**Experiment-3**

**Aim:**

1. To write and simulate the VHDL codes for the following basic combinational logic using dataflow modelling.

1. 4:2 priority encoder (using if-else)
2. 1:8 Demux (using case)
3. BCD to Seven Segment decoder (using case)
4. Implement 8:1 Mux using if-else and case statements. Compare the hardware generated (RTL Schematic)

**Software required:**

1.Quartus II

**VHDL code:**

**Code A:**

library ieee;

use ieee.std\_logic\_1164.all;

Entity priority\_encoder\_4\_2 is

port (I: in std\_logic\_vector(3 downto 0);

En: in std\_logic;

y: out std\_logic\_vector(1 downto 0));

end priority\_encoder\_4\_2;

Architecture behave of priority\_encoder\_4\_2 is

begin

process(I,En)

begin

if En = '1' then

if I(3) = '1' then

y<="11";

elsif I(2) = '1' then

y<="10";

elsif I(1) = '1' then

y<="01";

elsif I(0) = '1' then

y<="00";

else

y<="00";

end if;

else

y<="00";

end if;

end process;

end behave ;

**Code B:**

library ieee;

use ieee.std\_logic\_1164.all;

Entity demux\_1\_8 is

port (I: in std\_logic;

En: in std\_logic;

s: in std\_logic\_vector(2 downto 0);

y: out std\_logic\_vector(7 downto 0));

end demux\_1\_8;

Architecture behave of demux\_1\_8 is

begin

process(I,En,s)

begin

if En = '1' then

case s is

when "000" => y <= ("0000000" & I);

when "001" => y <= ("000000" & I & '0');

when "010" => y <= ("00000" & I & "00");

when "011" => y <= ("0000" & I & "000");

when "100" => y <= ("000" & I & "0000");

when "101" => y <= ("00" & I & "00000");

when "110" => y <= ('0' & I & "000000");

when "111" => y <= (I & "0000000");

when others => y <= "00000000";

end case;

else

y<= "00000000";

end if;

end process;

end behave ;

**Code C:**

library ieee;

use ieee.std\_logic\_1164.all;

Entity bcd\_2\_sevenSegment is

port (I: in std\_logic\_vector(3 downto 0);

En: in std\_logic;

y: out std\_logic\_vector(6 downto 0));

end bcd\_2\_sevenSegment;

Architecture behave of bcd\_2\_sevenSegment is

begin

process(I,En)

begin

if En = '1' then

case I is

when "0000" => y <= "1111110";

when "0001" => y <= "0110000";

when "0010" => y <= "1101101";

when "0011" => y <= "1111001";

when "0100" => y <= "0110011";

when "0101" => y <= "1011011";

when "0110" => y <= "1011111";

when "0111" => y <= "1110000";

when "1000" => y <= "1111111";

when "1001" => y <= "1111011";

when others => y <= "0000000";

end case;

else

y<= "0000000";

end if;

end process;

end behave ;

**Code D.a:**

library ieee;

use ieee.std\_logic\_1164.all;

Entity mux\_8\_1\_case is

port (y: out std\_logic;

En: in std\_logic;

s: in std\_logic\_vector(2 downto 0);

I: in std\_logic\_vector(7 downto 0));

end mux\_8\_1\_case;

Architecture behave of mux\_8\_1\_case is

begin

process(I,En,s)

begin

if En = '1' then

case s is

when "000" => y <= I(0);

when "001" => y <= I(1);

when "010" => y <= I(2);

when "011" => y <= I(3);

when "100" => y <= I(4);

when "101" => y <= I(5);

when "110" => y <= I(6);

when "111" => y <= I(7);

when others => y <= '0';

end case;

else

y<= '0';

end if;

end process;

end behave ;

**Code D.b:**

library ieee;

use ieee.std\_logic\_1164.all;

Entity mux\_8\_1\_if is

port (I: in std\_logic\_vector(7 downto 0);

s: in std\_logic\_vector(2 downto 0);

En: in std\_logic;

y: out std\_logic);

end mux\_8\_1\_if;

Architecture behave of mux\_8\_1\_if is

begin

process(I,En,s)

begin

if En = '1' then

if s = "000" then

y<=I(0);

elsif s = "001" then

y<=I(1);

elsif s = "010" then

y<=I(2);

elsif s = "011" then

y<=I(3);

elsif s = "100" then

y<=I(4);

elsif s = "101" then

y<=I(5);

elsif s = "110" then

y<=I(6);

elsif s = "111" then

y<=I(7);

else

y<='1';

end if;

else

y<='0';

