**EXPERIMENT 6**

**Aim:** Design of Arithmetic Logic Shift Unit.

**Exercise#1** : Design and verify N-bit, 8-operations (given in Table) Arithmetic Logic Shift Unit as shown in block diagram of figure-2.

**Design Code (Arithmetic):**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity Arithmetic\_component is

generic (n: integer := 3);

Port ( A : in STD\_LOGIC\_VECTOR (n-1 downto 0);

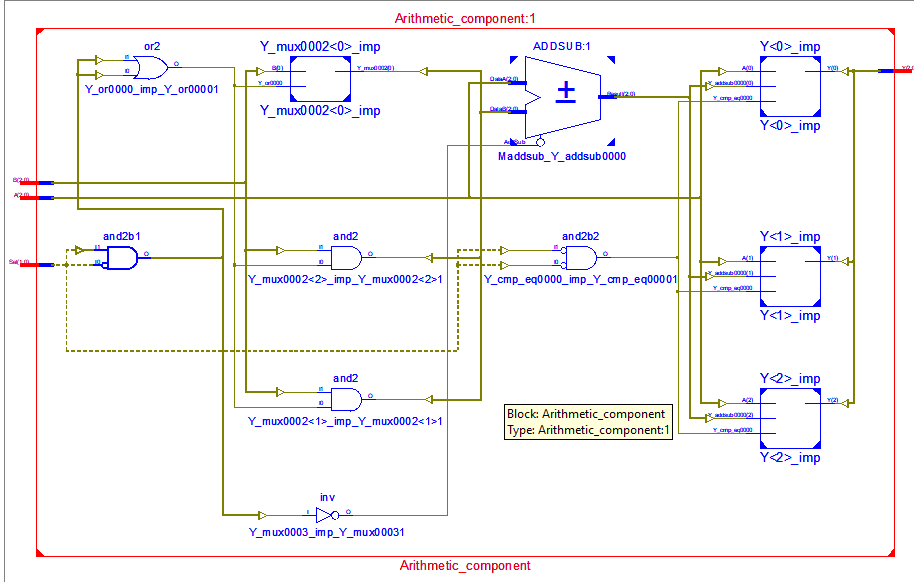
B : in STD\_LOGIC\_VECTOR (n-1 downto 0);

Sel : in STD\_LOGIC\_VECTOR (1 downto 0);

Y : out STD\_LOGIC\_VECTOR (n-1 downto 0));

end Arithmetic\_component;

architecture Behavioral of Arithmetic\_component is

begin

process (A,B,Sel)

begin

if(Sel = "00") then

Y <= A;

elsif (Sel = "01") then

Y <= A-B;

elsif (Sel = "10") then

Y <= A+B;

else

Y<= A-1;

end if;

end process;

end Behavioral;

**Design Code (Logical):**

library IEEE;

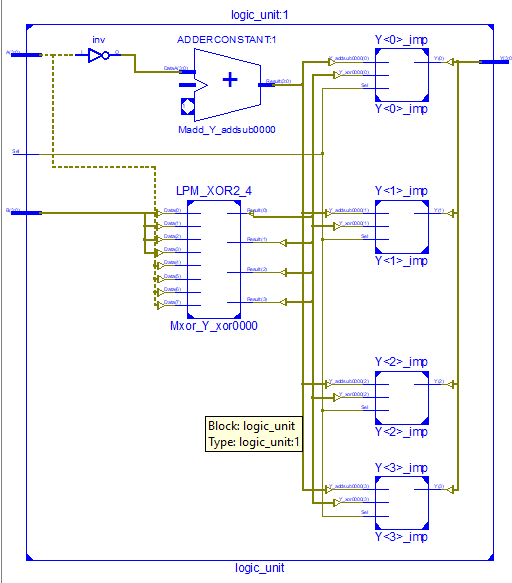
use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity logic\_unit is

generic (n: integer := 4);

 Port ( A : in STD\_LOGIC\_VECTOR (n-1 downto 0);

B : in STD\_LOGIC\_VECTOR (n-1 downto 0);

Sel : in STD\_LOGIC;

