```vhdl
--------------------------------------------------------------------------------
-- Design: VHDL Class Example 6-2 : design of sequential logic / FSM          --
--                                                                            --
-- Author : Peter Roessler                                                    --
-- Date   : 21 09 2014                                                        --
-- File   : railroad_xing.vhd                                                 --
--------------------------------------------------------------------------------

library IEEE;
use IEEE.std_logic_1164.all;

entity railroad_xing is
port (train_approaches_i : in  std_logic;
      train_leaves_i     : in  std_logic;
      clk                : in  std_logic;
      reset              : in  std_logic;
      light_o            : out std_logic;
      gate_close_o       : out std_logic;
      gate_open_o        : out std_logic);
end railroad_xing;

architecture rtl of railroad_xing is

  type t_state is (OPENED_S, OPEN1_S, OPEN2_S, OPEN3_S, OPEN4_S, OPEN5_S,
                   CLOSED_ON_S, CLOSED_OFF_S,
                   CLOSE1_S, CLOSE2_S, CLOSE3_S, CLOSE4_S, CLOSE5_S);

  signal s_states : t_state;

begin
  p_railroad_xing : process (clk, reset)
  begin
    if reset = '1' then
      -- brings system to a safe state, under all pre-conditions ...
      light_o <= '1';
      gate_close_o <= '1';
      gate_open_o <= '0';
      s_states <= CLOSE1_S;
    elsif clk'event and clk = '1' then
      case s_states is
        ------------------------------------------------------------------------
        -- STATE "Gate is completely open"
        ------------------------------------------------------------------------
        when OPENED_S =>
          light_o <= '0';
          gate_close_o <= '0';
          gate_open_o <= '0';
          if train_approaches_i = '1' then
            light_o <= '1';
            gate_close_o <= '1';
            s_states <= CLOSE1_S;
          else
            s_states <= OPENED_S;
          end if;
        ------------------------------------------------------------------------
        -- STATE "Gate will be closed (1st second)"
        ------------------------------------------------------------------------
```

```vhdl
        when CLOSE1_S =>
          light_o <= '0';
          gate_close_o <= '1';
          gate_open_o <= '0';
          s_states <= CLOSE2_S;
        ------------------------------------------------------------------------
        -- STATE "Gate will be closed (2nd second)"
        ------------------------------------------------------------------------
        when CLOSE2_S =>
          light_o <= '1';
          gate_close_o <= '1';
          gate_open_o <= '0';
          s_states <= CLOSE3_S;
        ------------------------------------------------------------------------
        -- STATE "Gate will be closed (3rd second)"
        ------------------------------------------------------------------------
        when CLOSE3_S =>
          light_o <= '0';
          gate_close_o <= '1';
          gate_open_o <= '0';
          s_states <= CLOSE4_S;
        ------------------------------------------------------------------------
        -- STATE "Gate will be closed (4th second)"
        ------------------------------------------------------------------------
        when CLOSE4_S =>
          light_o <= '1';
          gate_close_o <= '0';
          gate_open_o <= '0';
          s_states <= CLOSED_ON_S;
        ------------------------------------------------------------------------
        -- STATE "Gate is completely closed, light is turned on"
        ------------------------------------------------------------------------
        when CLOSED_ON_S =>
          light_o <= '0';
          gate_close_o <= '0';
          gate_open_o <= '0';
          if train_leaves_i = '1' then
            -- jump to OPEN2_S to ensure blinking frequency of exactly 0.5 Hz
            gate_open_o <= '1';
            s_states <= OPEN2_S;
          else
            s_states <= CLOSED_OFF_S;
          end if;
        ------------------------------------------------------------------------
        -- STATE "Gate is completely closed, light is turned off"
        ------------------------------------------------------------------------
        when CLOSED_OFF_S =>
          light_o <= '1';
          gate_close_o <= '0';
          gate_open_o <= '0';
          if train_leaves_i = '1' then
            gate_open_o <= '1';
            s_states <= OPEN1_S;
          else
            s_states <= CLOSED_ON_S;
          end if;
        ------------------------------------------------------------------------
        -- STATE "Gate will be opened (1st second)"
        -----------------------------------------------------------------------
        when OPEN1_S =>
          light_o <= '0';
          gate_close_o <= '0';
          gate_open_o <= '1';
          s_states <= OPEN2_S;
        -----------------------------------------------------------------------
        -- STATE "Gate will be opened (2nd or 1st second, respectively)"
        -----------------------------------------------------------------------
        when OPEN2_S =>
          light_o <= '1';
          gate_close_o <= '0';
          gate_open_o <= '1';
          s_states <= OPEN3_S;
        -----------------------------------------------------------------------
        -- STATE "Gate will be opened (3rd or 2nd second, respectively)"
        -----------------------------------------------------------------------
        when OPEN3_S =>
          light_o <= '0';
          gate_close_o <= '0';
          gate_open_o <= '1';
          s_states <= OPEN4_S;
        -----------------------------------------------------------------------
        -- STATE "Gate will be opened (4th or 3rd second, respectively)"
        -----------------------------------------------------------------------
        when OPEN4_S =>
          light_o <= '1';
          gate_close_o <= '0';
          gate_open_o <= '1';
          s_states <= OPEN5_S;
        -----------------------------------------------------------------------
        -- STATE "Gate will be opened (5tt or 4th second, respectively)"
        -----------------------------------------------------------------------
        when OPEN5_S =>
          light_o <= '0';
          gate_close_o <= '0';
          gate_open_o <= '0';
          s_states <= OPENED_S;
        -----------------------------------------------------------------------
        -- all other cases ...
        -----------------------------------------------------------------------
        when others =>
          light_o <= '1';
          gate_close_o <= '1';
          gate_open_o <= '0';
          s_states <= CLOSE1_S;
      end case;
    end if;
  end process p_railroad_xing;
end rtl;