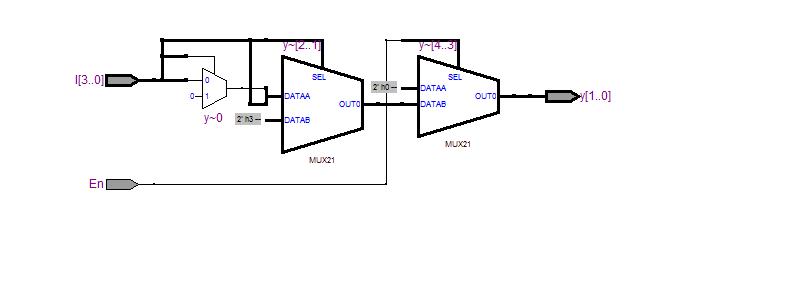
end if;

end process;

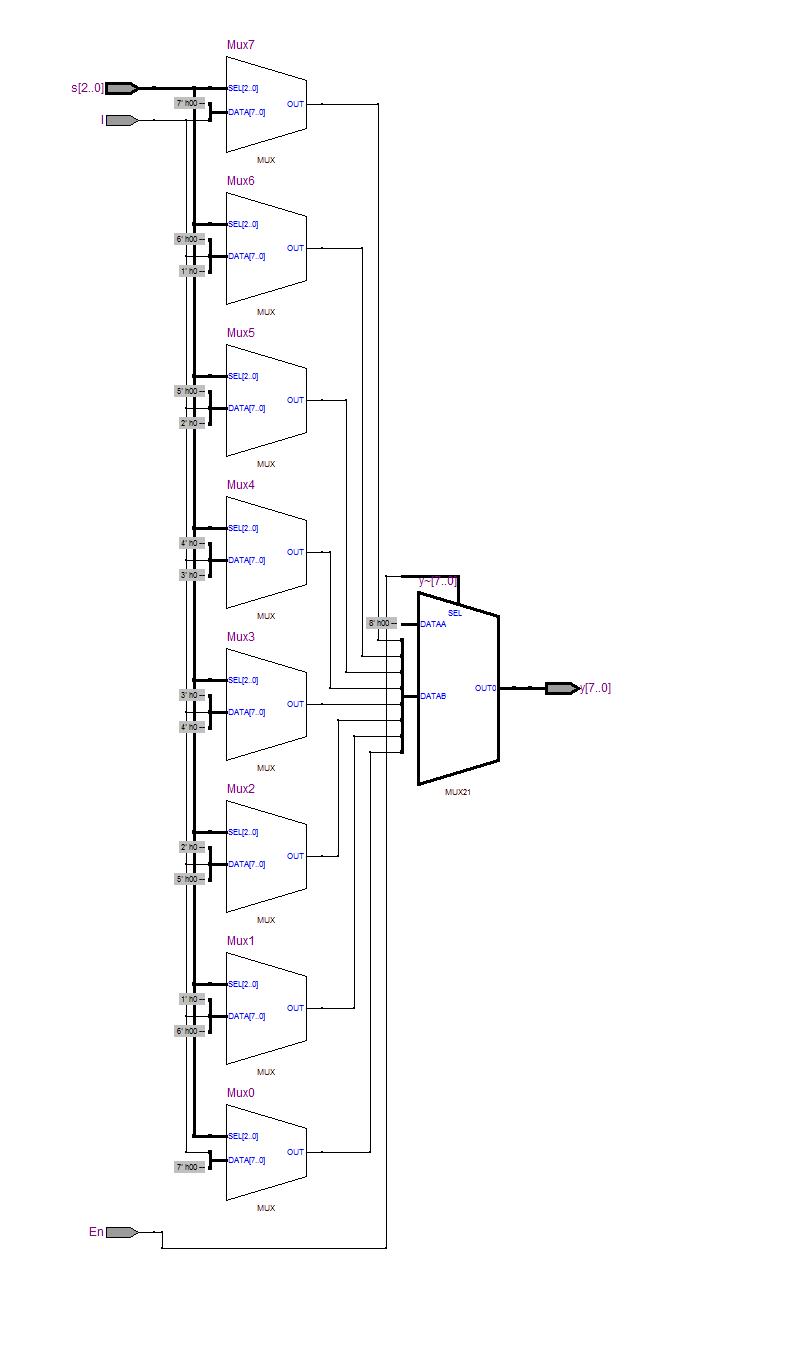
end behave ;

