perplexity

Please generate a 64-to-1 multiplexer design in Verilog defined as:

module mux64_1(in[63:0], select, out);
//Insert code here
endmodule

Here is a Verilog code implementation for a 64-to-1 multiplexer. The design uses a `case` statement to select one of the 64 input bits based on the 6-bit `select` signal.

```
module mux64_1(
    input [63:0] in,       // 64-bit input
    input [5:0] select,    // 6-bit select signal
    output reg out         // Output
);

    always @(*) begin
        case (select)
            6'd0: out = in[^0];
            6'd1: out = in[^1];
            6'd2: out = in[^2];
            6'd3: out = in[^3];
            6'd4: out = in[^4];
            6'd5: out = in[^5];
            6'd6: out = in[^6];
            6'd7: out = in[^7];
            6'd8: out = in[^8];
            6'd9: out = in[^9];
            6'd10: out = in[^10];
            6'd11: out = in[^11];
            6'd12: out = in[^12];
            6'd13: out = in[^13];
            6'd14: out = in[^14];
            6'd15: out = in[^15];
            6'd16: out = in[^16];
            6'd17: out = in[^17];
            6'd18: out = in[^18];
            6'd19: out = in[^19];
            6'd20: out = in[^20];
            6'd21: out = in[^21];
            6'd22: out = in[^22];
            6'd23: out = in[^23];
            6'd24: out = in[^24];
            6'd25: out = in[^25];
            6'd26: out = in[^26];
            6'd27: out = in[^27];
            6'd28: out = in[^28];
            6'd29: out = in[^29];
            6'd30: out = in[^30];
            6'd31: out = in[^31];
            6'd32: out = in[^32];
            6'd33: out = in[^33];
            6'd34: out = in[^34];
            6'd35: out = in[^35];
            6'd36: out = in[^36];
            6'd37: out = in[^37];
            6'd38: out = in[^38];
            6'd39: out = in[^39];
            6'd40: out = in[^40];
            6'd41: out = in[^41];
            6'd42: out = in[^42];
            6'd43: out = in[^43];
            6'd44: out = in[^44];
            6'd45: out = in[^45];
            6'd46: out = in[^46];
            6'd47: out = in[^47];
            6'd48: out = in[^48];
            6'd49: out = in[^49];
            6'd50: out = in[^50];
            6'd51: out = in[^51];
            6'd52: out = in[^52];
            6'd53: out = in[^53];
            6'd54: out = in[^54];
            6'd55: out = in[^55];
            6'd56: out = in[^56];
            6'd57: out = in[^57];
            6'd58: out = in[^58];
            6'd59: out = in[^59];
            6'd60: out = in[^60];
            6'd61: out = in[^61];
            6'd62: out = in[^62];
            default :out=1'bx;
        endcase
    end

endmodule
```

⁂