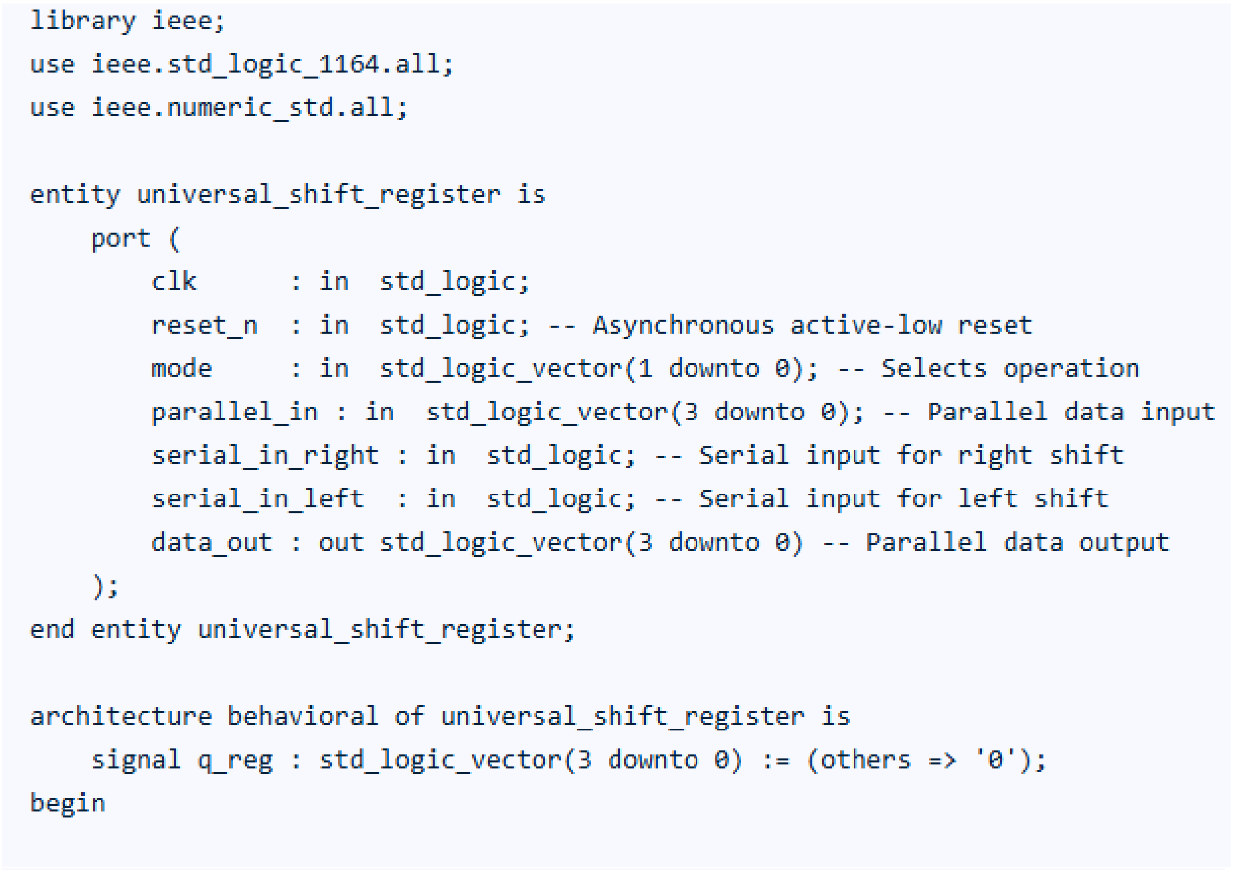
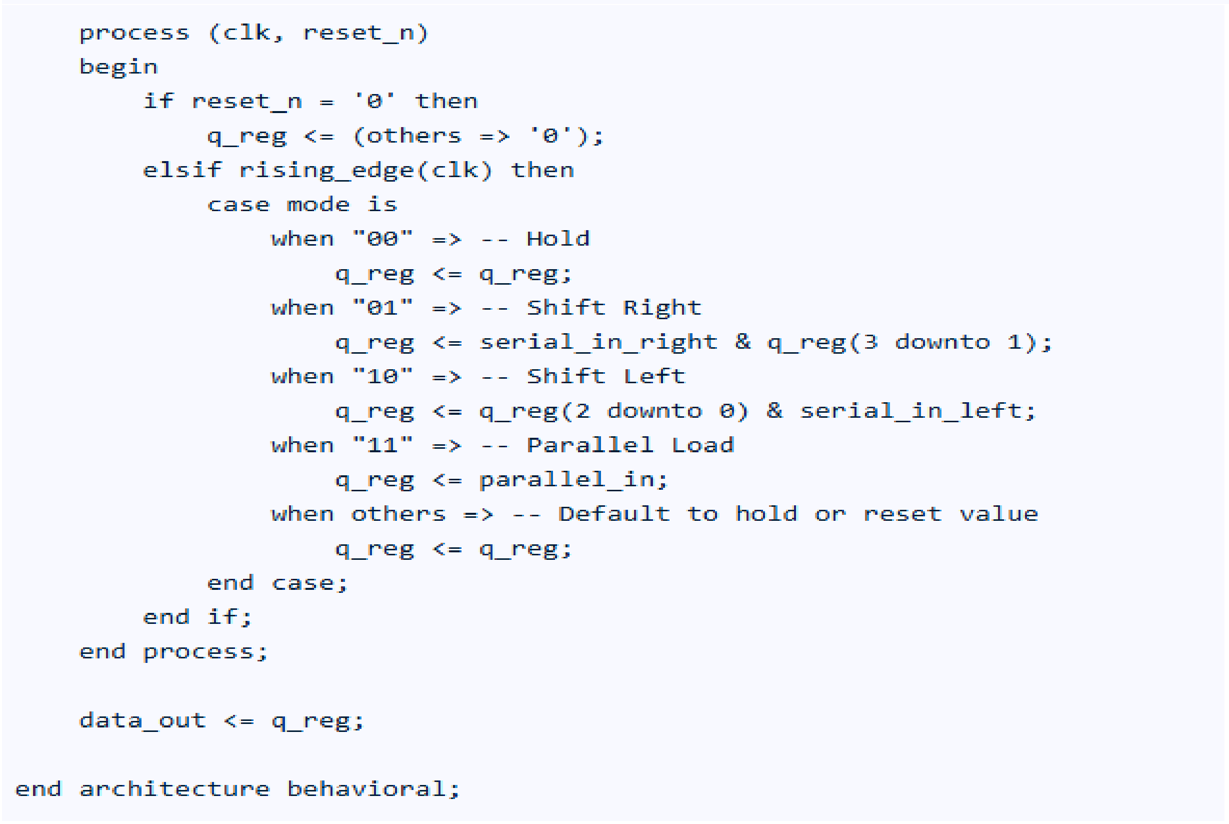
VHDL CODE



```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity universal_shift_register is
    port (
        clk       : in  std_logic;
        reset_n   : in  std_logic; -- Asynchronous active-low reset
        mode      : in  std_logic_vector(1 downto 0); -- Selects operation
        parallel_in : in  std_logic_vector(3 downto 0); -- Parallel data input
        serial_in_right : in  std_logic; -- Serial input for right shift
        serial_in_left  : in  std_logic; -- Serial input for left shift
        data_out : out std_logic_vector(3 downto 0) -- Parallel data output
    );
end entity universal_shift_register;

architecture behavioral of universal_shift_register is
    signal q_reg : std_logic_vector(3 downto 0) := (others => '0');
begin
```

