

# Assignment 4

IT451 : COA LAB

**ANINDYA KUNDU**

IT, 4<sup>th</sup> Semester

ID: 510817020 (Hx-19)

31-03-2019

## Question 1

Design and simulate a 5-bit bidirectional shift register. Use 'DIRECTION' as a control input as STD\_LOGIC with its value '1' for right shift and '0' for left shift. Synthesize it for Spartan 6. Report device utilization.

**VHDL Module:**     *mod\_bidirShiftReg.vhd*

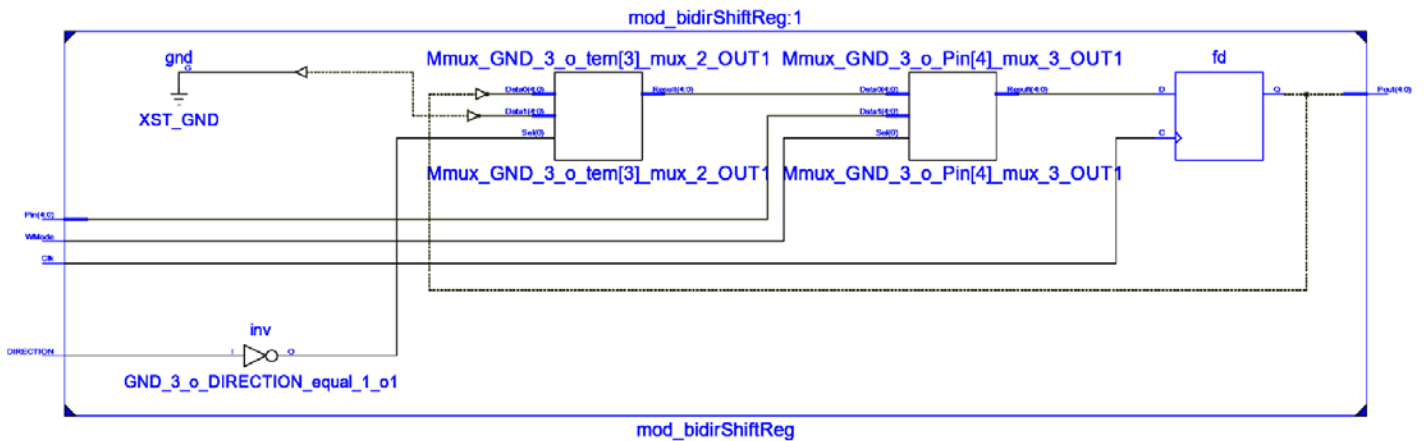
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity mod_bidirShiftReg is
    Port ( Clk          : in  STD_LOGIC;
           WMode        : in  STD_LOGIC;
           DIRECTION    : in  STD_LOGIC;
           Pin          : in  STD_LOGIC_VECTOR (4 downto 0);
           Pout         : out STD_LOGIC_VECTOR (4 downto 0));
end mod_bidirShiftReg;

