Practical No.03: Universal Shift Regiter

Code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
use IEEE.NUMERIC_STD.ALL;

entity usr1 is
  Port (
      clk : in  STD_LOGIC;
      clk_div : inout std_logic;
      rst : in  STD_LOGIC;
      load: in  STD_LOGIC;
      mode : in  STD_LOGIC_VECTOR(1 downto 0);  -- mode selection input
      si : in  STD_LOGIC;  -- serial input
      pi : in  STD_LOGIC_VECTOR(3 downto 0);  -- parallel input
      so : out  STD_LOGIC;  -- serial output
      po : out  STD_LOGIC_VECTOR(3 downto 0)  -- parallel output
   );
end usr1;

architecture Behavioral of usr1 is
```

```
    signal shift_reg : STD_LOGIC_VECTOR(3 downto 0);
    signal counter: std_logic_vector(26 downto 0):=( others=>'0');
begin
    process(clk_div, rst,load)
    begin
        if rst = '1' then
            shift_reg <= (others => '0');
        elsif clk_div'event and clk_div= '1' then
            case mode is
              when "00" =>  -- SISO mode
                shift_reg( 3 downto 1) <= shift_reg(2 downto 0);
                  shift_reg(0)<=si;
                  so <= shift_reg(3);
              when "01" =>  -- SIPO mode
              shift_reg( 3 downto 1) <= shift_reg(2 downto 0);
                  shift_reg(0)<=si;
                   po <= shift_reg;
             when "10" =>  -- PISO mode
                    if(load='0') then
                            shift_reg <= pi;
                     else
             shift_reg( 3 downto 1) <= shift_reg(2 downto 0);
                     end if;
                     so <= shift_reg(3);
```

```
            when "11" =>  -- PIPO mode
               po<= pi;

            when others =>
               null;
         end case;
      end if;
   end process;

  -- so <= shift_reg(3);
  -- po <= shift_reg;

  process(clk)
   begin
      if clk'event and clk = '1' then
         if rst = '1' then
            counter <= (others => '0');
         else
            counter <= counter + '1';
         end if;
      end if;
   end process;

clk_div<= counter(26);
```

```
end Behavioral;
```

TB-Code

```
library IEEE;
 use IEEE.STD_LOGIC_1164.ALL;
 use IEEE.NUMERIC_STD.ALL;
 entity usr1tb is
 end usr1tb;
 architecture Behavioral of usr1tb is
 signal clk,clk_div : STD_LOGIC := '0';
 signal rst : STD_LOGIC := '0';
 signal load : STD_LOGIC := '0';
 signal mode : STD_LOGIC_VECTOR(1 downto 0) := "00";
 signal si : STD_LOGIC := '0';
 signal pi : STD_LOGIC_VECTOR(3 downto 0) := (others => '0');
 signal so : STD_LOGIC;
 signal po : STD_LOGIC_VECTOR(3 downto 0);
 component usr1
 Port (
 clk : in STD_LOGIC;
 clk_div : inout std_logic;
 rst : in STD_LOGIC;
 load: in STD_LOGIC;
 mode:in STD_LOGIC_VECTOR(1 downto 0);
```

```
 si : in STD_LOGIC;
 pi : in STD_LOGIC_VECTOR(3 downto 0);
so : out STD_LOGIC;
 po : out STD_LOGIC_VECTOR(3 downto 0)
 );
 end component;
 begin
 uut : usr1
 Port Map(
 clk => clk,
 clk_div => clk_div,
 rst => rst,
 load => load,
 mode=>mode,
 si => si,
 pi => pi,
 so => so,
 po =>po
 );
 clk_process : process
 begin
 clk <= '0';
 wait for 10 ns;
 clk <= '1';
```

```
 wait for 10 ns;
 end process;
 stim_proc : process
 begin-- Reset the UUT
 rst <= '1';
 wait for 20 ns;
 rst <= '0';-- Test SISO mode
 mode<="00";
 si <= '1';
 wait for 200 ns;
 si <= '0';
 wait for 200 ns;-- Test SIPO mode
 mode<="01";
 si <= '1';
 wait for 100 ns;
 si <= '0';
 wait for 200 ns;-- Test PISO mode
 mode<="10";
 load<='0';
 pi <= "1110";
 wait for 100 ns;
 load<='1';
wait for 300 ns;-- Test PIPO mode
 mode<="11";
```

```
pi <= "1010";
wait for 50 ns;
pi <= "1100";
wait for 50 ns;
wait;
end process;
end Behavioral;
```

Port Names	FPGA Pins
clk	W5
clk_div	L1
rst	U1
load	W2
mode1	R2
mode0	T1
si	W15
so	W18
pi3	W17
pi2	W16
pi1	V16
pi0	V17
po3	V19
po2	U19
po1	E19
po0	U16