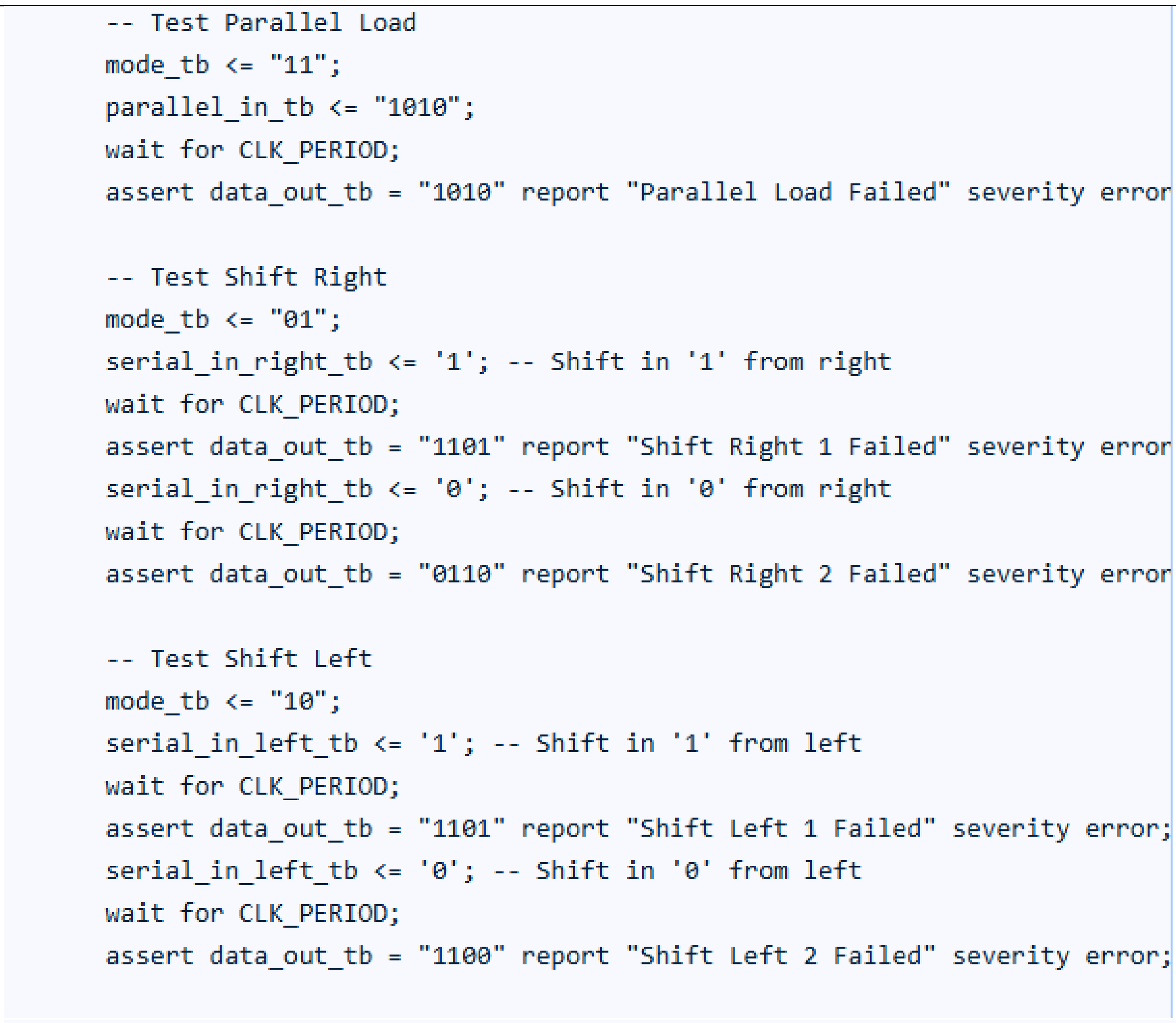


```
    process (clk, reset_n)
    begin
        if reset_n = '0' then
            q_reg <= (others => '0');
        elsif rising_edge(clk) then
            case mode is
                when "00" => -- Hold
                    q_reg <= q_reg;
                when "01" => -- Shift Right
                    q_reg <= serial_in_right & q_reg(3 downto 1);
                when "10" => -- Shift Left
                    q_reg <= q_reg(2 downto 0) & serial_in_left;
                when "11" => -- Parallel Load
                    q_reg <= parallel_in;
                when others => -- Default to hold or reset value
                    q_reg <= q_reg;
            end case;
        end if;
    end process;

    data_out <= q_reg;

end architecture behavioral;
```

