CS2052 Computer Architecture

Department of Computer Science and Engineering, University of Moratuwa

Lab 4: Combinational Circuits

Student Index No : *190019B*

Prepared By : *Abinesh Thaventhirarajah*

Prepared for : CS2052

Date of performance : 03/02/2021

Date of submission : 03/02/2021

Contents

01. Introduction

A decoder converts binary data from *n* coded inputs to a maximum of 2*n* unique outputs. The decoder that we are going to build also has an enable pin/input. Enable input must be on for the decoder to function, otherwise we assume its outputs as a "disabled" output. A multiplex receives binary data from 2n lines and connect them to a single output line based on a given *n*-bit selection. We will also add an enable pin to the multiplexer.

In this lab we will design a decoder and a multiplexer. Decoders and multiplexers are 2 of the key components of a microprocessor. After completing the lab, we will be able to:

- design and develop a 3-to-8 decoder using schematics
- design and develop a 8-to-1 multiplexer using schematics
- verify their functionality via simulation and on the development board

02. 2 to 4 Decoder

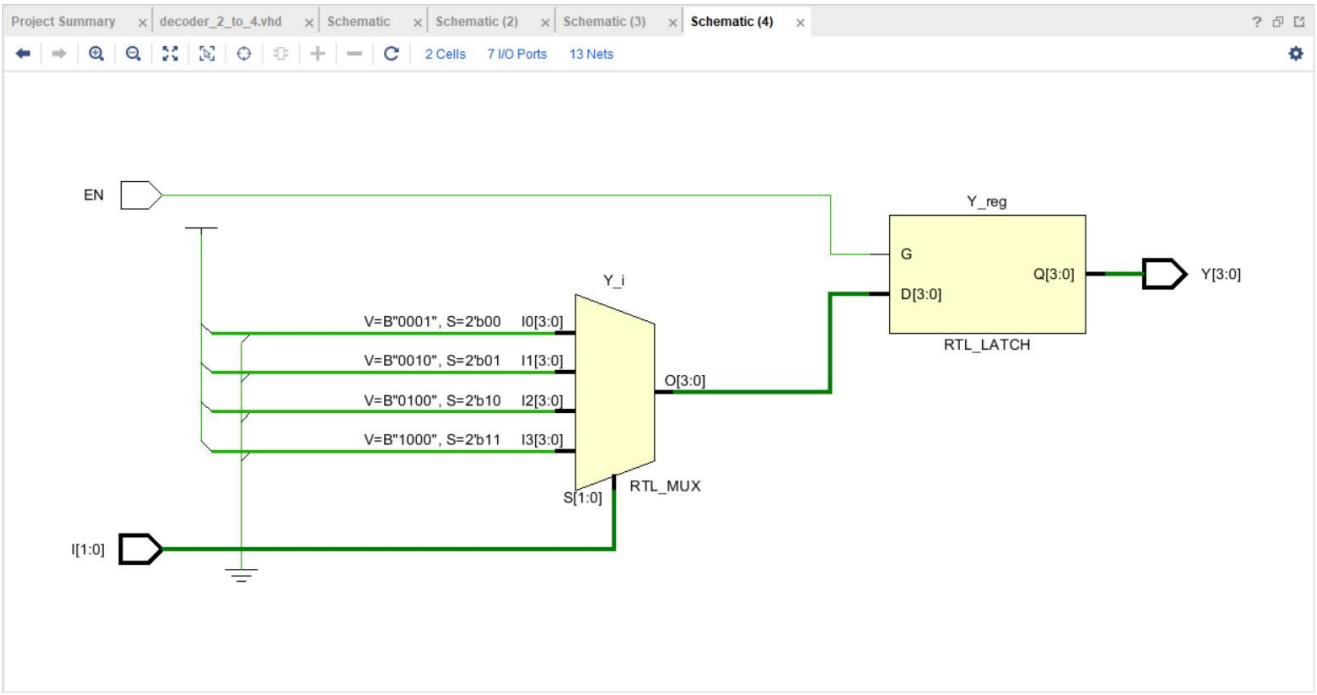
Truth table 2 to 4 Decoder (with enable switch)

