```
 1
 2  module marioReg   (clk, go, ground, updateX, updateY,
 3                     erM, jump, jumpL, jumpCounter, jumping, falling, drStage1, drStage2,
    drStage3,
 4                     right, left, up, down, outofBounds, pipe, next, flag, start, lvl1, lvl2,
    lvl3, dead);
 5
 6     output reg [7:0] updateX, updateY;
 7     input erM, jump, jumpL, jumpCounter, jumping, falling, drStage1, drStage2, drStage3,
    start, lvl1, lvl2, lvl3;
 8     input right, left, up, down;
 9     reg [7:0] initialx;
10     reg [7:0] initialy;
11
12     output reg outofBounds, pipe, next, flag, dead;
13     initial outofBounds = 0;
14     input clk, go;
15
16     reg goDown;
17     initial goDown = 1;
18     output reg ground;
19     reg [4:0] counter;
20
21     reg facingLeft, facingRight;
22
23     initial initialx = 10'd4;
24     initial initialy = 10'd89;
25
26     reg update;
27     initial update = 0;
28     reg stop;
29     initial stop = 1;
30     reg stop2;
31     initial stop2 = 1;
32     reg stop3;
33     initial stop3 = 1;
34
35     always@(posedge clk) begin
36     //movement
37        //initial value
38        if (start) begin
39           stop <= 1;
40           stop2 <= 1;
41           stop3 <= 1;
42           next <= 0;
43           flag <= 0;
44           pipe <= 0;
45           dead <= 0;
46        end
47        else if (drStage1 && stop && lvl1) begin
48           updateX <= 10'd4;
49           updateY <= 10'd89;
50           initialx <= 10'd4;
51           initialy <= 10'd89;
52           stop <= 0;
53        end
54        else if (drStage2 && stop2 && lvl2) begin
55           updateX <= 10'd20;
56           updateY <= 10'd20;
57           initialx <= 10'd20;
58           initialy <= 10'd20;
59           stop2 <= 0;
60        end
61        else if (drStage3 && stop3 && lvl3) begin
62           updateX <= 10'd2;
63           updateY <= 10'd89;
64           initialx <= 10'd2;
65           initialy <= 10'd89;
66           stop3 <= 0;
67        end
68        //jumping
69        else if (jumping && right && !outofBounds) begin
70           updateY <= initialy - 8'b00000001;
```

```
71              updateX <= initialx + 8'b00000001;
72              update <= 1;
73           end
74           else if (jumping && left && !outofBounds) begin
75              updateY <= initialy - 8'b00000001;
76              updateX <= initialx - 8'b00000001;
77              update <= 1;
78           end
79           else if (jumping) begin
80              updateY <= initialy - 8'b00000001;
81              update <= 1;
82           end
83           //falling
84           else if (falling && right && !outofBounds) begin
85              updateY <= initialy + 8'b00000001;
86              updateX <= initialx + 8'b00000001;
87              update <= 1;
88           end
89           else if (falling && left && !outofBounds) begin
90              updateY <= initialy + 8'b00000001;
91              updateX <= initialx - 8'b00000001;
92              update <= 1;
93           end
94           else if (falling) begin
95              updateY <= initialy + 8'b00000001;
96              update <= 1;
97           end
98           //right
99           else if (erM && right && !outofBounds) begin
100             updateX <= initialx + 8'b00000001;
101             update <= 1;
102          end
103          //left
104          else if (erM && left && !outofBounds) begin
105             updateX <= initialx - 8'b00000001;
106             update <= 1;
107          end
108          //update position
109          else if (update) begin
110             initialx <= updateX;
111             initialy <= updateY;
112             update <= 0;
113          end
114
115          //out of bounds conditions
116          if (lvl1) begin
117             if (updateX - 1'b1 == 8'd0) begin
118                outofBounds <= 1;
119                facingLeft <= 1;
120                facingRight <= 0;
121             end
122             else if (updateX + 5'd12 == 8'd160) begin
123                outofBounds <= 1;
124                facingRight <= 1;
125                facingLeft <= 0;