Y : out STD\_LOGIC\_VECTOR (n-1 downto 0));

end logic\_unit;

architecture Behavioral of logic\_unit is

begin

process (A,B,Sel)

begin

if(Sel = '0') then

Y <= A xor B;

else

Y <= not(A)+1;

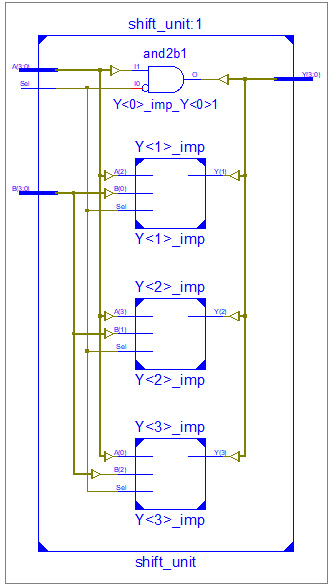
end if;

end process;

end Behavioral;

**Design Code (Shift):**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity shift\_unit is

generic (n: integer := 4);

Port ( A : in STD\_LOGIC\_VECTOR (n-1 downto 0);

B : in STD\_LOGIC\_VECTOR (n-1 downto 0);

Sel : in STD\_LOGIC;

Y : out STD\_LOGIC\_VECTOR (n-1 downto 0));

end shift\_unit;

architecture Behavioral of shift\_unit is

begin

process (A,B,Sel)

begin

if (Sel ='0') then

Y <= A(0) & A(n-1 downto 1);

else

Y <= b(n-2 downto 0)& '0';

end if;

end process;

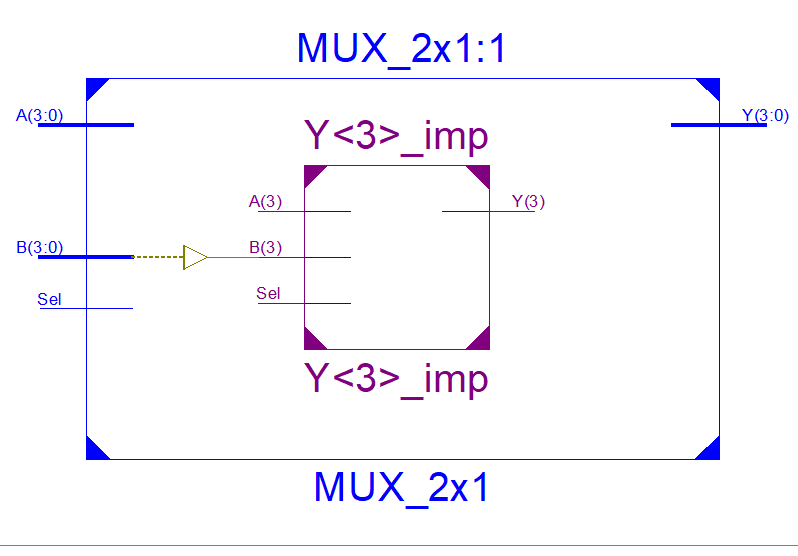
end Behavioral;

**Design Code (2x1 MUX):**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity MUX\_2x1 is

generic (n: integer := 4);

Port ( A : in STD\_LOGIC\_VECTOR (n-1 downto 0);

B : in STD\_LOGIC\_VECTOR (n-1 downto 0);

Sel : in STD\_LOGIC;

Y : out STD\_LOGIC\_VECTOR (n-1 downto 0));

end MUX\_2x1;

architecture Behavioral of MUX\_2x1 is

begin

process (A,B,Sel)

begin

if(Sel = '0') then

Y <= A;

else

Y<= B;

end if;

end process;

end Behavioral;

**Design Code:**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity ALU\_design is

generic (n: integer := 4);

Port ( A : in STD\_LOGIC\_VECTOR (n-1 downto 0);

B : in STD\_LOGIC\_VECTOR (n-1 downto 0);

Sel : in STD\_LOGIC\_VECTOR (2 downto 0);

Y : out STD\_LOGIC\_VECTOR (n-1 downto 0));

end ALU\_design;

architecture Structural of ALU\_design is

component Arithmetic\_component is

generic (n: integer := 4);