EN	A	B	D3	D2	D1	D0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

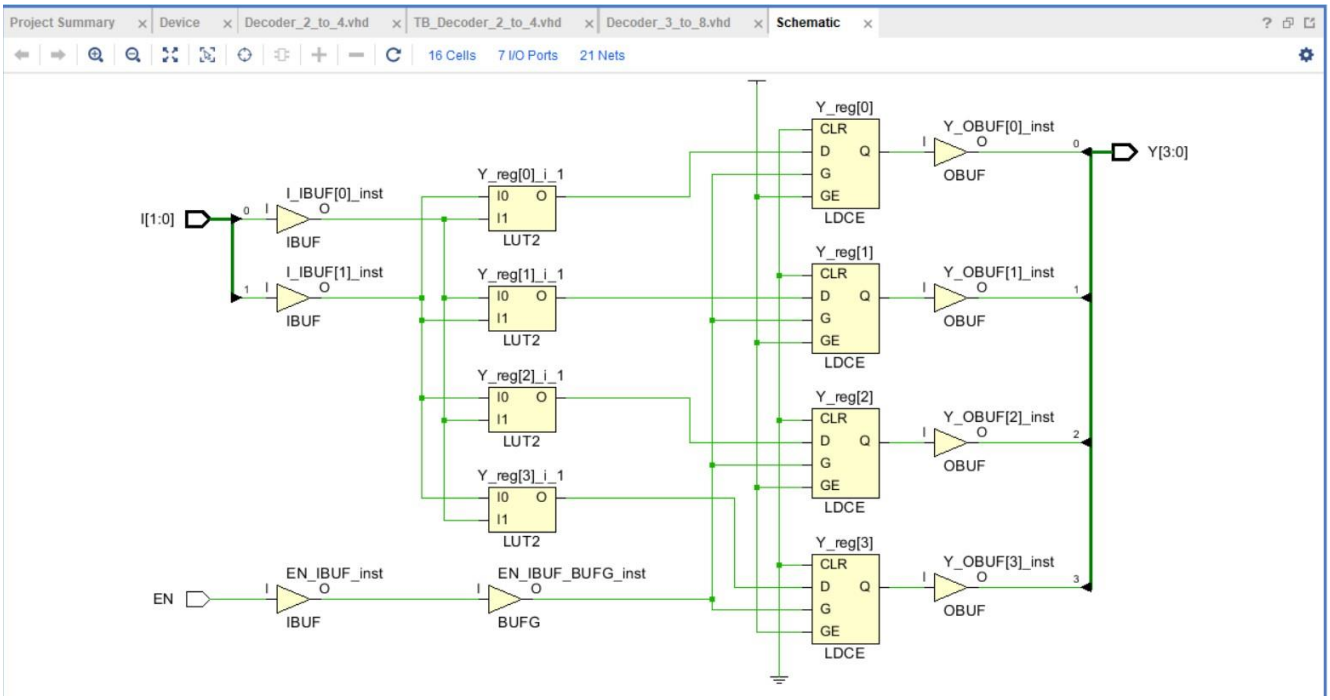
VHDL code for 2 to 4 Decoder (Decoder_2_to_4)

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

entity Decoder_2_to_4 is
    Port ( I : in STD_LOGIC_VECTOR (1 downto 0);
           EN : in STD_LOGIC;
           Y : out STD_LOGIC_VECTOR (3 downto 0));
end decoder_2_to_4;

architecture Behavioral of Decoder_2_to_4 is

begin

process(I,EN)
    begin
        if (EN='1') then
            case I is
                when "00" => Y <= "0001";
                when "01" => Y <= "0010";
                when "10" => Y <= "0100";
                when "11" => Y <= "1000";
                when others => null;
            end case;
        end if;
end process;
end Behavioral;
```

Elaborated Design View for 2 to 4 Decoder (RTL Schematic view)



Schematic Representation of the 2 to 4 Decoder

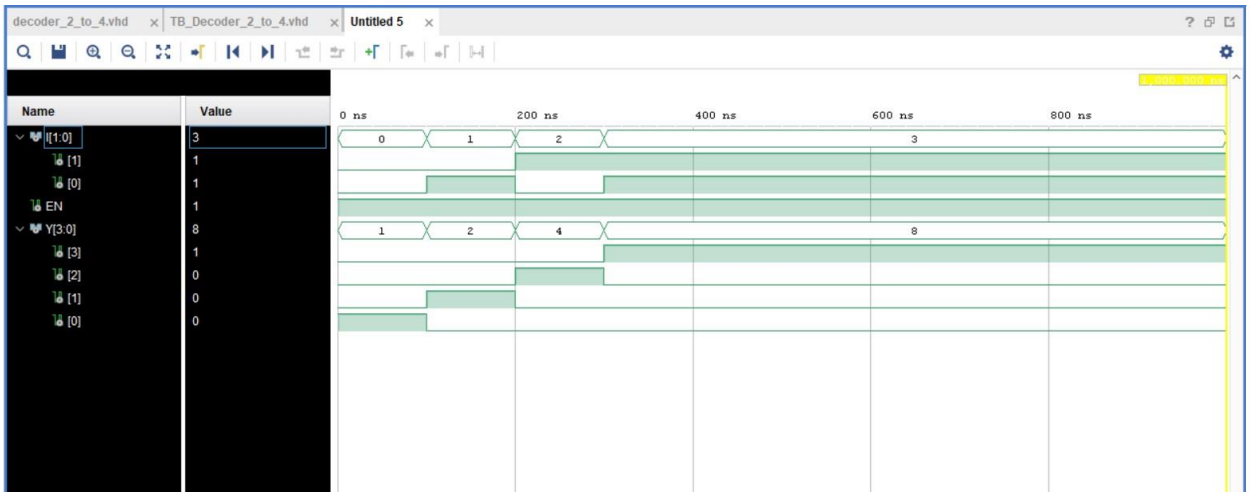


Behavioral Simulation Code for 2 to 4 Decoder (TB_Decoder_2_to_4)

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

entity TB_Decoder_2_to_4 is
--  Port ( );
end TB_Decoder_2_to_4;

architecture Behavioral of TB_Decoder_2_to_4 is
component Decoder_2_to_4 is
    Port ( I : in STD_LOGIC_VECTOR (1 downto 0);
           EN : in STD_LOGIC;
           Y : out STD_LOGIC_VECTOR (3 downto 0));
end component;

--input bus
signal I : STD_LOGIC_VECTOR(1 downto 0);
--enable pin
signal EN : STD_LOGIC;
--output bus
signal Y : STD_LOGIC_VECTOR(3 downto 0);

begin
UUT : Decoder_2_to_4 PORT MAP(
    I => I,
    EN => EN,
    Y => Y );
```

```
process
    begin
        EN <= '1';   --Initial values
        I(0) <= '0';
        I(1) <= '0';

        WAIT FOR 100 ns;
        I(0) <= '1';
        I(1) <= '0';

        WAIT FOR 100 ns;
        I(0) <= '0';
        I(1) <= '1';

        WAIT FOR 100 ns;
        I(0) <= '1';
        I(1) <= '1';

        wait; -- will wait forever
    end process;
end Behavioral;
```