126             end
127             else if (updateX + 5'd12 == 8'd39 && (updateY + 5'd16 > 8'd88)) begin
128                outofBounds <= 1;
129                facingRight <= 1;
130                facingLeft <= 0;
131             end
132             else if (updateX + 5'd12 == 8'd55 && (updateY > 8'd72)) begin
133                outofBounds <= 1;
134                facingRight <= 1;
135                facingLeft <= 0;
136             end
137             else if (updateX + 5'd12 == 8'd71 && (updateY > 8'd56)) begin
138                outofBounds <= 1;
139                facingRight <= 1;
140                facingLeft <= 0;
141             end
142             else if (updateX + 5'd12 == 8'd133 && (updateY > 8'd84)) begin
143                outofBounds <= 1;
```

```
144                facingRight <= 1;
145                facingLeft <= 0;
146             end
147             else if (updateX - 5'd1 == 8'd86) begin
148                outofBounds <= 1;
149                facingRight <= 0;
150                facingLeft <= 1;
151             end
152             else
153                outofBounds <= 0;
154          end
155
156          else if (lvl2) begin
157             if (updateX - 1'b1 == 8'd14) begin
158                outofBounds <= 1;
159                facingLeft <= 1;
160                facingRight <= 0;
161             end
162             else if (updateY - 1'b1 < 8'd19) begin
163                outofBounds <= 1;
164                facingLeft <= 0;
165                facingRight <= 1;
166             end
167          end
168          else if (lvl3) begin
169             if (updateX - 1'b1 == 8'd0) begin
170                outofBounds <= 1;
171                facingLeft <= 1;
172                facingRight <= 0;
173             end
174             else if (updateX + 5'd12 == 8'd98 && (updateY + 5'd16 > 8'd88)) begin
175                outofBounds <= 1;
176                facingRight <= 1;
177                facingLeft <= 0;
178             end
179          end
180
181          //out of bounds reset
182          if (facingRight == 1 && left) begin
183             outofBounds <= 0;
184             facingRight <= 0;
185          end
186          else if (facingLeft == 1 && right) begin
187             outofBounds <= 0;
188             facingLeft <= 0;
189          end
190
191          //ground conditions
192          if (updateY == 8'd194 && lvl1)
193             ground <= 0;
194          else if ( ((updateY + 5'd16 > 8'd104 && !(updateX > 8'd86 && updateX + 5'd12 < 8'd106)
     ) || (updateY + 5'd16 == 8'd89 && (updateX + 5'd11 > 8'd38 && updateX < 8'd55))
195                   || (updateY + 5'd16 == 8'd73 && (updateX + 5'd11 > 8'd54 && updateX < 8'd71))
     || (updateY + 5'd16 == 8'd57 && (updateX + 5'd11 > 8'd70 && updateX < 8'd87))
196                   || (updateY + 5'd16 == 8'd85 && (updateX + 5'd11 > 8'd132))) && lvl1)
197             ground <= 1;
198          else if (updateY + 5'd16 > 8'd104 && (lvl2 || lvl3))
199             ground <= 1;
200          else if (updateY + 5'd16 == 8'd89 && (updateX + 5'd12 > 8'd97 && updateX < 8'd114) &&
   lvl3)
201             ground <= 1;
202          else
203             ground <= 0;
204
205          //pipe conditions
206          if (lvl1) begin
207             if (updateY + 5'd16 == 8'd85 && (updateX > 8'd132 && updateX + 5'd12 < 8'd154) &&
   down)
208                pipe <= 1;
209             else
210                pipe <= 0;
211          end
212
```

```
213         //next level conditions
214         if (lvl2) begin
215            if (updateX == 8'd159)
216               next <= 1;
217            else
218               next <= 0;
219         end
220
221         //flag conditions
222         if (lvl3) begin
223            if (updateX == 8'd100)
224               flag <= 1;
225            else
226               flag <= 0;
227         end
228
229         //dead
230         if (lvl1) begin
231            if (updateY + 5'd15 == 8'd119)
232               dead <= 1;
233            else
234               dead <= 0;
235         end
236      end
237
238   endmodule
```