Str. Polithenicii 1
500024 – Brașov
tel.: (+40) 268.474.718 | fax: (+40) 268.474.718
f-iesc@unitbv.ro | www.unitbv.ro/fiesc

Limbaje de Descriere Hardware

Cerinta Tema : LFSR Counter with Loadable Input

Cadru didactic: Prof. Nicula Dan

Student:Chelemen Antonia

Specializare: Calculatoare

An: 3, Grupa 4FL791

Features

- Parameterized word length

- Loadable counter registers

- High speed, area-efficient

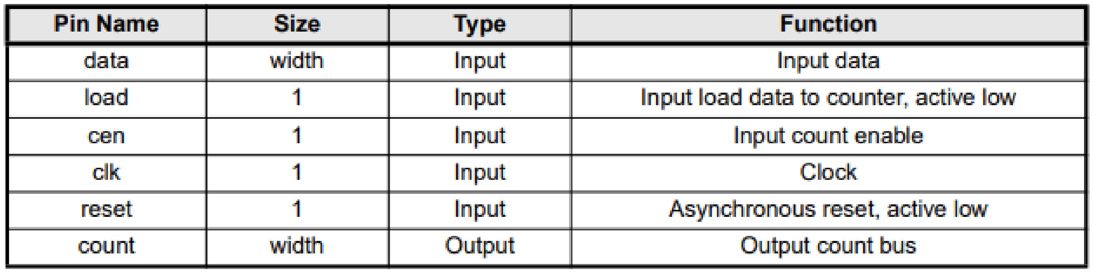
- Asynchronous reset

- Terminal count

Description DW03_lfsr_load is a parameterized word length up counter with loadable data input. DW03_lfsr_load implements a counter as LFSR (linear feedback shift register) which also acts as a pseudorandom counter constructed as primitive characteristic polynomials. The shift register is fed back from two or more taps. The number of taps does not increase with a large counter width. An LFSR counter runs faster than a binary counter in synchronous systems because the first bit is calculated and the remaining bits are shifted. DW03_lfsr_load can be used in built-in test circuitry for VLSI chips and multiple-input signature registers. Counter Function The data input bus, data, ranges from width-1 to 0. When the input signal, load, is LOW, data is exclusive ORd with count. When load is HIGH, the counter operates as a normal LFSR up counter.

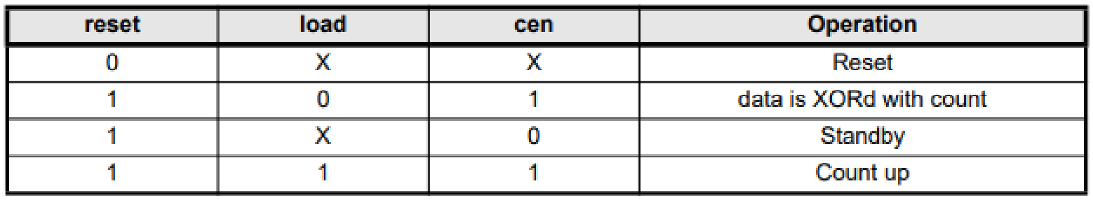
When the count enable pin, cen, is HIGH, the counter is active. When cen is LOW, the counter is disabled and count remains at the same value. reset, active low, is an asynchronous reset signal. When reset is LOW, the counter output is “00...00”. When reset is HIGH, the counter operates normally. count is the output port of pseudorandom binary sequences, ranging from width-1 to 0. Refer to Table 2 in the Sequential Overview section for a listing of the primitive polynomials. A value of 2width-2 (“11...11”) is an illegal state; therefore, the counter stops at “11...11”.

Table 1 - Pin Description



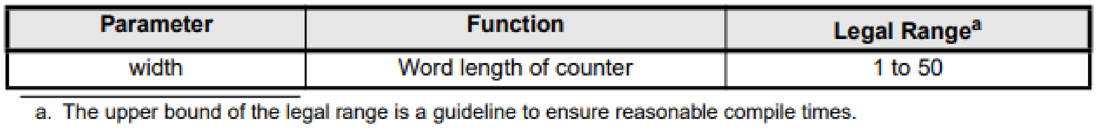
Pin Name	Size	Type	Function
data	width	Input	Input data
load	1	Input	Input load data to counter, active low
cen	1	Input	Input count enable
clk	1	Input	Clock
reset	1	Input	Asynchronous reset, active low
count	width	Output	Output count bus

