```
1   ----------------------------------------------------------------------------------
2   -- Company:
3   -- Engineer:
4   --
5   -- Create Date:    14:52:33 05/14/2021
6   -- Design Name:
7   -- Module Name:    TimerReset - Behavioral
8   -- Project Name:
9   -- Target Devices:
10  -- Tool versions:
11  -- Description:
12  --
13  -- Dependencies:
14  --
15  -- Revision:
16  -- Revision 0.01 - File Created
17  -- Additional Comments:
18  --
19  ----------------------------------------------------------------------------------
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22  use IEEE.NUMERIC_STD.ALL;
23
24  -- Uncomment the following library declaration if using
25  -- arithmetic functions with Signed or Unsigned values
26
27
28  -- Uncomment the following library declaration if instantiating
29  -- any Xilinx primitives in this code.
30  --library UNISIM;
31  --use UNISIM.VComponents.all;
32
33  entity TimerReset is
34      Port ( reset : in  STD_LOGIC;
35             clock : in  STD_LOGIC;
36             TrainButton : in  STD_LOGIC; -- Train Button Input --
37             CarButton : in  STD_LOGIC; -- Car Button Input --
38             PedButton : in  STD_LOGIC; -- Pedestrain Button Input --
39             Found : out  STD_LOGIC -- Output To Reset Timer --
40             );
41  end TimerReset;
42
43  architecture Behavioral of TimerReset is
44
45  type StateType is (Train, Car, Pedestrian);
46
47  signal state, nextState : StateType;
48
49  begin
50
51        SynchronousProcess:
52        process (reset, clock)
53        begin
54           if (reset = '1') then -- If Reset Button Is Pressed --
55              state <= Car; -- Set Initial State As Car --
56
57           elsif rising_edge(clock) then -- If Rising Clock Edge --
```

```
58              state <= nextState; -- Change Current State To Next State --
59
60           end if;
61        end process SynchronousProcess;
62
63        TimerProcess:
64           process (state, TrainButton, CarButton, PedButton)
65           begin
66
67           Found <= '0'; -- Set Initial Found Output As 0 --
68           nextState <= Car; -- Set Next State As Car --
69
70           case state is
71              when Train => -- When State Is Train --
72
73                 if (CarButton = '1') then -- If CarButton Is Pressed --
74                    Found <= '1'; -- Found Output Is 1 --
75                    nextState <= Car; -- Change Next State To Car --
76
77                 elsif (PedButton = '1') then -- If PedButton Is Pressed --
78                    Found <= '1'; -- Found Output Is 1 --
79                    nextState <= Pedestrian; -- Change Next State To Pedestrian --
80
81                 else -- If None Of The Above Condition Is Satisfied --
82                    nextState <= Train; -- Hold Current State --
83
84                 end if;
85
86              when Car => -- When State Is Car --
87
88                 if (TrainButton = '1') then -- If TrainButton Is Pressed --
89                    Found <= '1'; -- Found Output Is 1 --
90                    nextState <= Train; -- Change Next State To Train --
91
92                 elsif (PedButton = '1') then -- If PedButton Is Pressed --
93                    Found <= '1'; -- Found Output Is 1 --
94                    nextState <= Pedestrian; -- Change Next State To Pedestrian --
95
96                 else -- If None Of The Above Condition Is Satisfied --
97                    nextState <= Car; -- Hold Current State --
98
99                 end if;
100
101             when Pedestrian => -- When State Is Pedestrian --
102
103                if (TrainButton = '1') then -- If TrainButton Is Pressed --
104                   Found <= '1'; -- Found Output Is 1 --
105                   nextState <= Train; -- Change Next State To Train --
106
107                elsif (CarButton = '1') then -- If CarButton Is Pressed --
108                   nextState <= Car; -- Change Next State To Car --
109
110                else -- If None Of The Above Condition Is Satisfied --
111                   nextState <= Pedestrian; -- Hold Current State --
112
113                end if;
114          end case;
```

```
115
116         end process TimerProcess;
117
118   end Behavioral;
119
120
```