```
2 //
3 // Company:
4 // Engineer:
                    Ralnikov Vadim Dmitryevich
5 //
6 // Create Date:
                    20:00 04/03/2019
7 // Design Name:
8 // Module Name:
                    test_bench
9 // Project Name:
10 // Target Devices:
11 // Tool versions:
12 // Description:
                    test_bench for memory RAM
13 //
15
16 `timescale 1ns / 100ps
17 `default_nettype none
18
19
20 module test_bench;
21
22 parameter PERIOD = 10;
23
24
25 wire
                    CLK;
                                                    // in, u[ 1], Clock freq 100MHz
26 wire
                    EN;
                                                       in, u[ 1], enable work 1 -enable
                                                    // in, u[ 1], address input 0 to 2048
27 wire
                    ADDRESS;
        [ 10 : 0 ]
                                                    // in, u[ 1], enable write 1 - enable
28 wire
                    WE;
29 wire
       [ 7: 0]
                    DI;
                                                    // in, u[ 8], input data, parallel
30
31 wire
                                                    // out, u[ 8], out data, parallel
       [ 7: 0]
                    DO;
32
33
34
35
                             = 1'b0;
36 reg
                    r_clk
                             = 1'b0;
                    r_en
37 reg
                             = 11'b0;
38 reg
        [
          10 : 0 ]
                    r_adr
39 reg
                    r_we
                             = 1'b0;
                    r_di
40 reg
           7:0]
                             = 8'b0;
                    r_counter = 8'b11111111;
41 reg
          7:0]
42
43
44 assign CLK
                = r_clk;
45 assign EN
                = r_en;
46 assign ADDRESS = r_adr;
47 assign WE
                = r we;
48 assign DI
                = r_di;
49
50
51 integer
            i;
52
53 //----
54
55 task do_write;
56
57
  input [ 10 : 0 ] i_addr;
                    i_data;
58 input [ 7:0]
59
60
     begin
61
62
    #20
63
64
    @(posedge CLK);
65
66
    r en = 1;
    r_we = 1;
67
68
    r_adr = i_addr;
69
    r_di = i_data;
70
71
    #20
72
```

```
73
      @(posedge CLK);
 74
 75
      r_we = 0;
 76
 77
       end
 78
 79
   endtask
80
81
    //----
82
83 task do_read;
84
85
    input [ 10 : 0 ] i_addr;
86
87
       begin
88
      #20
89
90
91
      r_we = 0;
92
93
      @(posedge CLK);
94
95
      r_en = 1;
96
      r_we = 0;
97
      r_adr = i_addr;
98
99
       end
100
101 endtask
102
103 //----
104
105 initial begin
                 forever
106
107
                 \#(PERIOD/2) r_clk = \sim r_clk;
108
            end
109
110
111 always @(posedge WE)
                                                               // cycle for data write
112
       begin
113
           r_counter <= r_counter - 8'b00000001;
114
             if(r_counter==0)
           r_counter<=8'b11111111;
115
116
       end
117
118
119
120
121 initial begin
                                                               // cycle for address write
122
123
      for (i=0; i<2048; i = i+1)
124
        begin
125
126 #10;
127
128 do_write (i, r_counter);
129
130
       end
131
132 #10;
133 do_read (0);
134 #40;
135 do_read (11'b00000000001);
136 #40;
137 do_read (11'b00000000111);
138 #40;
139 do_read (11'b11110000111);
140 #40;
141 do_read (11'b1111111111);
142
143
       end
144
```

```
145
146
147
148
149
150
151
152 memory
153
             memory_inst
154
                                                                                                 // in, u[ 1], Clock freq 100MHz
// in, u[ 1], enable work 1 -enable
// in, u[11], address from 0 to 2048
// in, u[ 1], write enable, 1 -enable
// in, u[ 8], input data, parallel
                                                           (CLK),
155
                    .CLK
156
                    .EN
                                                           (EN),
                                                           (ADDRESS),
157
                    .ADDRESS
158
                    .WE
                                                           (WE),
159
                    .DI
                                                           (DI),
160
                    .DO
161
                                                           (DO)
                                                                                                  // out, u[ 8], output data, parallel
162
             );
163
164 endmodule
```