EEE 4484

Lab 05

Submitted by:

Sidratul Muntaha
CSE-A
Std_ID: 180041118

TASK 01: FULL ADDER

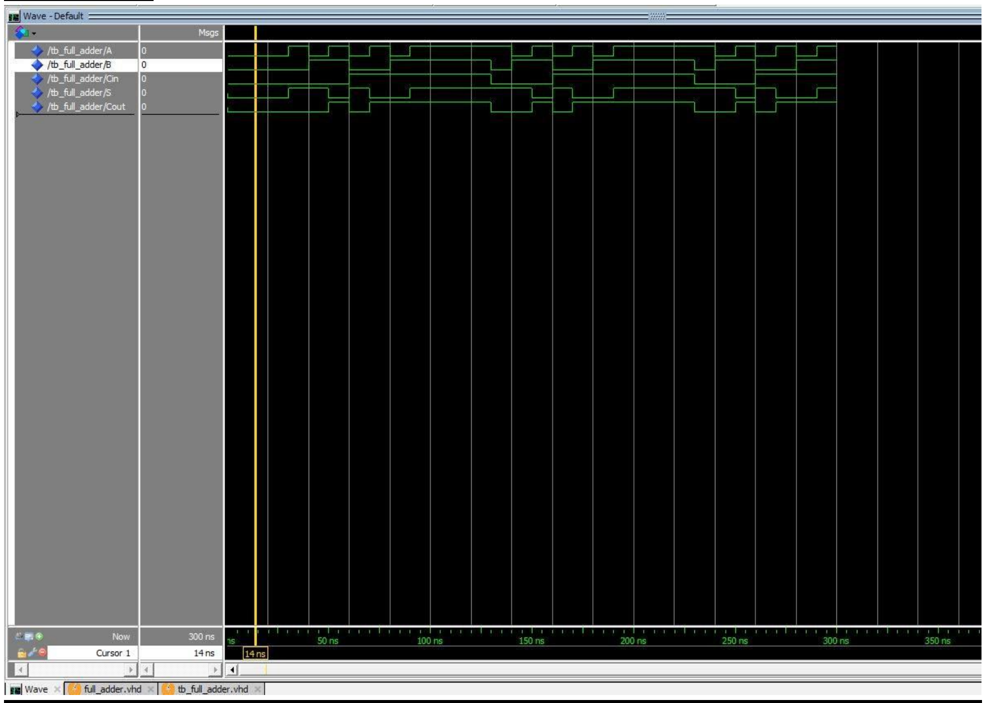
VHDL Code

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

entity full_adder is
 Port ( A : in STD_LOGIC;
 B : in STD_LOGIC;
 Cin : in STD_LOGIC;
 S : out STD_LOGIC;
 Cout : out STD_LOGIC);
end full_adder;

architecture gate_level of full_adder is

begin

 S <= A XOR B XOR Cin ;
 Cout <= (A AND B) OR (Cin AND A) OR (Cin AND B)

; end gate_level;
```

TESTBENCH Code

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

ENTITY tb_full_adder IS
END tb_full_adder;

ARCHITECTURE behavior OF tb_full_adder IS

 --Component Declaration for the Unit Under Test (UUT)

 COMPONENT full_adder
 PORT(
 A : IN std_logic;
 B : IN std_logic;
 Cin : IN std_logic;
 S : OUT std_logic;
 Cout : OUT
 std_logic );
 END COMPONENT;

 --Inputs
 signal A : std_logic := ‘0’;
 signal B : std_logic := ‘0’;
 signal Cin : std_logic := ‘0’;

  --Outputs
 signal S : std_logic;
 signal Cout : std_logic;

BEGIN

 --Instantiate the Unit Under Test (UUT)
 uut: full_adder PORT MAP (
 A=>A,
 B=>B,
 Cin => Cin,
 S=>S,
 Cout => Cout
 );

  --Stimulus process
 stim_proc: process
 begin
  --hold reset state for 100 ns.
 Wait for 100 ns;

  --insert stimulus here
 A <= ‘1’;
 B <= ‘0’;
 Cin <= ‘0’;
 wait for 10 ns;

 A <= ‘0’;
 B <= ‘1’;
 Cin <= ‘0’;
 wait for 10 ns;

 A <= ‘1’;
 B <= ‘1’;
 Cin <= ‘0’;
 wait for 10 ns;

 A <= ‘0’;
 B <= ‘0’;
 Cin <= ‘1’;
 wait for 10 ns;
 A <= ‘1’;
 B <= ‘0’;
 Cin <= ‘1’;
 wait for 10 ns;

 A <= ‘0’;
 B <= ‘1’;
 Cin <= ‘1’;
 wait for 10 ns;

 A <= ‘1’;
 B <= ‘1’;
 Cin <= ‘1’;
 wait for 10 ns;

 end process;

END;
```

