module ssg2\_f(

data,

en, clk, rst, make, // enable, clock, reset.

shrinkout // intermediate output(shrinkout), output(key\_seq).

);

output reg [8:0]shrinkout;

//output [31:0]key\_seq;

//output [31:0]new\_seq;

input [4:0] data;

input en, clk, rst, make;

reg [4:0] out;

reg [30:0]interm\_out;

reg [61:0] interout;

//reg [15:0]shrink\_rev;

//wire [15:0]shrink\_new\_seq;

reg [7:0] rev = 8'b00000000;

integer i='d0,k='d0,j,m;

/\*initial

begin

out = data;

end\*/

always @ (posedge clk)

begin

if (rst) begin out <= 5'b0 ; end // active high reset

else if (make == 1'b1) begin out <= data; end

else if (en == 1'b1)begin // operation of LFSR

out[4] <= out[0]; // with initial states

out[3] <= out[4]; // and connection polynomial

out[2] <= out[3];

out[1] <= out[2] ^ out[0];

out[0] <= out[1] ^ out[0];

interm\_out[i] <= out[0]; // obtaining intermediate output

i=i+1'd1;

rev = rev + 1'b1;end

else if (rev[5]== 1'b1)begin

for (j=0;j<62;j=j+3)

begin

if ((interout[j]+interout[j+1] )== 1'b1) begin shrinkout[k] <= interout[j+2]; //shrinking the intermediate output(LFSR output)

k=k+1'd1;end //if pair is- '10' then output- '0' // else if pair is- '01' or '00' then output- 'discarded.

else begin k=k;end

end

rev = 8'b00000000;end

interout <= {interm\_out,interm\_out};

end

//always @ (shrinkout) // from here onwards the program is on modified logic of shrinked output of Self Shrinking Generator.

// begin

// for(m=0;m<16;m=m+1)

// begin

// shrink\_rev[m]<=shrinkout[15-m];

// end

//// end

//assign shrink\_new\_seq = ~(shrink\_rev);

//assign new\_seq = {shrinkout,shrink\_new\_seq};

//assign key\_seq = new\_seq ^ interm\_out;

//ila\_0 my\_ila (

// .clk(clk), // input wire clk

// .probe0({shrinkout, key\_seq, new\_seq}) // input wire [79:0] probe0

//);

endmodule