module lr (input clk, input inbits, input reset, output reg detect);

reg[2:0] state;

initial begin

state = 3'b000;

end

always @(posedge clk, posedge reset) begin

if (reset)

state <= 3'b000;

else

begin

case (state)

3'b000:

if (inbits) state <= 3'b001;

else state <= 3'b011;

3'b001:

if (inbits) state <= 3'b010;

else state <= 3'b001;

3'b010:

if (inbits) state <= 3'b011;

else state <= 3'b110;

3'b011:

if (inbits) state <= 3'b100;

else state <= 3'b110;

3'b100:

if (inbits) state <= 3'b100;

else state <= 3'b000;

3'b101:

if (inbits) state <= 3'b011;

else state <= 3'b111;

3'b110:

if (inbits) state <= 3'b110;

else state <= 3'b101;

3'b111:

if (inbits) state <= 3'b100;

else state <= 3'b010;

default:

state <= 3'b000;

endcase

end

always @(state) begin

case (state)

3'b000:

detect <= 3'b000;

3'b001:

detect <= 3'b001;

3'b010:

detect <= 3'b010;

3'b011:

detect <= 3'b011;

3'b100:

detect <= 3'b100;

3'b101:

detect <= 3'b101;

3'b110:

detect <= 3'b110;

3'b111:

detect <= 3'b111;

default:

detect <= 3'b000;

endcase

end

endmodule