```vhdl
--------------------------------------------------------------------------------
-- Company: 
-- Engineer:
--
-- Create Date:   12:41:02 05/20/2021
-- Design Name:   
-- Module Name:   C:/Xilinx/14.7/VHDL_Design_Project/TB_TrafficLights.vhd
-- Project Name:  VHDL_Design_Project
-- Target Device:  
-- Tool versions:  
-- Description:   
-- 
-- VHDL Test Bench Created by ISE for module: TrafficLights
-- 
-- Dependencies:
-- 
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
-- Notes: 
-- This testbench has been automatically generated using types std_logic and
-- std_logic_vector for the ports of the unit under test.  Xilinx recommends
-- that these types always be used for the top-level I/O of a design in order
-- to guarantee that the testbench will bind correctly to the post-implementation 
-- simulation model.
--------------------------------------------------------------------------------
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
 
-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--USE ieee.numeric_std.ALL;
 
ENTITY TB_TrafficLights IS
END TB_TrafficLights;
 
ARCHITECTURE behavior OF TB_TrafficLights IS 
 
    -- Component Declaration for the Unit Under Test (UUT)
 
    COMPONENT TrafficLights
    PORT(
         reset : IN  std_logic;
         clock : IN  std_logic;
         FourHzPulse : IN  std_logic; -- 4Hz Pulse --
         Flash : IN  std_logic; -- Train Button --
         TrafficGreen : IN  std_logic; -- Car Button --
         PedGreen : IN  std_logic; -- Pedestrian Button --
         MotorEnable : OUT  std_logic; -- Enable The Stepper Motor To Rotate --
         MotorClockwise : OUT  std_logic; -- Direction Of Rotation For Stepper Motor--
         HTrafficLightOutput : OUT  std_logic_vector(1 downto 0); -- Output for Horizontal Traffic Light --
         VTrafficLightOutput : OUT  std_logic_vector(1 downto 0) -- Output for Vertical Traffic Light --
        );
    END COMPONENT;
```

```vhdl


   --Inputs
   signal reset : std_logic := '0';
   signal clock : std_logic := '0';
   signal FourHzPulse : std_logic := '0';
   signal Flash : std_logic := '0';
   signal TrafficGreen : std_logic := '0';
   signal PedGreen : std_logic := '0';

   --Outputs
   signal MotorEnable : std_logic;
   signal MotorClockwise : std_logic;
   signal HTrafficLightOutput : std_logic_vector(1 downto 0);
   signal VTrafficLightOutput : std_logic_vector(1 downto 0);

   -- Clock period definitions
   constant clock_period : time := 10 ns;

BEGIN

   -- Instantiate the Unit Under Test (UUT)
   uut: TrafficLights PORT MAP (
          reset => reset,
          clock => clock,
          FourHzPulse => FourHzPulse,
          Flash => Flash,
          TrafficGreen => TrafficGreen,
          PedGreen => PedGreen,
          MotorEnable => MotorEnable,
          MotorClockwise => MotorClockwise,
          HTrafficLightOutput => HTrafficLightOutput,
          VTrafficLightOutput => VTrafficLightOutput
        );

   -- Clock process definitions
   clock_process :process
   begin
      clock <= '0';
      wait for clock_period/2;
      clock <= '1';
      wait for clock_period/2;
   end process;

   -- 4 Hz Pulse --
   FourHzPulse_process :process
   begin
      FourHzPulse <= '0';
      wait for 2.5 ns ;
      FourHzPulse <= '1';
      wait for clock_period/2;
   end process;

   -- Stimulus process
   stim_proc: process
   begin
      -- hold reset state for 100 ns.
      reset <= '1';
      wait for 100 ns;
      reset <= '0';

      -- insert stimulus here --

      -- When Train Button Is Pressed --
      wait for clock_period*10;
      Flash <= '1';
      TrafficGreen <= '0';
      PedGreen <= '0';

      -- When Car Button Is Pressed --
      wait for clock_period*10;
      Flash <= '0';
      TrafficGreen <= '1';
      PedGreen <= '0';

      -- When Pedestrian Button Is Pressed --
      wait for clock_period*10;
      Flash <= '0';
      TrafficGreen <= '0';
      PedGreen <= '1';

      -- When Train Button Is Pressed --
      wait for clock_period*10;
      Flash <= '1';
      TrafficGreen <= '0';
      PedGreen <= '0';

      wait;
   end process;

END;
```