Problem 1: [12 points] Drill problem
Filename: library.sv
AndrewID: xinyew

```
 1 `default_nettype none
 2 module Decoder
 3   (input  logic en,
 4    input  logic [2:0] I,
 5    output logic [7:0] D);
 6
 7   always_comb begin
 8     unique case (I)
 9       3'b000: D = 8'b00000001;
10       3'b001: D = 8'b00000010;
11       3'b010: D = 8'b00000100;
12       3'b011: D = 8'b00001000;
13       3'b100: D = 8'b00010000;
14       3'b101: D = 8'b00100000;
15       3'b110: D = 8'b01000000;
16       3'b111: D = 8'b10000000;
17     endcase
18     D = en ? D : 0;
19   end
20
21 endmodule : Decoder
22
23 module BarrelShifter
24   (input  logic [15:0] v,
25    input  logic [3:0] by,
26    output logic [15:0] s);
27
28   always_comb begin
29     s = by[3] ? v << 8 : v;
30     s = by[2] ? s << 4 : s;
31     s = by[1] ? s << 2 : s;
32     s = by[0] ? s << 1 : s;
33   end
34
35 endmodule : BarrelShifter
36
37 module Multiplexer
38   (input  logic [7:0] I,
39    input  logic [2:0] s,
40    output logic Y);
41
42   always_comb
43     case (s)
44       3'b000: Y = I[0];
45       3'b001: Y = I[1];
46       3'b010: Y = I[2];
47       3'b011: Y = I[3];
48       3'b100: Y = I[4];
49       3'b101: Y = I[5];
50       3'b110: Y = I[6];
51       3'b111: Y = I[7];
52     endcase
53
54 endmodule: Multiplexer
55
56 module Mux2to1
57   (input  logic [6:0] I0,
58    input  logic [6:0] I1,
59    input  logic S,
60    output logic [6:0] Y);
61
62   always_comb
63     case (S)
64       0: Y = I0;
65       1: Y = I1;
66     endcase
67
68 endmodule : Mux2to1
69
```

```
 70 module MagComparator
 71   (input  logic [7:0] A,
 72    input  logic [7:0] B,
 73    output logic AltB,
 74    output logic AeqB,
 75    output logic AgtB);
 76
 77    assign AltB = A < B;
 78    assign AgtB = A > B;
 79    assign AeqB = A == B;
 80
 81 endmodule : MagComparator
 82
 83 module Comparator
 84   (input  logic [3:0] A,
 85    input  logic [3:0] B,
 86    output logic AeqB);
 87
 88    assign AeqB = A == B;
 89
 90 endmodule : Comparator
```

Problem 1: [12 points] Drill problem
Filename: library_tests.sv
AndrewID: xinyew

```
`default_nettype none
module Decoder_test;
  logic en;
  logic [2:0] I;
  logic [7:0]D;

  Decoder DUT(.en(en),
              .I(I),
              .D(D));

  initial begin
    $monitor($time,,
      "en = %b, I = %b, D = %b", en, I, D);
    en = 1;
    for (I = 3'b000; I < 3'b111; I++)
      #1;
    en = 0;
    for (I = 3'b000; I < 3'b111; I++)
      #1;
    #1 $finish;
  end

endmodule: Decoder_test

module BarrelShifter_test;
  logic [15:0] v;
  logic [3:0] by;
  logic [15:0] s;

  BarrelShifter DUT(.v(v),
                    .by(by),
                    .s(s));

  initial begin
    $monitor($time,,
      "v = %b, by = %b, s = %b", v, by, s);
    v = 16'b0101010101010101;
    for (by = 4'b0000; by < 4'b1111; by++)
      #1;
    #1 $finish;
  end

