```
1   // Name:  Ron Kalin, Date: 5-31-24, Design: Lesson3A3P2: 7-seg display 2 digits
2   // Group: Ron Kalin/Lamin Jammeh _____
3   module SevenSegDisplay2Digits (output [6:0] d1,d0, input [3:0] V);
4        wire [3:0] A;
5        wire [3:0] muxout;
6        wire z;
7   //instantiate submodules, cannot use if statements to instantiate
8   comp4bit COMP9 (z, V, gr); //if V>9, gr=1, else gr=0
9   CircuitA CIRA (V, A); //input V output A, A=V if V<=9, else A=V-10
10  mux2to1_4bit MUX (V, A, z, muxout);//gr=0, select V, else select A
11  //assign z1 = (gr==1) ? 1 : 0; //if gr=1 then z1=1, else z1=0
12  Seven_Seg_Disp SEV1 (d1,z); //d1 output, z1 input (either 1 or 0), blank =4'b1111
13  Seven_Seg_Disp SEV0 (d0, muxout);//display data from MUX
14
15  endmodule
16
17  //4-bit comparator
18  module comp4bit(b, a, gr);
19      input [3:0] a; // 4-bit inputs
20      output reg [3:0] b; // 4-bit output
21      output gr;
22      //wire eq, ls;
23      //assign eq = (a == 9) ? 1 : 0; // Equal condition
24      assign gr = (a > 9) ? 1 : 0; // Greater than condition
25      //assign ls = (a < 9) ? 1 : 0; // Less than condition
26      always @ (a)
27        if (a>9)
28         begin
29           b=a;
30         end
31      // If input is greater than 9, output the same four bits
32      //always @(a or b) begin
33      //    if (a > 9)
34      //        gr = 1;
35      //    else
36      //        gr = 0;
37      //end
38  endmodule
39
40  //4-bit wide 2 to 1 multiplexer
41  module mux2to1_4bit(
42      input [3:0] data0, // 4-bit input data 0
43      input [3:0] data1, // 4-bit input data 1
44      input select,      // Select signal (0 or 1), z
45      output reg [3:0] out); //4-bit output
46      always @ (data0, data1)  //put input only in sensitivity list
47          if (select)  //select = 1
48              out = data1;  //data1 from circuit A
49          else
50              out = data0;  //data from V
51  endmodule
52
53  //Circuit A
54  module CircuitA (
55   input [3:0] V, // Input V 4-bit word
56   output reg [3:0] A ); // output 4-bit word
57
58      always @ (V)
59         case (V)
60            4'b1010: A = 4'b0000; // 10 returns 0
61            4'b1011: A = 4'b0001; // 11 returns 1
62            4'b1100: A = 4'b0010; // 12 returns 2
63            4'b1101: A = 4'b0011; // 13 returns 3
64            4'b1110: A = 4'b0100; // 14 returns 4
65            4'b1111: A = 4'b0101; // 15 returns 5
66            default: A = 4'b1111; // Default unique value, detect invalid input
67          endcase
68  endmodule
69
70  // 7-segment display
```

```
71  module Seven_Seg_Disp ( output reg [6:0] Disp, input [3:0] BCD); //input BCD
72  //                output Disp   abc_defg (seven segments, not including dec. pt)
73  parameter       BLANK       = 7'b111_1111;  //blank
74  parameter       ZERO        = 7'b000_0001;  //h01 hexadecimal 1st 3-digits = 0 = 000
75  parameter       ONE         = 7'b100_1111;  //h4F hexadecimal 2nd 4-digits = F = 1111
76  parameter       TWO         = 7'b001_0010;  //h12
77  parameter       THREE       = 7'b000_0110;  //h06
78  parameter       FOUR        = 7'b100_1100;  //h4c
79  parameter       FIVE        = 7'b010_0100;  //h24
80  parameter       SIX         = 7'b010_0000;  //h20
81  parameter       SEVEN       = 7'b000_1111;  //h0f
82  parameter       EIGHT       = 7'b000_0000;  //h00
83  parameter       NINE        = 7'b000_0100;  //h04
84  always @ (BCD)
85    case (BCD) //BCD is decimal value
86     0:          Disp = ZERO;
87     1:          Disp = ONE;
88     2:          Disp = TWO;
89     3:          Disp = THREE;
90     4:          Disp = FOUR;
91     5:          Disp = FIVE;
92     6:          Disp = SIX;
93     7:          Disp = SEVEN;
94     8:          Disp = EIGHT;
95     9:          Disp = NINE;
96     default:    Disp = BLANK;
97    endcase
98  endmodule
99
```