```
1   //
2   // Digital Logic Lab Part 5
3   // (c) Neil Nie
4   // Oct 31st, 2018
5   //
6
7
8   module half_display(
9
10      in,         // accepts a four bit input
11      out         // returns a six bit output
12  );
13
14  input in;         // A
15  output [6:0] out; // D C B A G
16
17
18  assign out[6] = 1;
19  assign out[5] = 1;
20  assign out[4] = 1;
21  assign out[3] = 1;
22  assign out[2] = ~in;
23  assign out[1] = ~in;
24  assign out[0] = 1;
25
26  endmodule
27
28  // full seven segment display module.
29
30  module full_display(
31
32      in,         // accepts a four bit input
33      out         // returns a six bit output
34  );
35
36  input [3:0] in;   // A B C D
37  output [6:0] out; // F E D C B A
38
39  assign out[6] = (~in[3] & ~in[2] & ~in[1]) + (~in[3] & in[2] & in[1] & in[0]);
40  assign out[5] = (in[3] & in[2] & ~in[1]) + (~in[3] & in[1] & in[0]) + (~in[3] & ~in[2] & in[
    0]) + (~in[3] & ~in[2] & in[1]);
41  assign out[4] = (~in[1] & in[2]) | (in[0]);
42  assign out[3] = (in[2] & in[1] & in[0]) + (~in[3] & in[2] & ~in[1] & ~in[0]) + (in[3] & ~in[
    2] & in[1] & ~in[0]) + (~in[2] & ~in[1] & in[0]);
43  assign out[2] = (in[3] & in[2] & ~in[0]) + (~in[3] & ~in[2] & in[1] & ~in[0]) + (in[3] & in[
    2] & in[1]);
44  assign out[1] = (in[2] & in[1] & ~in[0]) + (in[3] & in[1] & in[0]) + (~in[3] & in[2] & ~in[1
    ] & in[0]) + (in[3] & in[2] & ~in[0]);
45  assign out[0] = (in[2] & ~in[1] & ~in[0]) + (in[3] & in[2] & ~in[1]) + (~in[3] & ~in[2] & ~
    in[1] & in[0]) + (in[3] & ~in[2] & in[1] & in[0]);
46
47  endmodule
48
49
50  module Part5(
51
52      SW,
53      HEX0,
54      HEX1
55  );
56
57  // declare and assign all wires
58
59  input  [8:0] SW;
60  output [6:0] HEX0;
61  output [6:0] HEX1;
62
63  wire [3:0] A;
64  wire [3:0] B;
65  wire c_0;
66  assign A = SW[3:0];
67  assign B = SW[7:4];
68  assign c_0 = SW[8];
69
70  // -------- the fun begins
71
```

```
72    reg [3:0] sum;
73    reg [3:0] z;
74    reg [3:0] s0;
75    reg c_1;
76
77    always @ (A or B or c_0) begin
78       sum = A + B + c_0;
79    end
80
81    always @ (sum or z or c_1) begin
82       if (sum > 10) begin
83          z = 10;
84          c_1 = 1;
85       end else begin
86          z = 0;
87          c_1 = 0;
88       end
89       s0 = sum - z;
90    end
91
92    // -------- the fun ends
93
94    wire [6:0] display0;
95    wire [6:0] display1;
96
97    full_display(s0, display0);
98    half_display(c_1, display1);
99
100   assign HEX1 = display1;
101   assign HEX0 = display0;
102
103   endmodule
104
```