

```

59 module registerfile (
60     input clock,
61     input clear,
62     input[3:0] write_data,
63     input[2:0] write_index,
64     input write,
65     input[2:0] read_index,
66     output[3:0] read_data);
67
68     wire[7:0] d;
69     wire[3:0] q0;
70     wire[3:0] q1;
71     wire[3:0] q2;
72     wire[3:0] q3;
73     wire[3:0] q4;
74     wire[3:0] q5;
75     wire[3:0] q6;
76     wire[3:0] q7;
77
78     decoder decoder1(write, write_index, d1);
79     register register1(clock, clear, d[0], write_data, q0);
80     register register2(clock, clear, d[1], write_data, q1);
81     register register3(clock, clear, d[2], write_data, q2);
82     register register4(clock, clear, d[3], write_data, q3);
83     register register5(clock, clear, d[4], write_data, q4);
84     register register6(clock, clear, d[5], write_data, q5);
85     register register7(clock, clear, d[6], write_data, q6);
86     register register8(clock, clear, d[7], write_data, q7);
87     mux mux1(q0, q1, q2, q3, q4, q5, q6, q7, read_index, read_data);
88
89 endmodule
90
91

```

RegisterFile.v

```

1 module decoder (
2     input enable,
3     input[2:0] in,
4     output[7:0] out);
5
6     assign out[0] = enable & ~in[2] & ~in[1] & ~in[0];
7     assign out[1] = enable & ~in[2] & ~in[1] & in[0];
8     assign out[2] = enable & ~in[2] & in[1] & ~in[0];
9     assign out[3] = enable & ~in[2] & in[1] & in[0];
10    assign out[4] = enable & in[2] & ~in[1] & ~in[0];
11    assign out[5] = enable & in[2] & ~in[1] & in[0];
12    assign out[6] = enable & in[2] & in[1] & ~in[0];
13    assign out[7] = enable & in[2] & in[1] & in[0];
14
15 endmodule
16
17 module mux (
18     input[3:0] in0,
19     input[3:0] in1,
20     input[3:0] in2,
21     input[3:0] in3,
22     input[3:0] in4,
23     input[3:0] in5,
24     input[3:0] in6,
25     input[3:0] in7,
26     input[2:0] selector,
27     output reg[3:0] out);
28
29     always @(in0, in1, in2, in3, in4, in5, in6, in7, selector)
30     case (selector)
31         0: out = in0;
32         1: out = in1;
33         2: out = in2;
34         3: out = in3;
35         4: out = in4;
36         5: out = in5;
37         6: out = in6;
38         7: out = in7;
39     endcase
40

```

RegisterFile.v

```
42
43 module register (
44     input clock,
45     input clear,
46     input load,
47     input[3:0] in,
48     output reg[3:0] out);
49
50     always @(posedge clear, negedge clock)
51     if (clear) begin
52         out = 0;
53     end
54     else if (load) begin
55         out = in;
56     end
57
58 endmodule
59
60 module registerfile (
61     input clock,
62     input clear,
63     input[3:0] write_data,
64     input[2:0] write_index,
65     input write,
66     input[2:0] read_index,
67     output[3:0] read_data);
68
69     wire[7:0] d;
70     wire[3:0] q0;
71     wire[3:0] q1;
72     wire[3:0] q2;
73     wire[3:0] q3;
74     wire[3:0] q4;
75     wire[3:0] q5;
76     wire[3:0] q6;
77     wire[3:0] q7;
78
79     decoder decoder1(write, write_index, d1);
80     register register1(clock, clear, d[0], write_data, q0);
81     register register2(clock, clear, d[1], write_data, q1);
```