**RTL schematic:**

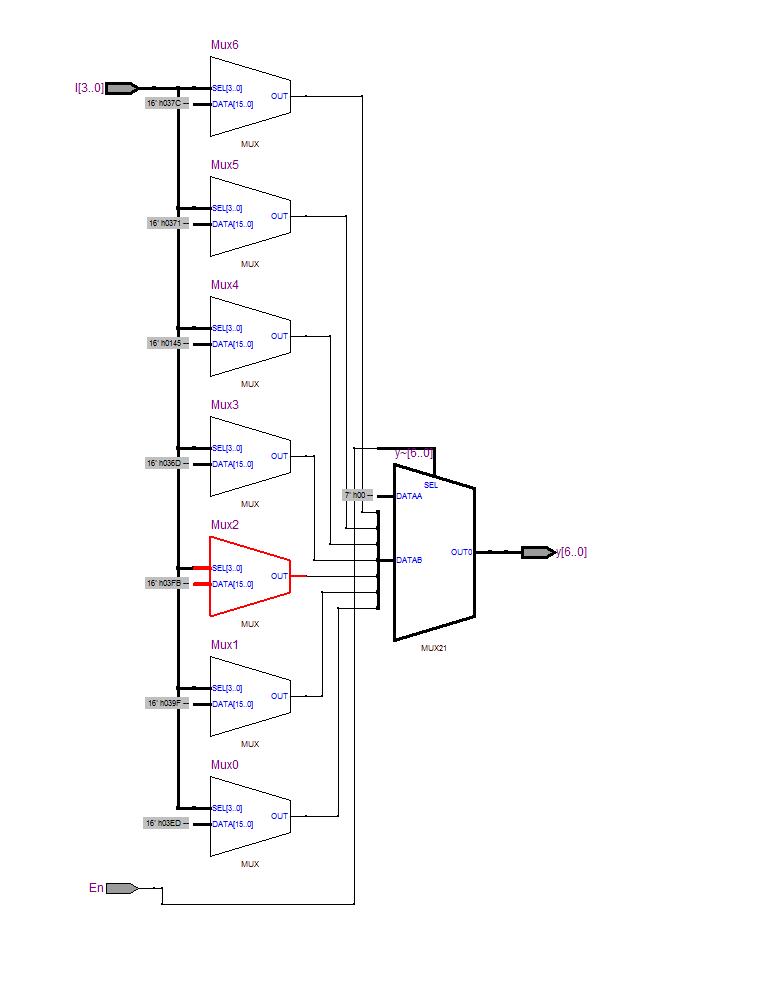
**RTL A:**



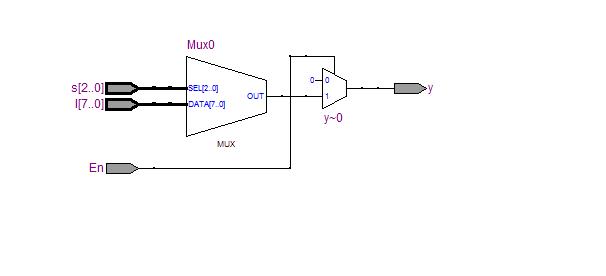
**RTL B:**



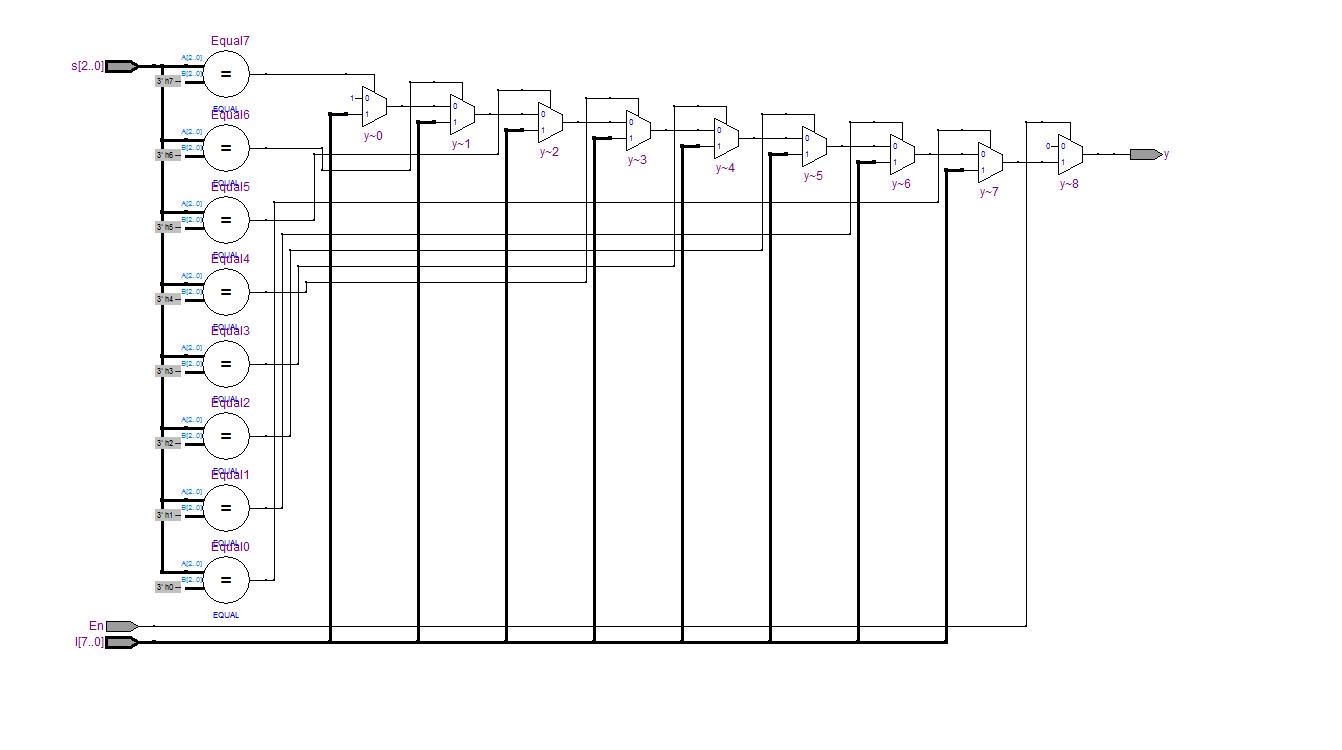
**RTL C:**

****

**RTL D.a:**

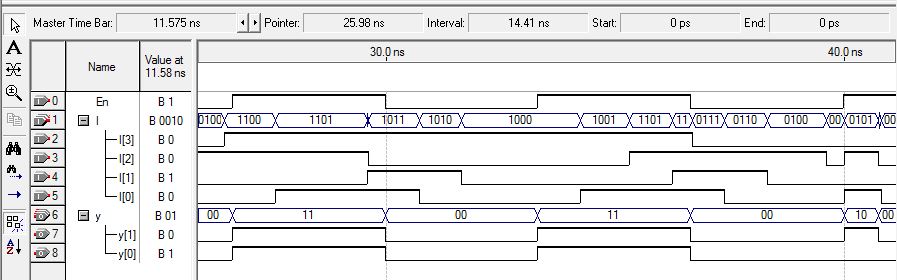
****

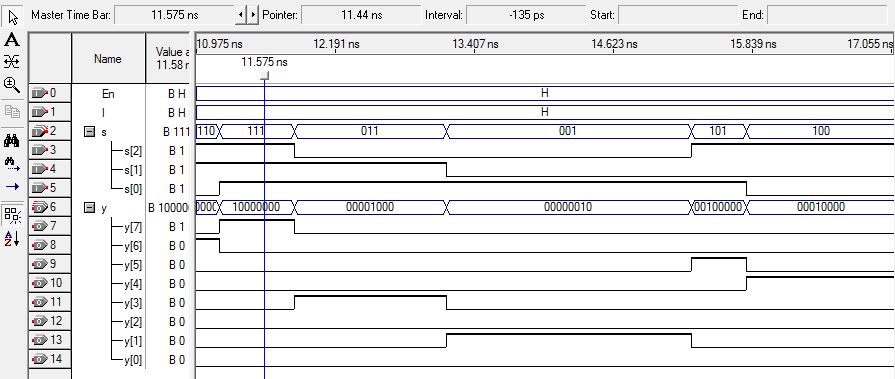
**RTL D.b:**



**Waveforms:**

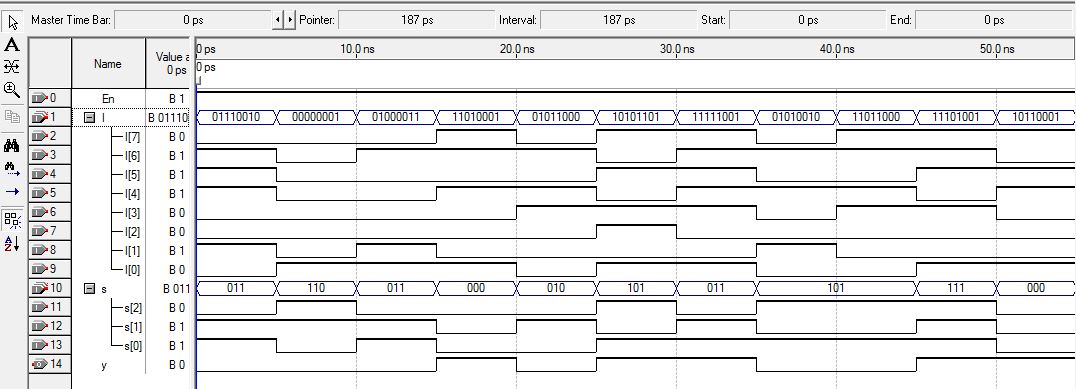
**Waveform A:**

****

**Waveform B: **

**Waveform C:**

****

**Waveform D.a:**

**Result:**

Hence the VHDL codes were written and the simulation was successful for the given basic combinational logic. Waveforms and RTL schematic were recorded and verified.

**Conclusion:**

It was observed that code containing **if-else** statements required more hardware than code containing **case** statements.