`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company: Ratner Surf Designs

// Engineer: James Ratner

//

// Create Date: 11/14/2018 02:46:31 PM

// Design Name:

// Module Name: cntr\_up\_clr\_nb

// Project Name:

// Target Devices:

// Tool Versions:

// Description: Generic n-bit up counter with asynchronous reset.

// When up input not asserted, counter holds states. This

// counter has RCO that works for both up & down counting.

//

// cntr\_up\_clr\_nb #(.n(16)) MY\_CNTR (

// .clk (my\_clk),

// .clr (my\_clr),

// .up (my\_up),

// .ld (my\_ld),

// .D (my\_D),

// .count (my\_count),

// .rco (my\_rco) );

//

// Dependencies:

//

// Revision:

// Revision 1.00 - File Created (11-14-2018)

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

module cntr\_up\_clr\_nb(clk, clr, up, ld, D, count, rco);

input clk, clr, up, ld;

input [n-1:0] D;

output reg [n-1:0] count;

output reg rco;

//- default data-width

parameter n = 8;

always @(posedge clr, posedge clk)

begin

if (clr == 1) // asynch reset

count <= 0;

else if (ld == 1) // load new value

count <= D;

else if (up == 1) // count up (increment)

count <= count + 1;

end

//- handles the RCO, which is direction dependent

always @(count, up)

begin

if ( up == 1 && &count == 1'b1)

rco = 1'b1;

else if (up == 0 && |count == 1'b0)

rco = 1'b1;

else

rco = 1'b0;

end

endmodule