```
--Programmed by: Luis Barquero
--Purpose: This program will take the T-Bird's light example, and implement the brakes.
     --For example, when the brakes are activated, all six lights are on.
     --When the left or turn signal are on, the lights the lights in the direction of the turn
     -- function as before ? blinking sequentially, the lights opposite the tail lights
     --are all on continuously.

library IEEE;
use IEEE.std_logic_1164.all;

ENTITY tbird_lc is
  PORT(clk           : IN  std_logic;
       rst           : IN  std_logic;
       left          : IN std_logic;
       right         : IN std_logic;
       haz           : in std_logic;
       brakes        : in std_logic;
       left_tail_lt  : OUT std_logic_vector(3 downto 1);
       right_tail_lt : OUT std_logic_vector(1 to 3));
END tbird_lc;

ARCHITECTURE behavior OF tbird_lc IS

  TYPE state_type IS (IDLE, LR3, L1, L2, L3, L4, L5, L6, R1, R2, R3, R4, R5, R6);
  SIGNAL present_state, next_state : state_type;
  CONSTANT leftoff : std_logic_vector(3 downto 1) := "000";
  CONSTANT left1on : std_logic_vector(3 downto 1) := "001";
  CONSTANT left2on : std_logic_vector(3 downto 1) := "011";
  CONSTANT left3on : std_logic_vector(3 downto 1) := "111";
  CONSTANT rightoff : std_logic_vector(3 downto 1) := "000";
  CONSTANT right1on : std_logic_vector(3 downto 1) := "100";
  CONSTANT right2on : std_logic_vector(3 downto 1) := "110";
  CONSTANT right3on : std_logic_vector(3 downto 1) := "111";

BEGIN

 clocked : PROCESS(clk,rst)
   BEGIN
     IF(rst='1') THEN
       present_state <= idle;
    ELSIF(rising_edge(clk)) THEN
      present_state <= next_state;
    END IF;
 END PROCESS clocked;

 nextstate : PROCESS(present_state,left,right,haz, brakes)
  BEGIN
     CASE present_state IS
       WHEN idle => --(000_000). Idle will have all lights turned off.
         IF(haz = '1'OR (left = '1' AND right = '1') OR brakes = '1') THEN
           next_state <= LR3;
         ELSIF(left = '1') THEN
           next_state <= L1;
         ELSIF(right = '1') THEN
           next_state <= R1;
         ELSE
           next_state <= idle;
         END IF;

       WHEN LR3 => --(111_111). LR3 will have all six lights(all 3 from left side and all 3 from right side).
     IF(left = '1' AND brakes = '1') THEN
       next_state <= L4;
     ELSIF (right = '1' AND brakes = '1') THEN
        next_state <= R4;
       ELSIF ((left = '0') AND (right = '0') AND (brakes = '1')) THEN
          next_state <= present_state;
     ELSE
```

```
68            next_state <= idle;
69            END IF;
70
71          WHEN L1 => --(001_000). L1 will implement the first left light, with the right
            lights off because of no brakes.
72            IF(haz = '1') THEN
73              next_state <= LR3;
74            ELSIF (brakes = '1') THEN
75          next_state <= L6;
76            ELSIF (haz = '0') THEN
77          next_state <= L2;
78            END IF;
79
80          WHEN L2 => --(011_000). L2 will implement the first two left light, with the
            right lights off because of no brakes.
81            IF(haz = '1' OR brakes = '1') THEN
82              next_state <= LR3;
83            ELSIF (haz = '0' AND brakes = '0') THEN
84              next_state <= L3;
85            END IF;
86
87          WHEN L3 => --(111_000). L3 will have all 3 lights from left side, with the right
            side lights off because of no brakes.
88            IF(brakes = '1') THEN
89               next_state <= L4;
90            ELSIF (haz = '1' OR brakes = '1') THEN
91               next_state <= LR3;
92            ELSE
93               next_state <= idle;
94          END IF;
95
96          WHEN L4 => -- (000_111) Left and Brakes. L4 will reset the left turn lights
            after all three have been lit when the brake has been activated.
97         IF (brakes = '0') THEN
98            next_state <= L1;
99         ELSIF (brakes = '1') THEN
100           next_state <= L5;
101         END IF;
102
103         WHEN L5 => --(001_111) Left and Brakes. L5 will have the first left light on and
            all the right lights on, because of the brakes.