architecture Behavioral of mod_bidirShiftReg is
begin

    process(Clk, WMode, DIRECTION, Pin)
        variable tem : STD_LOGIC_VECTOR (4 downto 0);
    begin
        if rising_edge(Clk) then
            if WMode = '1' then tem := Pin;
            else
                case DIRECTION is
                    when '0' =>
                        tem(4) := tem(3); tem(3) := tem(2);
                        tem(2) := tem(1); tem(1) := tem(0);
                        tem(0) := '0';
                    when '1' =>
                        tem(0) := tem(1); tem(1) := tem(2);
                        tem(2) := tem(3); tem(3) := tem(4);
                        tem(4) := '0';
                    when others =>
                        tem := "00000";
                end case;
            end if;
        end if;
        Pout <= tem;
    end process;
end Behavioral;
```

## RTL Schematic:



## Device Utilisation Summary:

Device Utilization Summary (estimated values)				
Logic Utilization	Used	Available	Utilization	
Number of Slice Registers	5	54576	0%	
Number of Slice LUTs	5	27288	0%	
Number of fully used LUT-FF pairs	0	10	0%	
Number of bonded IOBs	13	296	4%	
Number of BUFG/BUFGCTRLs	1	16	6%	

## VHDL Test Bench: *tb\_bidirShiftReg.vhd*

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
```

```
ENTITY tb_bidirShiftReg IS
END tb_bidirShiftReg;
```

```
ARCHITECTURE behavior OF tb_bidirShiftReg IS
```

```
    COMPONENT mod_bidirShiftReg
```

```
    PORT(
```

```
        Clk          : IN  std_logic;
        WMode        : IN  std_logic;
        DIRECTION    : IN  std_logic;
        Pin           : IN  std_logic_vector(4 downto 0);
        Pout          : OUT std_logic_vector(4 downto 0)
    );
```

```
END COMPONENT;
```

```
signal clk      : std_logic := '0';
signal wmode    : std_logic := '0';
signal dir      : std_logic := '0';
```

```

signal pin          : std_logic_vector(4 downto 0) := (others => '0');
signal pout         : std_logic_vector(4 downto 0);

constant clk_per    : time := 100 ns;

BEGIN
  uut: mod_bidirShiftReg PORT MAP (
    Clk          => clk,
    WMode        => wmode,
    DIRECTION    => dir,
    Pin          => pin,
    Pout         => pout
  );

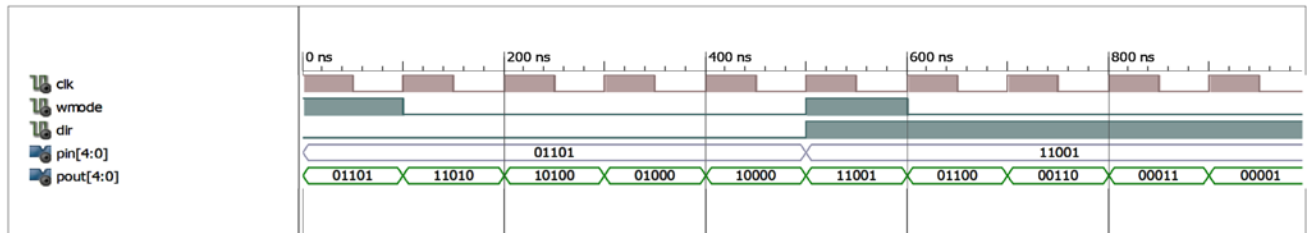
  Clk_proc: process
  begin
    clk <= '1';
    wait for clk_per/2;
    clk <= '0';
    wait for clk_per/2;
  end process;

  write_mode: process
  begin
    wmode <= '1'; wait for clk_per;
    wmode <= '0'; wait for clk_per * 4;
    wmode <= '1'; wait for clk_per;
    wmode <= '0'; wait;
  end process;

  dir_mode: process
  begin
    dir <= '0', '1' after clk_per * 5;
    wait;
  end process;

  stim_proc: process
  begin
    pin <= "01101", "11001" after clk_per * 5;
    wait;
  end process;
END;
```

## Simulation:



## Question 2

Design and simulate a UART transmitter. Synthesize it for Spartan 6. Report device utilization.

**VHDL Module:**     *mod\_uartTransmitter.vhd*

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

entity mod_uartTransmitter is
    Port ( Clk      : in  STD_LOGIC;
          Wmode    : in  STD_LOGIC;
          Pin      : in  STD_LOGIC_VECTOR (3 downto 0);
          Sout     : out STD_LOGIC);
end mod_uartTransmitter;

architecture Behavioral of mod_uartTransmitter is

begin

process(Clk, WMode, Pin)
    variable tem : STD_LOGIC_VECTOR (6 downto 0);
    variable cnt : INTEGER;

begin
    if rising_edge(Clk) then
        if WMode = '1' then
            Sout <= '0';
            tem(0) := '0';          tem(6) := '1';
            tem(1) := Pin(0); tem(2) := Pin(1);
            tem(3) := Pin(2); tem(4) := Pin(3);

            cnt := 0;
            for i in 0 to 3 loop
                if Pin(i) = '1' then cnt := cnt + 1;
                end if;
            end loop;

            if cnt rem 2 = 1 then tem(5) := '1';
            else                  tem(5) := '0';
            end if;

