**#submodule for 2:1 mux**

**module mux2to1(w,s,f);**

**input [1:0]w;**

**input s;**

**output f;**

**always @ (w or s)**

**case(s)**

**1'b0: f = w[0];**

**1'b1: f = w[1];**

**endcase**

**endmodule**

**#submodule for 4:1 mux**

**module mux4to1(w,s,f);**

**input [3:0]w;**

**input [1:0]s;**

**output f;**

**always @ (w,s)**

**case(s)**

**0: f = w[0];**

**1: f = w[1];**

**2: f = w[2];**

**3: f = w[3];**

**default: f = 1'bx;**

**endcase**

**endmodule**

**#main module for 8:1 mux**

**module mux8to1(d,w,s);**

**input [7:0]w;**

**input [2:0]s;**

**output d;**

**wire [5:1]f;**

**mux2to1 m1(f[1],w[1:0],s[0]);**

**mux2to1 m2(f[2],w[3:2],s[0]);**

**mux2to1 m3(f[3],w[5:4],s[0]);**

**mux2to1 m4(f[4],w[7:6],s[0]);**

**mux4to1 m5(d,f[4:1],s[2:1]);**

**endmodule**

**Report**

**A three-variable 8:1 multiplexer consists of 8 inputs going in with an output coming out. We can pass the 8 inputs through four 2:1 multiplexers first. Then, we can send the 4 corresponding outputs through another 4:1 multiplexer to generate the desired output. Here, the inner wires, f, are the outputs of the 2:1 multiplexers and the corresponding inputs of the 4:1 multiplexer. The 8 initial inputs are denoted with ‘w’. The final output is ‘d’. The single selector pin for the 2:1 multiplexer is s0 and the other two selector pins for the single 4:1 multiplexer are s1 and s2. Here, we first write two submodules for the 2:1 and 4:1 multiplexers using the related tokens. Then we put in a main module for the entire 8:1 multiplexer. Finally, we call the submodules for the 2:1 multiplexer four times and the 4:1 multiplexer once using functions, m1 to m4, to make them work via Verilog.**