endmodule: BarrelShifter_test

module Multiplexer_test;
  logic [7:0] I;
  logic [2:0] s;
  logic Y;

  Multiplexer DUT(.I(I),
                  .s(s),
                  .Y(Y));

  initial begin
    $monitor($time,,
      "I = %b, s = %b, Y = %b", I, s, Y);
    I = 8'b01010101;
    for (s = 3'b000; s < 3'b111; s++)
      #1;
    #1 $finish;
  end

endmodule : Multiplexer_test

module Mux2to1_test;
  logic [6:0] I0;
  logic [6:0] I1;
  logic S;
  logic [6:0] Y;
```

```
 70
 71   Mux2to1 DUT(.I0(I0),
 72               .I1(I1),
 73               .S(S),
 74               .Y(Y));
 75
 76   initial begin
 77     $monitor($time,,
 78       "I0 = %b, I1 = %b, S = %b, Y = %b", I0, I1, S, Y);
 79     I0 = 7'b0000000;
 80     I1 = 7'b1111111;
 81     S = 0;
 82     #1 S = 1;
 83     #1 $finish;
 84   end
 85
 86 endmodule : Mux2to1_test
 87
 88 module MagComparator_test;
 89   logic [7:0] A;
 90   logic [7:0] B;
 91   logic AltB;
 92   logic AgtB;
 93   logic AeqB;
 94
 95   MagComparator DUT (.A(A),
 96                      .B(B),
 97                      .AltB(AltB),
 98                      .AgtB(AgtB),
 99                      .AeqB(AeqB));
100
101   initial begin
102     $monitor($time,,
103     "A = %b, B = %b, AltB = %b, AgtB = %b, AeqB = %b", A, B , AltB, AgtB, AeqB);
104     A = 8'b00000000;
105     B = 8'b11111111;
106     #2 A = 8'b11111111;
107     #2 B = 8'b00000000;
108     #2 $finish;
109   end
110
111 endmodule : MagComparator_test
112
113 module Comparator_test;
114   logic [3:0] A;
115   logic [3:0] B;
116   logic AeqB;
117
118   MagComparator DUT (.A(A),
119                      .B(B),
120                      .AeqB(AeqB));
121
122   initial begin
123     $monitor($time,,
124     "A = %b, B = %b, AeqB = %b", A, B, AeqB);
125     A = 8'b00000000;
126     B = 8'b11111111;
127     #2 A = 8'b11111111;
128     #2 $finish;
129   end
130
131 endmodule : Comparator_test
```

Problem 2: [6 points] Drill problem
Filename: hw2prob2.sv
AndrewID: xinyew

```
 1 module hw2prob2
 2   (input  logic a, b ,c , d,
 3     output logic f, g, h);
 4
 5   always_comb
 6     case ({a, b, c})
 7       3'd1: f = 1;
 8       3'd3: f = 1;
 9       3'd4: f = 1;
10       default: f = 0;
11     endcase
12
13   always_comb begin
14     if ({a, b, c, d} == 4'd1 || {a, b, c, d} == 4'd3
15      || {a, b, c, d} == 4'd5 || {a, b, c, d} == 4'd12)
16       g = 0;
17     else
18       g = 1;
19   end
20
21   assign h = ({b, d} == 2'b11 || {c, d} == 2'b01 || {a, b, c} == 3'b000
22             || {a, b, c} == 3'b011 || {b, c, d} == 3'b010) | 0;
23
24 endmodule : hw2prob2
25
26 module hw2prob2_test;
27   logic [3:0] vector;
28   logic f, g, h;
29
30   hw2prob2 DUT (.a(vector[3]),
31                 .b(vector[2]),
32                 .c(vector[1]),
33                 .d(vector[0]),
34                 .f(f),
35                 .g(g),
36                 .h(h));
37