            Sout <= tem(0);
```

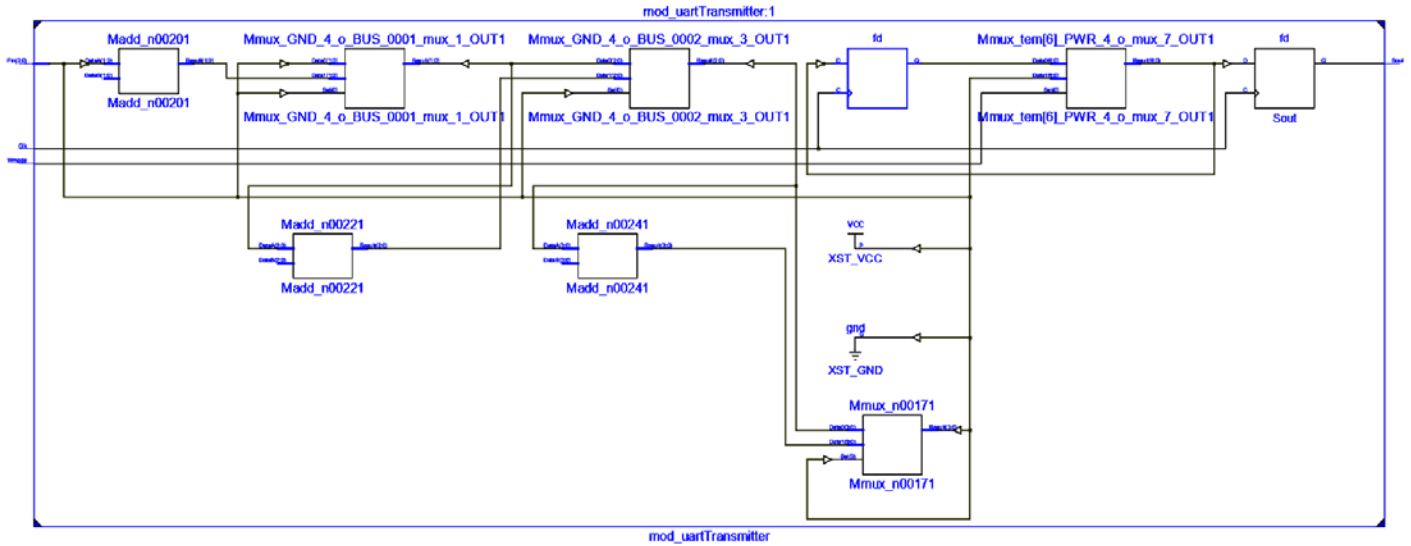
```

    for i in 0 to 5 loop
        tem(i) := tem(i + 1);
    end loop;
    end if;
end process;

end Behavioral;

```

### RTL Schematic:



### Device Utilisation Summary:

Device Utilization Summary (estimated values)				<a href="#">[-]</a>
Logic Utilization	Used	Available	Utilization	
Number of Slice Registers	7	54576	0%	
Number of Slice LUTs	7	27288	0%	
Number of fully used LUT-FF pairs	0	14	0%	
Number of bonded IOBs	7	296	2%	
Number of BUFG/BUFGCTRLs	1	16	6%	

**VHDL Test Bench:**      *tb\_uartTransmitter.vhd*

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY tb_uartTransmitter IS
END tb_uartTransmitter;

ARCHITECTURE behavior OF tb_uartTransmitter IS
    COMPONENT mod_uartTransmitter
    PORT(
        Clk      : IN  std_logic;
        Wmode     : IN  std_logic;
        Pin       : IN  std_logic_vector(3 downto 0);
        Sout      : OUT std_logic
    );
    END COMPONENT;

    signal clk      : std_logic := '0';
    signal wmode    : std_logic := '0';
    signal pin      : std_logic_vector(3 downto 0) := (others => '0');
    signal sout     : std_logic;
    constant clk_per : time := 100 ns;

BEGIN
    uut: mod_uartTransmitter PORT MAP (
        Clk      => clk,
        Wmode     => wmode,
        Pin       => pin,
        Sout      => sout
    );

    Clk_process: process
    begin
        clk <= '1';
        wait for clk_per/2;
        clk <= '0';
        wait for clk_per/2;
    end process;

    write_mode: process
    begin
        wmode <= '1'; wait for clk_per;
        wmode <= '0'; wait for clk_per * 7;
        wmode <= '1'; wait for clk_per;
```



```

        wmode <= '0'; wait;
    end process;

    stim_proc: process
    begin
        pin <= "1101", "0110" after clk_per * 8;
        wait;
    end process;
END;

```

### Simulation:

