```
1   ----------------------------------------------------------------------------------
2   -- Company:
3   -- Engineer:
4   --
5   -- Create Date:    15:57:45 05/20/2021
6   -- Design Name:
7   -- Module Name:    SyncButton - 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
23  -- Uncomment the following library declaration if using
24  -- arithmetic functions with Signed or Unsigned values
25  --use IEEE.NUMERIC_STD.ALL;
26
27  -- Uncomment the following library declaration if instantiating
28  -- any Xilinx primitives in this code.
29  --library UNISIM;
30  --use UNISIM.VComponents.all;
31
32  entity SyncButton is
33      Port ( reset : in  STD_LOGIC;
34             clock : in  STD_LOGIC;
35             TrainButton : in  STD_LOGIC; -- Train Button Input --
36             CarButton : in  STD_LOGIC; -- Car Button Input --
37             PedButton : in  STD_LOGIC; -- Pedestrain Button Input --
38             Counter : in  STD_LOGIC_VECTOR (4 downto 0); -- Timer --
39             MotorEnabler : out  STD_LOGIC; -- Enable And Disable Stepper Motor --
40             TrainAction : out  STD_LOGIC; -- Tran Button Output --
41             CarWaitOutput : out  STD_LOGIC; -- Car Output When Train Is And Is Not
   Passing Through --
42             PedWaitOutput : out  STD_LOGIC -- Pedestrain Output When Train Is And Is
   Not Passing Through --
43             );
44  end SyncButton;
45
46  architecture Behavioral of SyncButton is
47
48  type StateType is (GateClose, GateOpenCar, GateOpenPed, TrainPassed, TrainPassing, Car
   , PedestrianPassing);
49
50  signal State, NextState : StateType;
51
52  begin
53
54     SyncButtonProcess:
```

```
55          process (reset, clock)
56
57          begin
58             if (reset = '1') then -- If reset is pressed --
59                State <= Car; -- Change The State To Traffic_Green --
60
61             elsif (rising_edge(clock)) then -- If Rising Clock Edge --
62                State <= NextState; -- Change The State To The Next State --
63
64             end if;
65
66          end process;
67
68
69       InputProcess:
70          process (State, Counter, TrainButton, PedButton, CarButton)
71
72          begin
73
74             TrainAction <= '0'; -- Set Initial TrainAction Output To Be 0 --
75             CarWaitOutput <= '0'; -- Set Initial CarWaitOutput Output To Be 0 --
76             PedWaitOutput <= '0'; -- Set Initial PedWaitOutput Output To Be 0 --
77             MotorEnabler <= '0'; -- Set Initial MotorSignalDetector Output To Be 0 --
78
79             NextState <= Car; -- Set Current State To Car --
80
81             case state is
82
83                when GateClose  => -- When The State Is GateClose --
84                   TrainAction <= '1'; -- Set TrainAction Output To Be 1 --
85                   CarWaitOutput <= '0'; -- Set CarWaitOutput Output To Be 0 --
86                   PedWaitOutput <= '0'; -- Set PedWaitOutput Output To Be 0 --
87                   MotorEnabler <= '1'; -- Set MotorEnabler Output To Be 1 --
88
89                   if (Counter = "00101") then -- If Counter Reached "00101" --
90                      NextState <= TrainPassing; -- Change State To TrainPassing --
91
92                   else
93                      NextState <= GateClose; -- If None Of The Above Occur, Hold State --
94
95                   end if;
96
97                when GateOpenCar  => -- When The State Is GateOpenCar --
98                   TrainAction <= '0'; -- Set TrainAction Output To Be 0 --
99                   CarWaitOutput <= '1'; -- Set CarWaitOutput Output To Be 1 --
100                  PedWaitOutput <= '0'; -- Set PedWaitOutput Output To Be 0 --
101                  MotorEnabler <= '1'; -- Set MotorEnabler Output To Be 1 --
102
103                  if (Counter = "00101") then -- If Counter Reached "00101" --
104                     NextState <= Car; -- Change State To Car --
105
106                  else
107                     NextState <= GateOpenCar; -- If None Of The Above Occur, Hold State
    --
108
109                  end if;
110
```

```
111               when GateOpenPed  => -- When The State Is GateOpenPed --
112                  TrainAction <= '0'; -- Set TrainAction Output To Be 0 --