OUTPUT



```
/tb_full_adder/A     0
/tb_full_adder/B     0
/tb_full_adder/Cin   0
/tb_full_adder/S     0
/tb_full_adder/Cout  0
```

TASK 02: 1 TO 8 DEMULTIPLEXER

VHDL Code

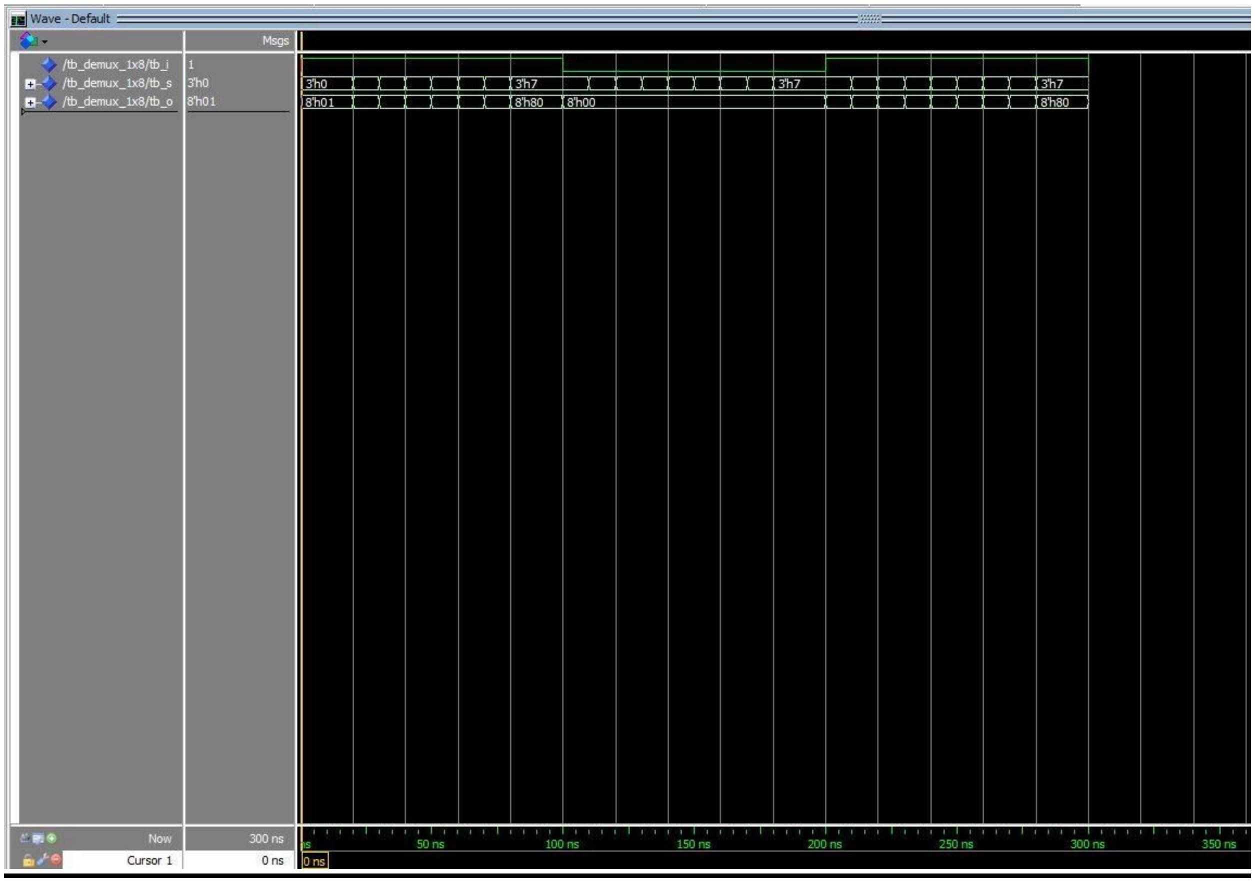
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity demux_1x8 is
port(
        i:in std_logic;
        s:in std_logic_vector(2 downto 0);
        o:out std_logic_vector(7 downto 0)
);
end demux_1x8;