Table 2 - Counter Operation Truth Table



reset	load	cen	Operation
0	X	X	Reset
1	0	1	data is XORd with count
1	X	0	Standby
1	1	1	Count up

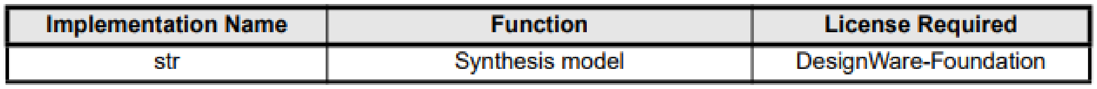
Table 3 - Parameter Description



Parameter	Function	Legal Range[a]
width	Word length of counter	1 to 50

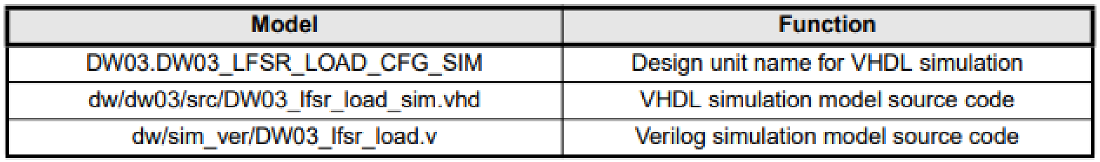
a. The upper bound of the legal range is a guideline to ensure reasonable compile times.

Table 4 - Synthesis Implementations



Implementation Name	Function	License Required
str	Synthesis model	DesignWare-Foundation

Table 5 - Simulation Models



Model	Function
DW03.DW03_LFSR_LOAD_CFG_SIM	Design unit name for VHDL simulation
dw/dw03/src/DW03_lfsr_load_sim.vhd	VHDL simulation model source code
dw/sim_ver/DW03_lfsr_load.v	Verilog simulation model source code

1. Descrierea Verilog a unui modul care modelează comportamental circuitul.

```
// Universitatea Transilvania din Brasov
// Departamentul de Electronica si Calculatoare
// Proiect    : Tema 1
// Modul      : LFSR Counter with Loadable Input
// Autor      : Chelemen Antonia
// Data       : 04.04.2022
//-----------------------------------------------------------------------------
// Descriere  : Linear feedback shift register counter with loadable input
```

```
//
//  rst load cen  | Operation
// --------- --------------------------------------------------
//   0    x   x   | Reset
//   1    0   1   | data is XORd with count
//   1    x   0   | Standby
//   1    1   1   | Count up
//--------------------------------------------------------------------------

module DW03_lfsr_load #(
parameter WIDTH          = 'd4          // numar de biti ai numaratorului
) (
input      [WIDTH -1:0]   data       , // data de load
input                   clk        , // semnal de clock
input                   rst_n      , // reset date
input                   cen        , // count enable
input                   load_n     , // incarcare date

output reg [WIDTH -1:0]   count       // iesire counter

);

always @(posedge clk or negedge rst_n) begin
   if(~rst_n)  count <= 'd1; else
   if(load_n && cen) count <= {count[WIDTH-1] ^ count[0], count[WIDTH-1:1]}; else
//xor intre primul si ultimul bit
   if(cen) count <= count ^ data; else // daca se face load, se face xor intre count si data
   count <= count; //daca counter enable e 0 se mentine valoarea lui count

end


endmodule
```

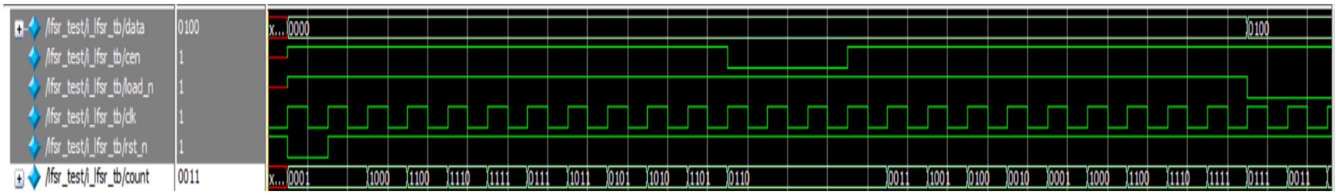
2. Descrierea Verilog a unui modul care generează un set de vectori de test pentru circuit (testbench)

```
// Universitatea Transilvania din Brasov
// Departamentul de Electronica si Calculatoare
// Proiect     : Tema 1
// Modul       : LFSR Counter
// Autor       : Chelemen Antonia
// Data        : 04.04.2021
//----------------------------------------------------------------------------
// Descriere   : LFSR Counter with Loadable Input
//----------------------------------------------------------------------------


module lfsr_tb #(
parameter WIDTH            = 'd4         // numar de biti ai operanzilor
) (

output    reg  [WIDTH -1:0]   data        ,
output    reg                 cen         ,
output    reg                 load_n      ,
output    reg              clk         , // ceas
output    reg              rst_n,
input     [WIDTH -1:0]            count
);

initial
begin
  clk = 1'b0;            // valoare initiala 0
  forever #10            // valoare complementata la fiecare semi-perioada
    clk = ~clk;
end

initial begin
  rst_n     <= 1'b1;
  cen       <= 1'bx;
  load_n    <= 'bx;
  data      <= {WIDTH{1'bx}};
  @(posedge clk);
  rst_n     <=1'b0;
  // initializeaza numaratorul la 0, la activarea reset asincron
  @(negedge rst_n);
  load_n    <= 1'b1;
  cen       <= 1'b1;
  data      <=  'd0;
  @(posedge clk);
  rst_n     <= 1'b1;

  repeat (10) @(posedge clk);

  cen    <= 1'b0;
  repeat (3) @(posedge clk);
  cen    <= 1'b1;

  repeat (10) @(posedge clk); //al doilea grafic

  load_n <= 1'b0;
  data   <= 'd4; //incarcare date
  repeat (5) @(posedge clk);
  load_n <= 1'b1;
  repeat (5) @(posedge clk);
  cen    <= 1'b0;
  @(posedge clk);
  cen    <= 1'b1;
  repeat (5) @(posedge clk);
  $stop;
end

endmodule // adder_tb
```

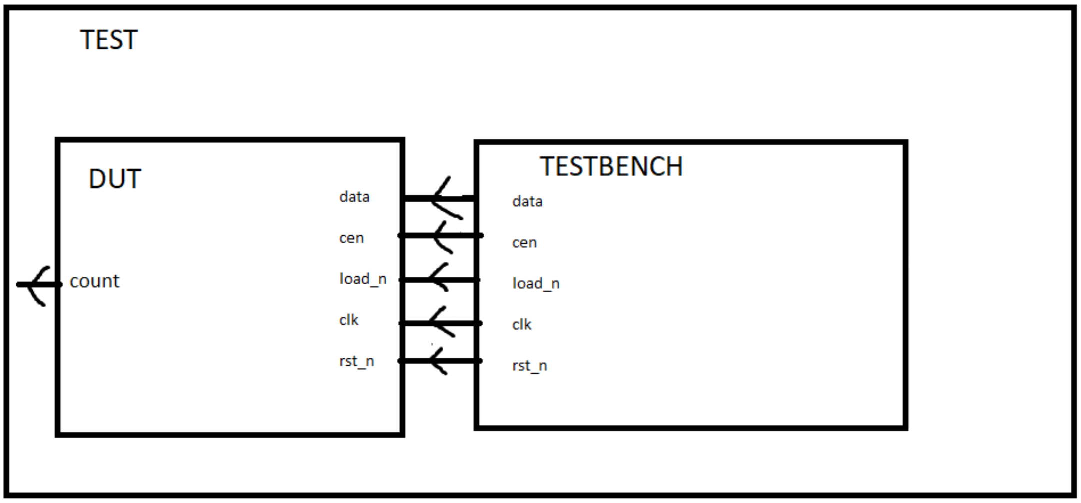
3. Descrierea Verilog a unui mediu de simulare (test) în care se vor instanția două componente:

```
// Universitatea Transilvania din Brasov
// Departamentul de Electronica si Calculatoare
// Proiect     : Tema 1
// Modul       : LFSR Counter
// Autor      : Chelemen Antonia
// Data       : 04.04.2021
//-----------------------------------------------------------------------------
// Descriere  : LFSR Counter with Loadable Data Input
//-----------------------------------------------------------------------------


module lfsr_test;
localparam            WIDTH        = 'd4;       // numar de biti ai numaratorului

wire  [WIDTH   -1:0]      data          ;  // data
wire                      cen           ;  // counter enable
wire                      load_n        ;  // load activ in 0
wire                      clk           ;  // clock
wire                      rst_n         ;  // reset activ in 0
wire  [WIDTH   -1:0]      count         ;  // output


DW03_lfsr_load #(
.WIDTH   (WIDTH)
) i_DW03_lfsr_load_DUT (
.clk          (clk       ),
.rst_n        (rst_n     ),
.load_n       (load_n    ),
.cen          (cen       ),
.data         (data      ),
.count        (count     )
);

lfsr_tb i_lfsr_tb (
.clk          (clk     ),
.rst_n        (rst_n   ),
.load_n       (load_n  ),
.cen          (cen     ),
.data         (data    ),
.count        (count   )
);


Endmodule
```