38   initial begin
39     $monitor($time,,
40       "a = %b, b = %b, c = %b, d = %b, f = %b, g = %b, h = %b",
41       vector[3], vector[2], vector[1], vector[0], f, g, h);
42     for (vector = 4'b0000; vector < 4'b1111; vector++)
43       #1;
44     #1 $finish;
45   end
46
47 endmodule : hw2prob2_test
```

Problem 3: [8 points] Drill problem
Filename: hw2prob3.sv
AndrewID: xinyew

```
 1 module hw2prob3
 2   (input logic a, b, c, d,
 3   output logic l_assign, l_assign_min, l_case, l_casez);
 4
 5   assign l_assign = {a, b, c, d} == 4'd0 || {a, b, c, d} == 4'd2
 6                   || {a, b, c, d} == 4'd4 || {a, b, c, d} == 4'd6
 7                   || {a, b, c, d} == 4'd7 || {a, b, d, d} == 4'd10
 8                   || {a, b, c, d} == 4'd13 || {a, b, c, d} == 4'd5
 9                   || {a, b, c, d} == 4'd8 || {a, b, c, d} == 4'd15;
10
11   assign l_assign_min = {b, d} == 2'b11 || {a, d} == 2'b00
12                       || {a, b, d} == 3'b100;
13
14   always_comb
15     unique case ({a, b, c, d})
16       4'd0: l_case = 1;
17       4'd2: l_case = 1;
18       4'd4: l_case = 1;
19       4'd6: l_case = 1;
20       4'd7: l_case = 1;
21       4'd10: l_case = 1;
22       4'd13: l_case = 1;
23       default: l_case = 0;
24     endcase
25
26   always_comb
27     unique casez ({a, b, c, d})
28       4'b?1?1: l_casez = 1;
29       4'b0??0: l_casez = 1;
30       4'b10?0: l_casez  = 1;
31       default: l_casez = 0;
32     endcase
33
34 endmodule : hw2prob3
35
36 module hw2prob3_test;
37   logic [3:0] vector;
38   logic l_assign, l_assign_min, l_case, l_casez;
39
40   hw2prob3 DUT (.a(vector[3]),
41                 .b(vector[2]),
42                 .c(vector[1]),
43                 .d(vector[0]),
44                 .l_assign(l_assign),
45                 .l_assign_min(l_assign_min),
46                 .l_case(l_case),
47                 .l_casez(l_casez));
48
49   initial begin
50     $monitor($time,,
51       "a = %b, b = %b, c = %b, d = %b, l1 = %b, l2 = %b, l3 = %b, l4 = %b",
52       vector[3], vector[2], vector[1], vector[0],
53       l_assign, l_assign_min, l_case, l_casez);
54     for (vector = 4'b0000; vector < 4'b1111; vector++)
55       #1;
56     #1 $finish;
57   end
58
59 endmodule : hw2prob3_test
```

Problem 4: [8 points]
Filename: hw2prob4.sv
AndrewID: xinyew

```
 1 `default_nettype none
 2 module hw2prob4
 3   (input  logic a, b, c, d, e,
 4    output logic m, n);
 5
 6   assign m = a ^ (b + c & (~d));
 7   always_comb
 8     case ({a, b, c, d, e})
 9       5'd0: n = 0;
10       5'd1: n = 0;
11       5'd2: n = 0;
12       5'd3: n = 0;
13       5'd5: n = 0;
14       5'd9: n = 0;
15       5'd16: n = 0;
16       5'd17: n = 0;
17       5'd22: n = 0;
18       5'd25: n = 0;
19       5'd27: n = 0;
20       default: n = 1;
21     endcase
22
23 endmodule : hw2prob4
24
25 module hw2prob4_test;
26   logic [4:0] vector;
27   logic m, n;
28
