

```
1  -----
2  -- Company:
3  -- Engineer:
4  --
5  -- Create Date:    15:42:20 08/24/2023
6  -- Design Name:
7  -- Module Name:    C:/Users/ETC-VLSI
LAB/Downloads/Xilinx_ISE_DS_14.7_1015_1_2/shiftregister/shiftregisterclk.vhd
8  -- Project Name:  shiftregister
9  -- Target Device:
10 -- Tool versions:
11 -- Description:
12 --
13 -- VHDL Test Bench Created by ISE for module: shiftregistrer
14 --
15 -- Dependencies:
16 --
17 -- Revision:
18 -- Revision 0.01 - File Created
19 -- Additional Comments:
20 --
21 -- Notes:
22 -- This testbench has been automatically generated using types std_logic and
23 -- std_logic_vector for the ports of the unit under test.  Xilinx recommends
24 -- that these types always be used for the top-level I/O of a design in order
25 -- to guarantee that the testbench will bind correctly to the post-implementation
26 -- simulation model.
27  -----
28  LIBRARY ieee;
29  USE ieee.std_logic_1164.ALL;
30
31  -- Uncomment the following library declaration if using
32  -- arithmetic functions with Signed or Unsigned values
33  --USE ieee.numeric_std.ALL;
34
35  ENTITY shiftregisterclk IS
36  END shiftregisterclk;
37
38  ARCHITECTURE behavior OF shiftregisterclk IS
39
40      -- Component Declaration for the Unit Under Test (UUT)
41
42      COMPONENT shiftregistrer
43      PORT (
44          s : IN  std_logic_vector(3 downto 0);
45          m : IN  std_logic_vector(1 downto 0);
46          y : OUT std_logic_vector(3 downto 0);
47          clk : IN  std_logic;
48          rst : IN  std_logic
49      );
50      END COMPONENT;
51
52
53  --Inputs
54  signal s : std_logic_vector(3 downto 0) := (others => '0');
55  signal m : std_logic_vector(1 downto 0) := (others => '0');
56  signal clk : std_logic := '0';
```

```
57     signal rst : std_logic := '0';
58
59     --Outputs
60     signal y : std_logic_vector(3 downto 0);
61
62     -- Clock period definitions
63     constant clk_period : time := 10 ns;
64
65 BEGIN
66
67     -- Instantiate the Unit Under Test (UUT)
68     uut: shiftregistrer PORT MAP (
69         s => s,
70         m => m,
71         y => y,
72         clk => clk,
73         rst => rst
74     );
75
76     -- Clock process definitions
77     clk_process :process
78     begin
79         clk <= '1';
80         wait for clk_period/2;
81         clk <= '0';
82         wait for clk_period/2;
83     end process;
84
85     -- Stimulus process
86     stim_proc: process
87     begin
88         -- hold reset state for 100 ns.
89         rst<='1';
90         wait for 10 ns;
91         rst<='0';
92         m<="00";
93         s<="0001";
94         wait for 80 ns;
95         s<="1110";
96         wait for 80 ns;
97         s<="0000";
98         wait for 80 ns;
99
100
101         m<="01";
102         s<="0001";
103         wait for 80 ns;
104         s<="1110";
105         wait for 80 ns;
106         s<="0000";
107         wait for 80 ns;
108
109         m<="10";
110         s<="0001";
111         wait for 80 ns;
112         s<="1110";
113         wait for 80 ns;
```

```
114         s<="0000";
115         wait for 80 ns;
116
117         m<="11";
118         s<="0001";
119         wait for 80 ns;
120         s<="1110";
121         wait for 80 ns;
122         s<="0000";
123         wait for 80 ns;
124
125
126         --wait for clk_period*10;
127
128         -- insert stimulus here
129
130         wait;
131     end process;
132
133 END;
```