module primary\_lsfr13 (

input clk,

input reset,

input write,

input pushin,

input [197:0] InitialData13,

output [197:0] rnd1

);

//Linear feedback shift registers

reg [197:0] lfsr13, random\_next1, random\_done1;

//Count for the number of shifts

reg [3:0] count1, count\_next1;

always @ (posedge clk or posedge reset)

begin

if (reset)

begin

lfsr13 <= #1 0;

//case1

//lfsr1 <= 185'h4751245563371bb82b2b5aacd05678a1b17e06c62eb0dace; //An LFSR cannot have an all 0 state, thus reset to 4751245563371bb82b2b5aacd05678a1b17e06c62eb0dace

end

else

begin

if (write)

begin

lfsr13 <= InitialData13;

//case2

//lfsr1 <= 185'h08AAC66E37215874F559A0ACF14362FC0D24CD61E1D5512;

count1 <= 0;

end

else if (pushin)

begin

lfsr13 <= #1 random\_next1;

count1 <= #1 count\_next1;

end

end

end

always @ (\*)

begin

//-----------Combinational code for shift register 1 --> 13 bits ----------//

random\_next1 = lfsr13; //default state stays the same

count\_next1 = count1;

random\_done1 = 0;

random\_next1 = { (lfsr13[187:154]), (lfsr13[197]^lfsr13[153]) ,(lfsr13[196]^lfsr13[152]) ,(lfsr13[195]^lfsr13[151]) ,(lfsr13[194]^lfsr13[150]) ,(lfsr13[193]^lfsr13[149]) ,

(lfsr13[192]^lfsr13[148]) ,(lfsr13[191]^lfsr13[147]) ,(lfsr13[190]^lfsr13[146]) ,(lfsr13[189]^lfsr13[145]) ,(lfsr13[188]^lfsr13[144]) ,

(lfsr13[197]^lfsr13[143]) ,(lfsr13[196]^lfsr13[142]) ,(lfsr13[195]^lfsr13[141]) ,(lfsr13[194]^lfsr13[140]) ,(lfsr13[193]^lfsr13[139]) ,

(lfsr13[192]^lfsr13[138]) ,(lfsr13[191]^lfsr13[137]) ,(lfsr13[190]^lfsr13[136]) ,(lfsr13[189]^lfsr13[135]) ,(lfsr13[188]^lfsr13[134]) ,

(lfsr13[133:97]),

(lfsr13[197]^lfsr13[96]) ,(lfsr13[196]^lfsr13[95]) ,(lfsr13[195]^lfsr13[94]) ,(lfsr13[194]^lfsr13[93]) ,(lfsr13[193]^lfsr13[92]) ,

(lfsr13[192]^lfsr13[91]) ,(lfsr13[191]^lfsr13[90]) ,(lfsr13[190]^lfsr13[89]) ,(lfsr13[189]^lfsr13[88]) ,(lfsr13[188]^lfsr13[87]) ,

(lfsr13[86:0]), (lfsr13[197:188]) };

count\_next1 = count1 + 1;

if (count1 == 1)

begin

count1 = 0;

random\_done1 = lfsr13; //assign the random number to output after 13 shifts

end

//--------------------------------------------End of combination logic for shift register 1----------------------------------//

end

assign rnd1 = lfsr13;

endmodule