29   hw2prob4 DUT (.a(vector[4]),
30                 .b(vector[3]),
31                 .c(vector[2]),
32                 .d(vector[1]),
33                 .e(vector[0]),
34                 .m(m),
35                 .n(n));
36
37   initial begin
38     $monitor($time,,
39       "a = %b, b = %b, c = %b, d = %b, e = %b, m = %b, n = %b",
40       vector[4], vector[3], vector[2], vector[1], vector[0], m, n);
41     for (vector = 5'b00000; vector < 5'b11111; vector++)
42       #1;
43     #1 $finish;
44   end
45
46 endmodule : hw2prob4_test
```

Problem 8: [6 points]
Filename: hw2prob8.sv
AndrewID: xinyew

```
`default_nettype none

module BCDtoSevenSegment
  (input  logic [3:0] bcd,
   output logic [6:0] segment);

   always_comb
    case (bcd)
      4'd0: segment = 7'b0111111;
      4'd1: segment = 7'b0000110;
      4'd2: segment = 7'b1011011;
      4'd3: segment = 7'b1001111;
      4'd4: segment = 7'b1100110;
      4'd5: segment = 7'b1101101;
      4'd6: segment = 7'b1111101;
      4'd7: segment = 7'b0000111;
      4'd8: segment = 7'b1111111;
      4'd9: segment = 7'b1101111;
      default: segment = 7'b0000000;
    endcase

endmodule : BCDtoSevenSegment
```

Problem 9: [6 points]
Filename: 7SegmentDisplay.sv
AndrewID: xinyew

```
`default_nettype none

module SevenSegmentDisplay
  (input  logic [3:0] BCD7, BCD6, BCD5, BCD4, BCD3, BCD2, BCD1, BCD0,
   input  logic [7:0] blank,
   output logic [6:0] HEX7, HEX6, HEX5, HEX4, HEX3, HEX2, HEX1, HEX0);

  always_comb begin
    HEX0 = 7'b0000000;
    if (~blank[0])
      case (BCD0)
        4'd0: HEX0 = 7'b0111111;
        4'd1: HEX0 = 7'b0000110;
        4'd2: HEX0 = 7'b1011011;
        4'd3: HEX0 = 7'b1001111;
        4'd4: HEX0 = 7'b1100110;
        4'd5: HEX0 = 7'b1101101;
        4'd6: HEX0 = 7'b1111101;
        4'd7: HEX0 = 7'b0000111;
        4'd8: HEX0 = 7'b1111111;
        4'd9: HEX0 = 7'b1101111;
        default: HEX0 = 7'b0000000;
      endcase
    HEX0 = ~HEX0;
  end

  always_comb begin
    HEX1 = 7'b0000000;
    if (~blank[1])
      case (BCD1)
        4'd0: HEX1 = 7'b0111111;
        4'd1: HEX1 = 7'b0000110;
        4'd2: HEX1 = 7'b1011011;
        4'd3: HEX1 = 7'b1001111;
        4'd4: HEX1 = 7'b1100110;
        4'd5: HEX1 = 7'b1101101;
        4'd6: HEX1 = 7'b1111101;
        4'd7: HEX1 = 7'b0000111;
        4'd8: HEX1 = 7'b1111111;
        4'd9: HEX1 = 7'b1101111;
        default: HEX1 = 7'b0000000;
      endcase
    HEX1 = ~HEX1;
  end

  always_comb begin
    HEX2 = 7'b0000000;
    if (~blank[2])
      case (BCD2)
        4'd0: HEX2 = 7'b0111111;
        4'd1: HEX2 = 7'b0000110;
        4'd2: HEX2 = 7'b1011011;
        4'd3: HEX2 = 7'b1001111;
        4'd4: HEX2 = 7'b1100110;
        4'd5: HEX2 = 7'b1101101;
        4'd6: HEX2 = 7'b1111101;
        4'd7: HEX2 = 7'b0000111;
        4'd8: HEX2 = 7'b1111111;
        4'd9: HEX2 = 7'b1101111;
        default: HEX2 = 7'b0000000;