Port ( A : in STD\_LOGIC\_VECTOR (n-1 downto 0);

B : in STD\_LOGIC\_VECTOR (n-1 downto 0);

Sel : in STD\_LOGIC\_VECTOR (1 downto 0);

Y : out STD\_LOGIC\_VECTOR (n-1 downto 0));

end component;

component logic\_unit is

generic (n: integer := 4);

Port ( A : in STD\_LOGIC\_VECTOR (n-1 downto 0);

B : in STD\_LOGIC\_VECTOR (n-1 downto 0);

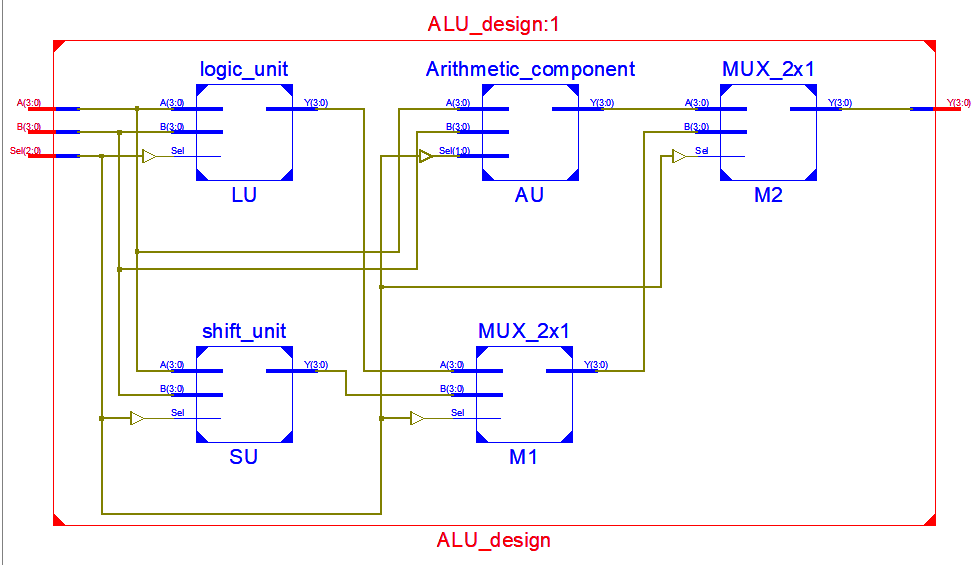
Sel : in STD\_LOGIC;

Y : out STD\_LOGIC\_VECTOR (n-1 downto 0));

end component;

component shift\_unit is

generic (n: integer := 4);

 Port ( A : in STD\_LOGIC\_VECTOR (n-1 downto 0);

B : in STD\_LOGIC\_VECTOR (n-1 downto 0);

Sel : in STD\_LOGIC;

Y : out STD\_LOGIC\_VECTOR (n-1 downto 0));

end component;

component MUX\_2x1 is

generic (n: integer := 4);

Port ( A : in STD\_LOGIC\_VECTOR (n-1 downto 0);

B : in STD\_LOGIC\_VECTOR (n-1 downto 0);

Sel : in STD\_LOGIC;

Y : out STD\_LOGIC\_VECTOR (n-1 downto 0));

end component;

signal S1, S2, S3, S4 :STD\_LOGIC\_VECTOR (n-1 downto 0);

begin

AU:Arithmetic\_component port map (A,B, sel(1 downto 0), S1);

LU: logic\_unit port map(A, B, Sel(0), S2);

SU: shift\_unit port map(A, B, Sel(0), S3);

M1: MUX\_2x1 port map (S2, S3, Sel (1), S4);

M2: MUX\_2x1 port map (S1, S4, Sel (2), Y);

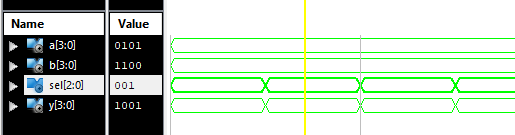
end Structural;

**Test Bench Code:**

A <= "0101";

B <= "1100";

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";

wait for 100 ns;