Timing Diagram for 2 to 4 Decoder



03. 3 to 8 Decoder

Truth table for 3 to 8 Decoder

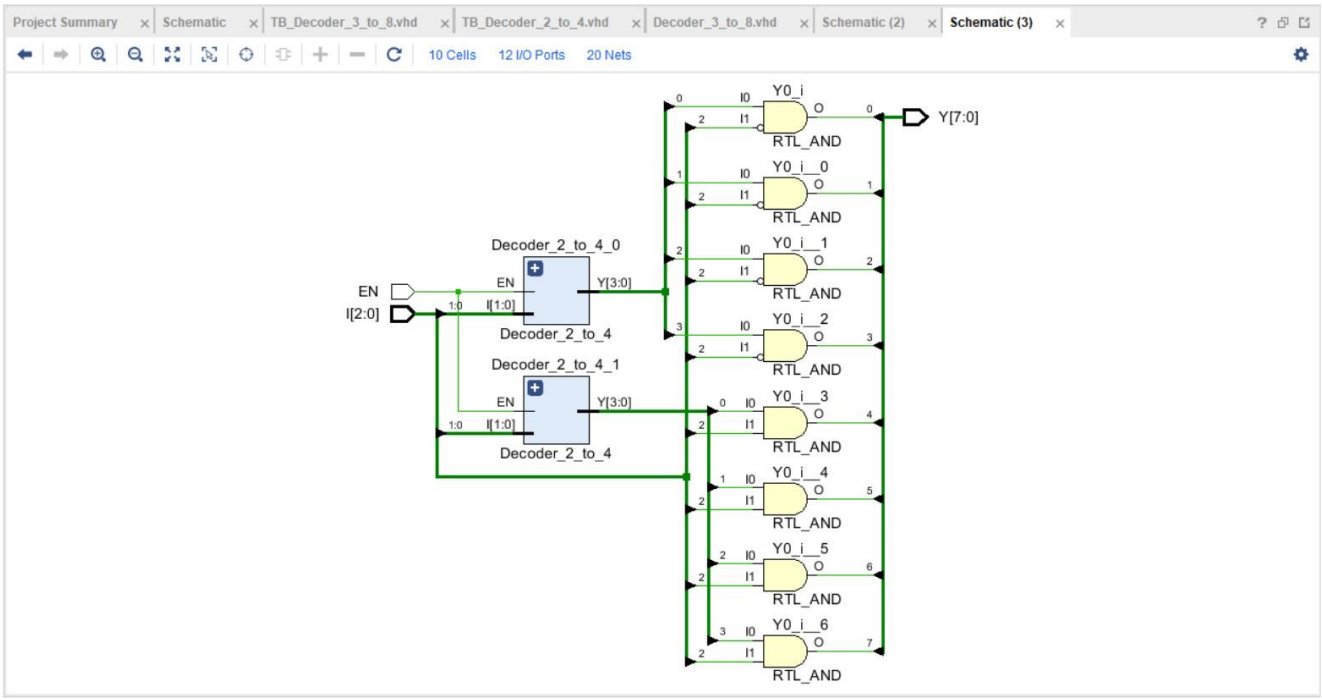
A	B	C	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