architecture behavioral of demux_1x8
is begin
        o(0)<=i when s=”000” else’0’;
        o(1)<=i when s=”001” else’0’;
        o(2)<=i when s=”010” else’0’;
        o(3)<=i when s=”011” else’0’;
        o(4)<=i when s=”100” else’0’;
        o(5)<=i when s=”101” else’0’;
        o(6)<=i when s=”110” else’0’;
        o(7)<=i when s=”111” else’0’;
end behavioral;
```

TESTBENCH Code

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

entity tb_demux_1x8 is
end tb_demux_1x8;

architecture behavioral of tb_demux_1x8 is

component demux_1x8
port(
        i:in std_logic;
        s:in std_logic_vector(2 downto 0);
        o:out std_logic_vector(7 downto 0)
);
end component;

-- Inputs
```

```
signal tb_i : std_logic := '0';
signal tb_s : std_logic_vector (2 downto 0) := (others => '0');

-- Outputs
signal tb_o : std_logic_vector (7 downto 0) := (others => '0');

begin

uut: demux_1x8 port map (
        i => tb_i,
        s => tb_s,
        o => tb_o
);


-- stimulus process
stim_process:
process begin
        tb_i<='1';
        wait for 10 ns;
        tb_s <= "000";
        wait for 10 ns;
        tb_s <= "001";
        wait for 10 ns;
        tb_s <= "010";
        wait for 10 ns;
        tb_s <= "011";
        wait for 10 ns;
        tb_s <= "100";
        wait for 10 ns;
        tb_s <= "101";
        wait for 10 ns;
        tb_s <= "110";
        wait for 10 ns;
        tb_s <= "111";
        wait for 20 ns;
        tb_i<='0'; wait
        for 10 ns; tb_s
        <= "000"; wait
        for 10 ns; tb_s
        <= "001"; wait
        for 10 ns; tb_s
        <= "010"; wait
        for 10 ns; tb_s
        <= "011"; wait
        for 10 ns; tb_s
        <= "100"; wait
        for 10 ns; tb_s
        <= "101"; wait
        for 10 ns; tb_s
        <= "110"; wait
        for 10 ns; tb_s
        <= "111";
        wait for 20 ns;
end process;

end architecture behavioral;
```