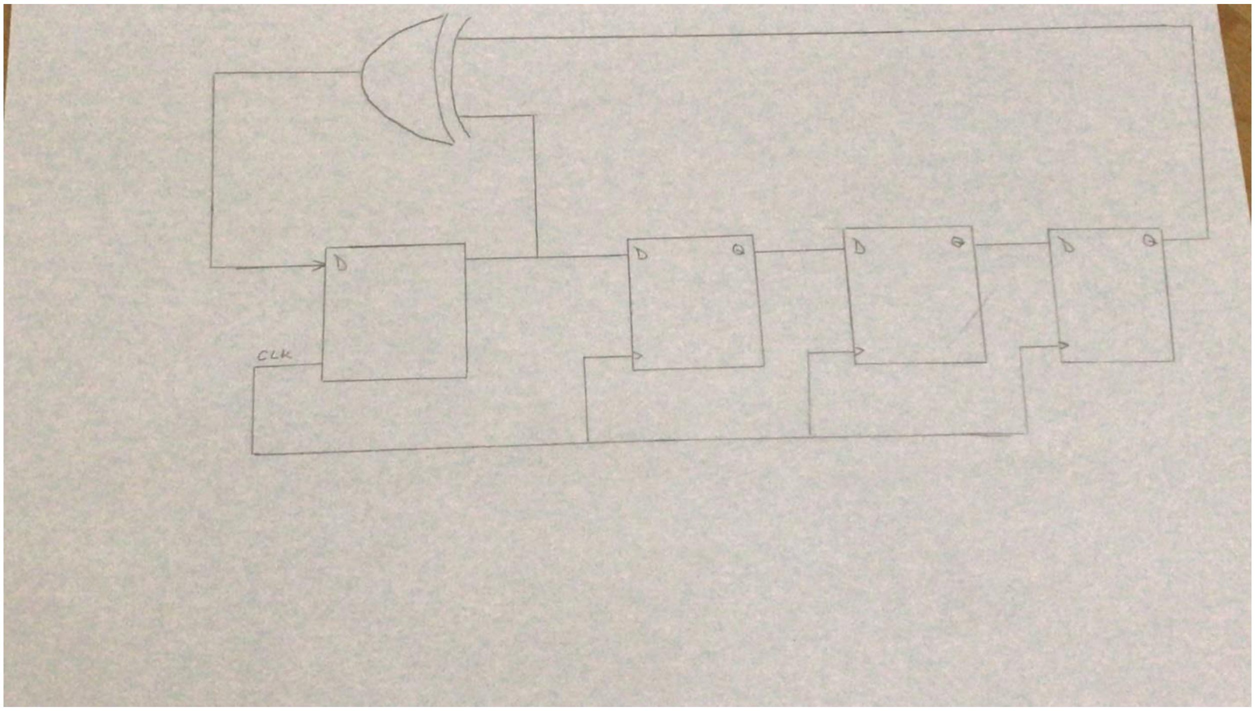
Reprezentare grafica:



```
/lfsr_test/i_lfsr_tb/data     0100
/lfsr_test/i_lfsr_tb/cen      1
/lfsr_test/i_lfsr_tb/load_n   1
/lfsr_test/i_lfsr_tb/clk      1
/lfsr_test/i_lfsr_tb/rst_n    1
/lfsr_test/i_lfsr_tb/count    0011
```



4. Descrierea Verilog a unui modul care modelează structural circuitul. În structură se vor instanția circuite logice elementare (porți logice, multiplexoare, decodificatoare, bistabile). Desenul structurii propuse. Se va considera parametrul width=4.


D
D
Q
D
Q
D
Q
CLK