113                  CarWaitOutput <= '0'; -- Set CarWaitOutput Output To Be 0 --
114                  PedWaitOutput <= '1'; -- Set PedWaitOutput Output To Be 1 --
115                  MotorEnabler <= '1'; -- Set MotorEnabler Output To Be 1 --
116
117                  if (Counter = "00101") then -- If Counter Reached "00101" --
118                     NextState <= PedestrianPassing; -- Change State To Car --
119
120                  else
121                     NextState <= GateOpenPed; -- If None Of The Above Occur, Hold State
    --
122
123                  end if;
124
125               when TrainPassed  => -- When The State Is TrainPassed --
126                  TrainAction <= '0'; -- Set TrainAction Output To Be 0 --
127                  CarWaitOutput <= '0'; -- Set CarWaitOutput Output To Be 0 --
128                  PedWaitOutput <= '0'; -- Set PedWaitOutput Output To Be 0 --
129                  MotorEnabler <= '0'; -- Set MotorEnabler Output To Be 0 --
130
131                  if (TrainButton = '1') then -- If TrainButton Is Pressed --
132                     NextState <= TrainPassing; -- Change State To TrainPassing --
133
134                  elsif (CarButton = '1') then -- If CarButton Is Pressed --
135                     NextState <= GateOpenCar; -- Change State To GateOpenCar --
136
137                  elsif (PedButton = '1') then -- If PedButton Is Pressed --
138                     NextState <= GateOpenPed; -- Change State To GateOpenPed --
139
140                  else
141                     NextState <= TrainPassed; -- If None Of The Above Occur, Hold State
    --
142                  end if;
143
144               when TrainPassing  => -- When The State Is TrainPassing --
145                  TrainAction <= '1'; -- Set TrainAction Output To Be 1 --
146                  CarWaitOutput <= '0'; -- Set CarWaitOutput Output To Be 0 --
147                  PedWaitOutput <= '0'; -- Set PedWaitOutput Output To Be 0 --
148                  MotorEnabler <= '0'; -- Set MotorEnabler Output To Be 0 --
149
150                  if (Counter = "11110") then -- If Counter Reached "11110" --
151                     NextState <= TrainPassed; -- Change State To TrainPassed --
152
153                  else
154                     NextState <= TrainPassing; -- If None Of The Above Occur, Hold
    State --
155                  end if;
156
157               when Car => -- When The State Is Car --
158                  TrainAction <= '0'; -- Set TrainAction Output To Be 0 --
159                  CarWaitOutput <= '1'; -- Set CarWaitOutput Output To Be 1 --
160                  PedWaitOutput <= '0'; -- Set PedWaitOutput Output To Be 0 --
161                  MotorEnabler <= '0'; -- Set MotorEnabler Output To Be 0 --
162
163                  if (TrainButton = '1') then -- If TrainButton Is Pressed --
164                     NextState <= GateClose; -- Change State To GateClose --
```

```
165
166                  elsif (PedButton = '1') then -- If PedButton Is Pressed
167                     NextState <= PedestrianPassing; -- Change State To
    PedestrianPassing --
168
169                  else
170                     NextState <= Car; -- If None Of The Above Occur, Hold State --
171                  end if;
172
173               when PedestrianPassing => -- When The State Is PedestrianPassing --
174                  TrainAction <= '0'; -- Set TrainAction Output To Be 0 --
175                  CarWaitOutput <= '0'; -- Set CarWaitOutput Output To Be 0 --
176                  PedWaitOutput <= '1'; -- Set PedWaitOutput Output To Be 1 --
177                  MotorEnabler <= '0';  -- Set MotorEnabler Output To Be 0 --
178
179                  if (Counter = "10100") then -- If Counter Reached "10100" --
180                     NextState <= Car; -- Change State To Car --
181
182                  elsif (TrainButton = '1') then -- If TrainButton Is Pressed --
183                        NextState <= GateClose; -- Change State To GateClose --
184
185                  else
186                     NextState <= PedestrianPassing; -- If None Of The Above Occur, Hold
    State --
187                  end if;
188
189               end case;
190
191      end process;
192   end Behavioral;
```