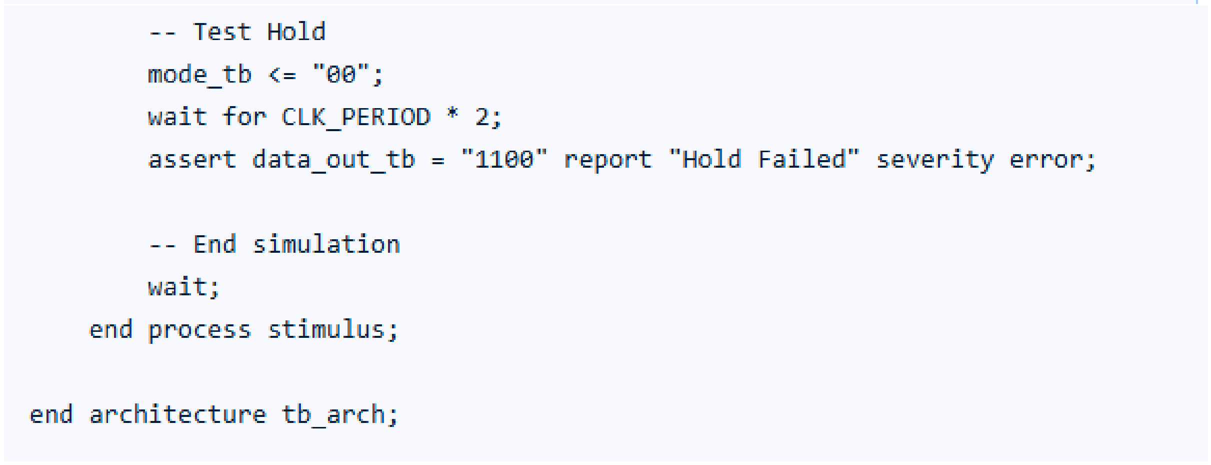
TEST BENCH

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity universal_shift_register_tb is
end entity universal_shift_register_tb;

architecture tb_arch of universal_shift_register_tb is

    -- Component declaration for the UUT
    component universal_shift_register
        port (
            clk       : in  std_logic;
            reset_n   : in  std_logic;
            mode      : in  std_logic_vector(1 downto 0);
            parallel_in : in  std_logic_vector(3 downto 0);
            serial_in_right : in  std_logic;
            serial_in_left  : in  std_logic;
            data_out : out std_logic_vector(3 downto 0)
        );
    end component;
    -- Signals for the testbench
    signal clk_tb      : std_logic := '0';
    signal reset_n_tb  : std_logic;
    signal mode_tb     : std_logic_vector(1 downto 0);
    signal parallel_in_tb : std_logic_vector(3 downto 0);
    signal serial_in_right_tb : std_logic;
    signal serial_in_left_tb  : std_logic;
    signal data_out_tb : std_logic_vector(3 downto 0);

    -- Clock period definition
    constant CLK_PERIOD : time := 10 ns;

begin

    -- Instantiate the Unit Under Test (UUT)
    uut : universal_shift_register
        port map (
            clk      => clk_tb,
            reset_n  => reset_n_tb,
            mode     => mode_tb,
            parallel_in => parallel_in_tb,
            serial_in_right => serial_in_right_tb,
            serial_in_left  => serial_in_left_tb,
            data_out => data_out_tb
        );
```

	```
	    -- Clock generation process
	    clk_gen : process
	    begin
	        loop
	            clk_tb <= '0';
	            wait for CLK_PERIOD / 2;
	            clk_tb <= '1';
	            wait for CLK_PERIOD / 2;
	        end loop;
	    end process clk_gen;

	    -- Test stimulus process
	    stimulus : process
	    begin
	        -- Initial reset
	        reset_n_tb <= '0';
	        mode_tb <= "00";
	        parallel_in_tb <= (others => '0');
	        serial_in_right_tb <= '0';
	        serial_in_left_tb <= '0';
	        wait for CLK_PERIOD * 2;
	        reset_n_tb <= '1';
	        wait for CLK_PERIOD;
	```



```
        -- Test Parallel Load
        mode_tb <= "11";
        parallel_in_tb <= "1010";
        wait for CLK_PERIOD;
        assert data_out_tb = "1010" report "Parallel Load Failed" severity error

        -- Test Shift Right
        mode_tb <= "01";
        serial_in_right_tb <= '1'; -- Shift in '1' from right
        wait for CLK_PERIOD;
        assert data_out_tb = "1101" report "Shift Right 1 Failed" severity error
        serial_in_right_tb <= '0'; -- Shift in '0' from right
        wait for CLK_PERIOD;
        assert data_out_tb = "0110" report "Shift Right 2 Failed" severity error

        -- Test Shift Left
        mode_tb <= "10";
        serial_in_left_tb <= '1'; -- Shift in '1' from left
        wait for CLK_PERIOD;
        assert data_out_tb = "1101" report "Shift Left 1 Failed" severity error;
        serial_in_left_tb <= '0'; -- Shift in '0' from left
        wait for CLK_PERIOD;
        assert data_out_tb = "1100" report "Shift Left 2 Failed" severity error;
```



```
        -- Test Hold
        mode_tb <= "00";
        wait for CLK_PERIOD * 2;
        assert data_out_tb = "1100" report "Hold Failed" severity error;

        -- End simulation
        wait;
    end process stimulus;

end architecture tb_arch;
```