```

```
-------------------------------------------------------------------------------
-- Design: VHDL Class Example 6-2: design of sequential logic/FSM, testbench --
--                                                                           --
-- Author : Peter Roessler                                                   --
-- Date   : 21 09 2014                                                       --
-- File   : tb_railroad_xing.vhd                                             --
-------------------------------------------------------------------------------

library IEEE;
use IEEE.std_logic_1164.all;

entity tb_railroad_xing is end tb_railroad_xing;

architecture sim of tb_railroad_xing is

  -- Declaration of the component under test
  component railroad_xing
    port (
      train_approaches_i : in  std_logic;
      train_leaves_i     : in  std_logic;
      clk                : in  std_logic;
      reset              : in  std_logic;
      light_o            : out std_logic;
      gate_close_o       : out std_logic;
      gate_open_o        : out std_logic);
  end component;

  signal train_approaches_i : std_logic;
  signal train_leaves_i : std_logic;
  signal clk : std_logic;
  signal reset : std_logic;
  signal light_o : std_logic;
  signal gate_close_o : std_logic;
  signal gate_open_o : std_logic;

begin

  -- Instantiate the design under test
  i_railroad_xing : railroad_xing
    port map (
      train_approaches_i => train_approaches_i,
      train_leaves_i     => train_leaves_i,
      clk                => clk,
      reset              => reset,
      light_o            => light_o,
      gate_close_o       => gate_close_o,
      gate_open_o        => gate_open_o);

  -- Generate clock
  p_clk : process
  begin
    clk <= '0';
    wait for 0.5 sec;
    clk <= '1';
    wait for 0.5 sec;
  end process p_clk;

```

```
  p_stim : process
  begin
    -- Initial states of all input ports and reset signal
    train_approaches_i <= '0';
    train_leaves_i <= '0';
    reset <= '1';
    wait for 2 sec;
    -- Deassert reset
    reset <= '0';
    wait for 10 sec;
    -- Start applying test patters
    train_leaves_i <= '1';
    wait for 1 sec;
    train_leaves_i <= '0';
    wait for 10 sec;
    train_approaches_i <= '1';
    wait for 1 sec;
    train_approaches_i <= '0';
    wait for 10 sec;
    train_leaves_i <= '1';
    wait for 1 sec;
    train_leaves_i <= '0';
    wait for 10 sec;
    -- Stop simulation
    assert false report "END OF SIMULATION" severity failure;
  end process p_stim;

end sim;

```