OUTPUT



TASK 03: 4 BIT ALU

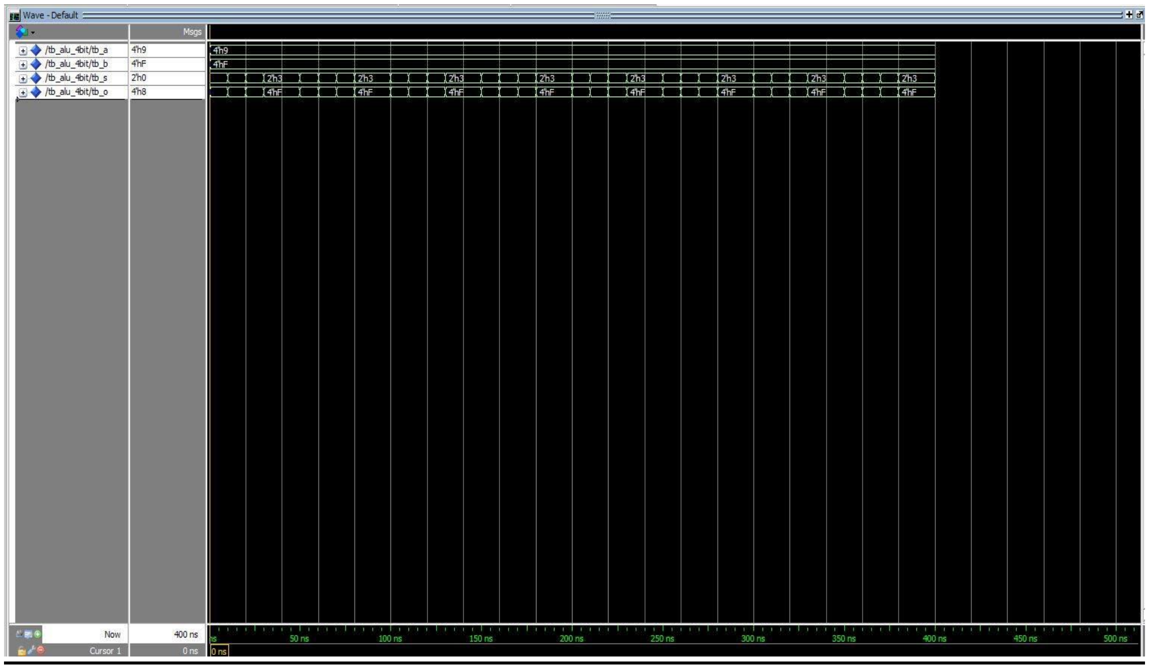
VHDL Code

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


entity alu_op is
 Port (
 inp_a : in signed(3 downto 0);
 inp_b : in signed(3 downto 0);
 sel  : in STD_LOGIC_VECTOR (2 downto 0);
 out_alu : out signed(3 downto 0));
end alu_op;

architecture Behavioral of alu_op is
begin
process(inp_a, inp_b, sel)
begin
case sel is
 when “000” =>
 out_alu<= inp_a + inp_b;  --addition
 when “001” =>
 out_alu<= inp_a – inp_b; --subtraction
 when “010” =>
 out_alu<= inp_a – 1;  --sub 1
 when “011” =>
 out_alu<= inp_a + 1;  --add 1
 when “100” =>
 out_alu<= inp_a and inp_b; -- AND gate
 when “101” =>
 out_alu<= inp_a or inp_b;  --OR gate
 when “110” =>
 out_alu<= not inp_a ;  --NOT gate
 when “111” =>
 out_alu<= inp_a xor inp_b; --XOR gate
 when others =>
 NULL;
end case;

end process;

end Behavioral;
```

TESTBENCH Code

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

ENTITY tb_alu_op IS
END tb_alu_op;

ARCHITECTURE behavior OF tb_alu_op IS

  --Component Declaration for the Unit Under Test (UUT)

 COMPONENT alu_op
 PORT(
 inp_a : IN signed(3 downto 0);
 inp_b : IN signed(3 downto 0);
 sel : IN std_logic_vector(2 downto 0);
 out_alu : OUT signed(3 downto 0)
 );
 END COMPONENT;


  --Inputs
 signal inp_a : signed(3 downto 0) := (others => '0');
 signal inp_b : signed(3 downto 0) := (others => '0');
 signal sel : std_logic_vector(2 downto 0) := (others => '0');

 -- Outputs
 signal out_alu : signed(3 downto 0);

BEGIN

 --Instantiate the Unit Under Test (UUT)
 uut: alu_op PORT MAP (
 inp_a => inp_a,
 inp_b => inp_b,
 sel => sel,
 out_alu => out_alu
 );

  --Stimulus process
 stim_proc: process
 begin
  -- hold reset state for 100 ns.

 Wait for 100 ns;
 -- insert stimulus here

 inp_a <= “1001”;
 inp_b <= “1111”;

 sel <= “000”;
 wait for 100 ns;
 sel <= “001”;
 wait for 100 ns;
 sel <= “010”;
 wait for 100 ns;
 sel <= “011”;
 wait for 100 ns;
 sel <= “100”;
 wait for 100 ns;
 sel <= “101”;
 wait for 100 ns;
 sel <= “110”;
 wait for 100 ns;
 sel <= “111”;
 end process;

END;
```

