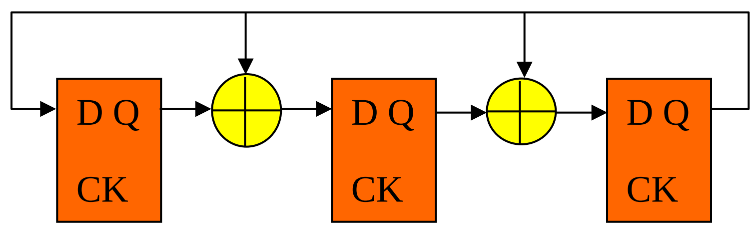
0 0 0
0 0 0

1 1 1
1 1 1

1 0 0
1 1 0
0 1 1
0 0 1
1 0 0

0 1 0
1 0 1
0 1 0

Internal Feedback LFSR



0 0 0
0 0 0

1 1 1
1 0 0
0 1 0
0 0 1
1 1 1

1 1 0
0 1 1
1 1 0

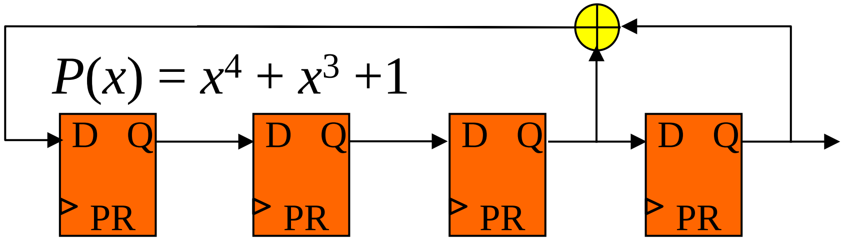
1 0 1
1 0 1

LFSRs (cont)

- Primitive polynomials with minimum # of XORs

Degree (n)	Polynomial
2,3,4,6,7,15,22	$x^n + x + 1$
5,11,21,29	$x^n + x^2 + 1$
8,19	$x^n + x^6 + x^5 + x + 1$
9	$x^n + x^4 + 1$
10,17,20,25,28	$x^{\mathrm{n}} + x^3 + 1$
12	$x^n + x^7 + x^4 + x^3 + 1$
13,24	$x^n + x^4 + x^3 + x + 1$
14	$x^n + x^{12} + x^{11} + x + 1$
16	$x^n + x^5 + x^3 + x^2 + 1$
18	$x^n + x^7 + 1$
23	$x^n + x^5 + 1$
26,27	$x^n + x^8 + x^7 + x + 1$
30	$x^n + x^{16} + x^{15} + x + 1$

HDL descriptions




```
-- VHDL model
entity LFSR is
port (CLK,PR: in bit;
        Q:  buffer bit_vector(4 downto 1));
end entity LFSR;
architecture RTL of LFSR is
begin
process (CLK)
begin
if (CLK'event and CLK = '1') then
     if (PR = '1') then
        Q <= "1111";
    else
          for K in 2 to 4 loop
               Q(K) <= Q(K-1);
          end loop;
     end if;
     Q(1) <= Q(4) xor Q(3);
end if;
end process;
end architecture RTL;
```

```
// Verilog model
module LFSR (CLK, PR, Q);
input CLK, PR;
output [4:1] Q;
/* end of "entity" portion and
     beginning of "architecture" portion */
reg Q;
integer K;
always @ (posedge CLK) begin
     if (PR == 1)
          Q = 4'b1111;
     else begin
          for (K=2 ; K < 5 ; K=K+1) begin
               Q[K] = Q[K-1];
          end
          Q[1] = Q[4] ^ Q[3];
      end
end
endmodule
```