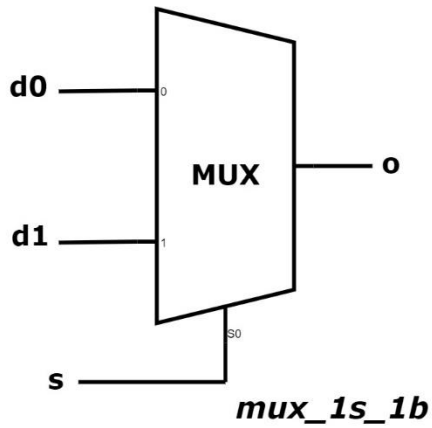
Week 3 - Verilog Module Instantiations

Combinational Always Blocks and Instances in Verilog

P.3.1 Design, using a combinational always block, a 2-to-1 multiplexer having 3 inputs: ***s***, ***d0***, and ***d1***, and having one output, denoted by ***o***.

Solution:

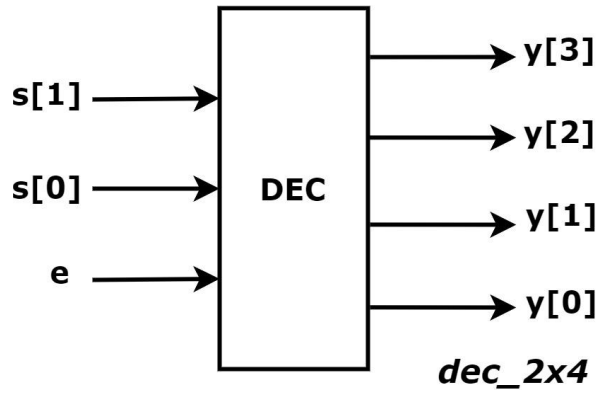
```
module mux_2_to_1 (
    input s,
    input d0,
    input d1,
    output reg o
);
always @(*) begin
if (s) o = d1;
else o = d0
end
endmodule
```



P.3.2 Consider the architecture of a 2-to-4 line decoder, named ***dec_2x4***. The decoder has a 2-bit selection line ***s***, a 4-bit declared ***y*** output, active on 1, and an enable line (***e***) which when set to 1 allows the decoder unit to operate. Construct the module using a combinational always block.

Solution:

```
module dec_2x4 (
    input [1:0] s,
    input e,
    output reg [3:0] y
);
always @(*) begin
case ({e,s})
3'b100 : y = 4'b0001;
3'b101 : y = 4'b0010;
3'b110 : y = 4'b0100;
3'b111 : y = 4'b1000;
3'b0?? : y = 4'b0000;
endcase
end
endmodule
```



P.3.3 Design, using Verilog, an 8-to-1 multiplexer with 2 inputs: ***s*** on 3 bits, ***d*** on 8 bits, and an output ***o***. Construct the hierarchical design of the multiplexer, using instances of the ***mux_1s_1b*** module implemented in ***P.3.1***.

Solution:

```
module mux_8_to_1 (
    input [2:0] s,
    input [7:0] d,
    output o
);
wire w1,w2,w3,w4,w5,w6;
// Layer 1
mux_1s_1b m1 (
.d1(d[7]), .d0(d[6]), .s(s[0]), .o(w1) );
mux_1s_1b m2 (
.d1(d[5]), .d0(d[4]), .s(s[0]), .o(w2) );
```

```
mux_1s_1b m3 (
.d1(d[3]), .d0(d[2]), .s(s[0]), .o(w3) );
mux_1s_1b m4 (
.d1(d[1]), .d0(d[0]), .s(s[0]), .o(w4) );
// Layer 2
mux_1s_1b m5 (
.d1(w1), .d0(w2), .s(s[1]), .o(w5) );
mux_1s_1b m6 (
.d1(w3), .d0(w4), .s(s[1]), .o(w6) );
// Layer 3
mux_1s_1b m7 (
.d1(w5), .d0(w6), .s(s[2]), .o(o) );
endmodule
```

P.3.4 Design, using Verilog, an 8-to-1 multiplexer with 2 inputs: ***s*** on 3 bits, ***d*** on 8 bits, and one output ***o***. Implement the hierarchical design of the multiplexer using instances of the ***mux_1s_1b*** and ***mux_2s_1b*** modules, respectively.

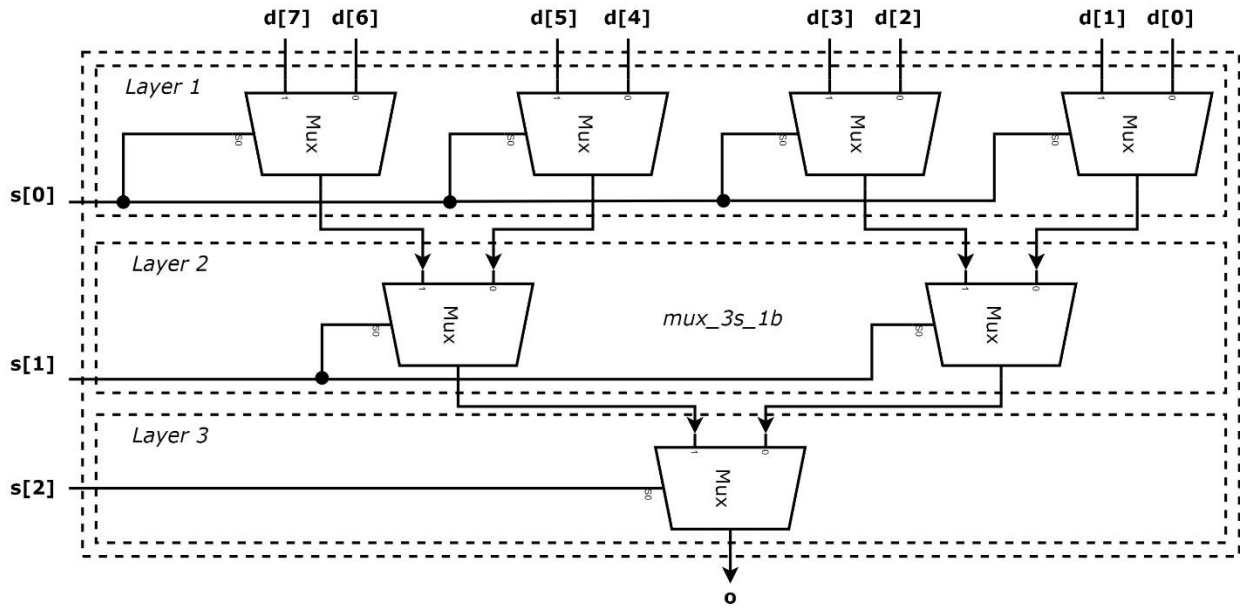
Solution:

```
module mux_2s_1b (
    input [1:0] s,
    input [3:0] d,
    output o
);
assign o = s[1] ? (s[0]?d[3]:d[2]):(s[0]?d[1]:d[0]);
endmodule
```

```
module mux_8_to_1 (
    input [2:0] s,
    input [7:0] d,
    output o        );
wire w1,w2;
// Layer 1
mux_2s_1b m1 (
.d(d[7:4]), .s(s[1:0]), .o(w1) );
mux_2s_1b m2 (
.d(d[3:0]), .s(s[1:0]), .o(w2) );
// Layer 2
mux_1s_1b m3 (
.d1(w1), .d0(w2), .s(s[2]), .o(o) );
endmodule
```

Annexes

P.3.3 Figure



P.3.4 Figure