OUTPUT



```
Wave - Default
                       Msgs
/tb_alu_4bit/tb_a      4'h9
/tb_alu_4bit/tb_b      4'hF
/tb_alu_4bit/tb_s      2'h0
/tb_alu_4bit/tb_o      4'h8

Now                    400 ns
Cursor 1               0 ns
```

TASK 04: 5-BIT SHIFT REGISTER

VHDL Code

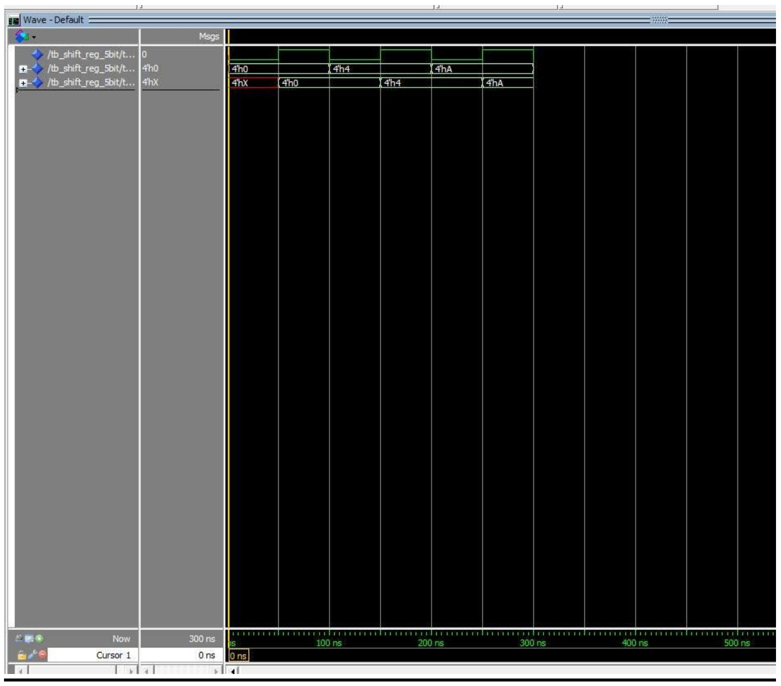
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity shift_reg_5bit is
port(
        clk : in std_logic;
        D: in std_logic_vector(3 downto 0);
        Q: out std_logic_vector(3 downto 0)
);
end shift_reg_5bit;

architecture behavioral of shift_reg_5bit is

begin
process (clk,D)
        begin
        if (clk’event and clk=’1’) then Q <=
        D; end if;
end process;

end behavioral;
```

TESTBENCH Code

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

entity tb_shift_reg_5bit is
end tb_shift_reg_5bit;

architecture behavioral of tb_shift_reg_5bit is

component shift_reg_5bit
port(
        clk : in std_logic;
        D  :  in  std_logic_vector(3  downto  0);
        Q  : out std_logic_vector(3 downto 0)
);
end component;

signal tb_clk : std_logic;
signal tb_D  : std_logic_vector(3 downto 0);
signal tb_Q  : std_logic_vector(3 downto 0);
constant clk_period : time := 100 ns;
```

```
begin
 uut: shift_reg_5bit port map(
        clk => tb_clk,
        D => tb_D,
        Q => tb_Q
);

-- clock process


clk_process: process
begin
        tb_clk <= ‘0’;
        wait for clk_period/2;
        tb_clk <= ‘1’;
        wait for clk_period/2;
end process;

-- stimulus process
stim_process:
process begin
        tb_D<=”0000”;
        wait for 100 ns;
        tb_D <= “0100”;
        wait for 100 ns;
        tb_D <= “1010”;
        wait for 100 ns;
        tb_D <= “0110”;
        wait for 100 ns;
end process;

end architecture behavioral;
```

OUTPUT


Wave - Default
Msgs
/tb_shift_reg_5bit/t... 0
/tb_shift_reg_5bit/t... 4'h0
/tb_shift_reg_5bit/t... 4'hX
4'h0
4'h4
4'hA
4'hX
4'h0
4'h4
4'hA
Now 300 ns
Cursor 1 0 ns
100 ns
200 ns
300 ns
400 ns
500 ns
0 ns