      endcase
    HEX2 = ~HEX2;
  end

  always_comb begin
    HEX3 = 7'b0000000;
    if (~blank[3])
      case (BCD3)
        4'd0: HEX3 = 7'b0111111;
```

```
 70         4'd1: HEX3 = 7'b0000110;
 71         4'd2: HEX3 = 7'b1011011;
 72         4'd3: HEX3 = 7'b1001111;
 73         4'd4: HEX3 = 7'b1100110;
 74         4'd5: HEX3 = 7'b1101101;
 75         4'd6: HEX3 = 7'b1111101;
 76         4'd7: HEX3 = 7'b0000111;
 77         4'd8: HEX3 = 7'b1111111;
 78         4'd9: HEX3 = 7'b1101111;
 79         default: HEX3 = 7'b0000000;
 80       endcase
 81     HEX3 = ~HEX3;
 82   end
 83
 84   always_comb begin
 85     HEX4 = 7'b0000000;
 86     if (~blank[4])
 87       case (BCD4)
 88         4'd0: HEX4 = 7'b0111111;
 89         4'd1: HEX4 = 7'b0000110;
 90         4'd2: HEX4 = 7'b1011011;
 91         4'd3: HEX4 = 7'b1001111;
 92         4'd4: HEX4 = 7'b1100110;
 93         4'd5: HEX4 = 7'b1101101;
 94         4'd6: HEX4 = 7'b1111101;
 95         4'd7: HEX4 = 7'b0000111;
 96         4'd8: HEX4 = 7'b1111111;
 97         4'd9: HEX4 = 7'b1101111;
 98         default: HEX4 = 7'b0000000;
 99       endcase
100     HEX4 = ~HEX4;
101   end
102
103   always_comb begin
104     HEX5 = 7'b0000000;
105     if (~blank[5])
106       case (BCD5)
107         4'd0: HEX5 = 7'b0111111;
108         4'd1: HEX5 = 7'b0000110;
109         4'd2: HEX5 = 7'b1011011;
110         4'd3: HEX5 = 7'b1001111;
111         4'd4: HEX5 = 7'b1100110;
112         4'd5: HEX5 = 7'b1101101;
113         4'd6: HEX5 = 7'b1111101;
114         4'd7: HEX5 = 7'b0000111;
115         4'd8: HEX5 = 7'b1111111;
116         4'd9: HEX5 = 7'b1101111;
117         default: HEX5 = 7'b0000000;
118       endcase
119     HEX5 = ~HEX5;
120   end
121
122   always_comb begin
123     HEX6 = 7'b0000000;
124     if (~blank[6])
125       case (BCD6)
126         4'd0: HEX6 = 7'b0111111;
127         4'd1: HEX6 = 7'b0000110;
128         4'd2: HEX6 = 7'b1011011;
129         4'd3: HEX6 = 7'b1001111;
130         4'd4: HEX6 = 7'b1100110;
131         4'd5: HEX6 = 7'b1101101;
132         4'd6: HEX6 = 7'b1111101;
133         4'd7: HEX6 = 7'b0000111;
134         4'd8: HEX6 = 7'b1111111;
135         4'd9: HEX6 = 7'b1101111;
136         default: HEX6 = 7'b0000000;
137       endcase
138     HEX6 = ~HEX6;
139   end
140
```

```
141   always_comb begin
142     HEX7 = 7'b1111111;
143     if (~blank[7])
144       case (BCD7)
145         4'd0: HEX7 = 7'b0111111;
146         4'd1: HEX7 = 7'b0000110;
147         4'd2: HEX7 = 7'b1011011;
148         4'd3: HEX7 = 7'b1001111;
149         4'd4: HEX7 = 7'b1100110;
150         4'd5: HEX7 = 7'b1101101;
151         4'd6: HEX7 = 7'b1111101;
152         4'd7: HEX7 = 7'b0000111;
153         4'd8: HEX7 = 7'b1111111;
154         4'd9: HEX7 = 7'b1101111;
155         default: HEX7 = 7'b0000000;
156       endcase
157     HEX7 = ~HEX7;
158   end
159
160 endmodule : SevenSegmentDisplay
```