104       IF (brakes = '0') THEN
105              next_state <= L2;
106        ELSIF (brakes = '1') THEN
107        next_state <= L6;
108        END IF;
109
110       WHEN L6 => --(011_111) Left and Brakes. L6 will have the firt two left lights on
          and all the right lights on, because of the brakes.
111        IF (brakes = '0') THEN
112           next_state <= L3;
113        ELSIF (brakes = '1') THEN
114           next_state <= LR3;
115            END IF;
116
117         WHEN R1 => --(000_100). R1 will implement the first right light, with the left
            lights off because of no brakes.
118           IF(haz = '1') THEN
119             next_state <= LR3;
120           ELSIF (haz = '0' AND brakes = '0') THEN
121             next_state <= R2;
122        ELSIF (brakes = '1') THEN
123             next_state <= R6;
124           END IF;
125
126         WHEN R2 => --(000_110). R2 will implement the first two right lights, with the
            left lights off because of no brakes.
127           IF(haz = '1' or brakes = '1') THEN
128             next_state <= LR3;
```

```
129             ELSIF (haz = '0' AND brakes = '0') THEN
130               next_state <= R3;
131             END IF;
132
133           WHEN R3 => --(000_111). R3 will have all 3 lights from right side, with the left
              side lights off because of no brakes.
134         IF (haz = '1' OR brakes = '1') THEN
135            next_state <= LR3;
136         ELSIF (brakes = '1') THEN
137            next_state <= R4;
138         ELSE
139                next_state <= idle;
140           END IF;
141
142        WHEN R4 => -- (111_000) Right and Brakes. R4 will reset the right turn lights after
           all three have been lit when the brake has been activated.
143         IF (brakes = '0') THEN
144            next_state <= R1;
145         ELSIF (brakes = '1') THEN
146            next_state <= R5;
147           END IF;
148
149        WHEN R5 => --(111_100) Right and Brakes. L5 will have the first right light on and
           all the left lights on, because of the brakes.
150        IF (brakes = '0') THEN
151                next_state <= R2;
152            ELSIF (brakes = '1') THEN
153            next_state <= R6;
154        END IF;
155
156        WHEN R6 => --(111_110) Right and Brakes. L5 will have the first two right light on
           and all the left lights on, because of the brakes.
157        IF (brakes = '0') THEN
158            next_state <= R3;
159        ELSIF (brakes = '1') THEN
160            next_state <= LR3;
161        END IF;
162
163        END CASE;
164      END PROCESS nextstate;
165
166      output : PROCESS(present_state)
167       BEGIN
168         CASE present_state IS
169           WHEN idle =>  --(000_000)
170             left_tail_lt <= leftoff;
171             right_tail_lt <= rightoff;
172
173           WHEN LR3 => --(111_111)
174             left_tail_lt <= left3on;
175             right_tail_lt <= right3on;
176
177           WHEN L1 => --(001_000)
178             left_tail_lt <= left1on;
179             right_tail_lt <= rightoff;
180
181           WHEN L2 => --(011_000)
182             left_tail_lt <= left2on;
183             right_tail_lt <= rightoff;
184
185           WHEN L3 => --(111_000)
186             left_tail_lt <= left3on;
187             right_tail_lt <= rightoff;
188
189          WHEN L4 => --(000_111)
190         left_tail_lt <= leftoff;
191         right_tail_lt <= right3on;
192
193          WHEN L5 => --(001_111)
```

```
194         left_tail_lt <= left1on;
195         right_tail_lt <= right3on;
196 
197          WHEN L6 => --(011_111)
198         left_tail_lt <= left2on;
199             right_tail_lt <= right3on;
200 
201           WHEN R1 => --(000_100)
202             left_tail_lt <= leftoff;
203             right_tail_lt <= right1on;
204 
205           WHEN R2 => --(000_110)
206             left_tail_lt <= leftoff;
207             right_tail_lt <= right2on;
208 
209           WHEN R3 => --(000_111)
210             left_tail_lt <= leftoff;
211             right_tail_lt <= right3on;
212 
213           WHEN R4 => --(111_000)
214         left_tail_lt <= left3on;
215         right_tail_lt <= rightoff;
216 
217          WHEN R5 => --(111_100)
218         left_tail_lt <= left3on;
219         right_tail_lt <= right1on;
220 
221          WHEN R6 => --(111_110)
222         left_tail_lt <= left3on;
223             right_tail_lt <= right2on;
224 
225 
226         END CASE;
227      END PROCESS output;
228 
229    END ARCHITECTURE behavior;
```