VHDL code for 3 to 8 Decoder (Decoder_3_to_8)

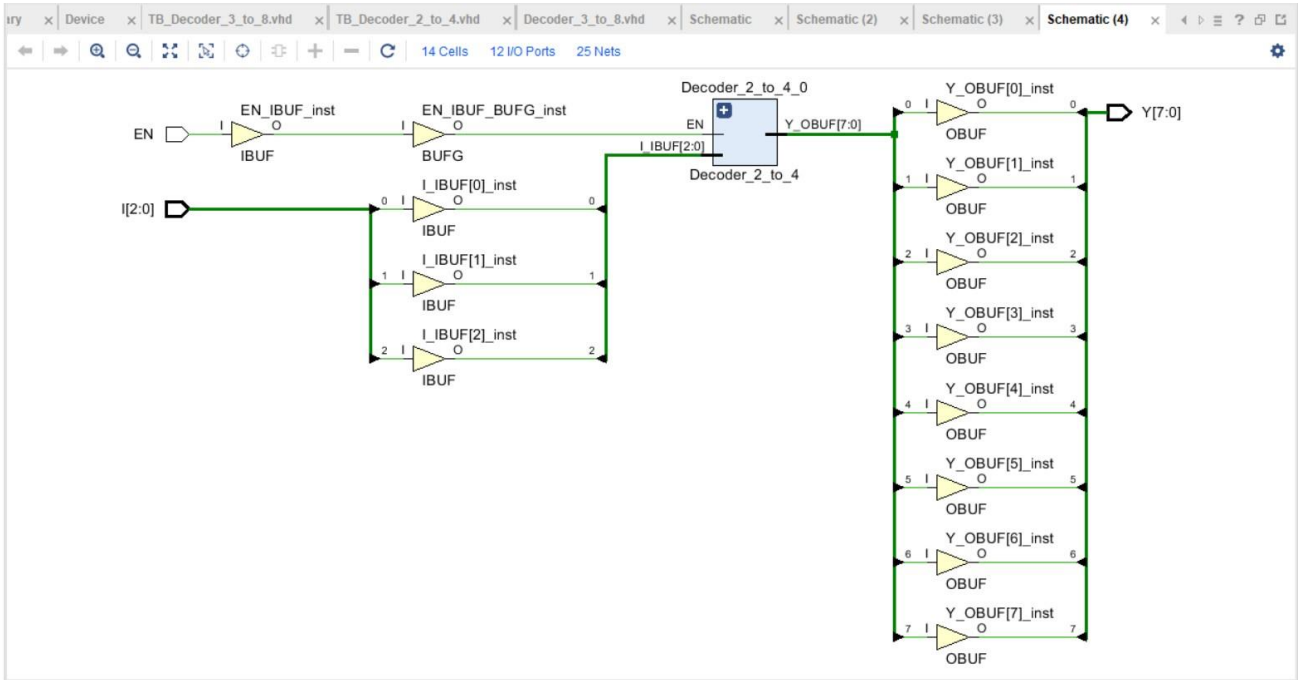
```
----------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Decoder_3_to_8 is
    Port ( I : in STD_LOGIC_VECTOR (2 downto 0);
           EN : in STD_LOGIC;
           Y : out STD_LOGIC_VECTOR (7 downto 0));
end Decoder_3_to_8;

architecture Behavioral of Decoder_3_to_8 is
component Decoder_2_to_4
    port (
    I: in std_logic_vector(1 downto 0);
    EN: in std_logic;
    Y: out std_logic_vector(3 downto 0));
```

```
end component;

SIGNAL Y0, Y1 : std_logic_vector (3 downto 0);

begin
    Decoder_2_to_4_0 : decoder_2_to_4
        port map (
        I(1 downto 0) => I(1 downto 0),
        EN => EN,
        Y(3 downto 0) => Y0(3 downto 0));
    Decoder_2_to_4_1 : decoder_2_to_4
        port map (
        I(1 downto 0) => I(1 downto 0),
        EN => EN,
        Y(3 downto 0) => Y1(3 downto 0));

    Y(0) <= Y0(0) AND (NOT I(2));
    Y(1) <= Y0(1) AND (NOT I(2));
    Y(2) <= Y0(2) AND (NOT I(2));
    Y(3) <= Y0(3) AND (NOT I(2));

    Y(4) <= Y1(0) AND I(2);
    Y(5) <= Y1(1) AND I(2);
    Y(6) <= Y1(2) AND I(2);
    Y(7) <= Y1(3) AND I(2);

end Behavioral;
```

Elaborated Design View for 3 to 8 Decoder (RTL Schematic view)



Schematic Representation of the 3 to 8 Decoder

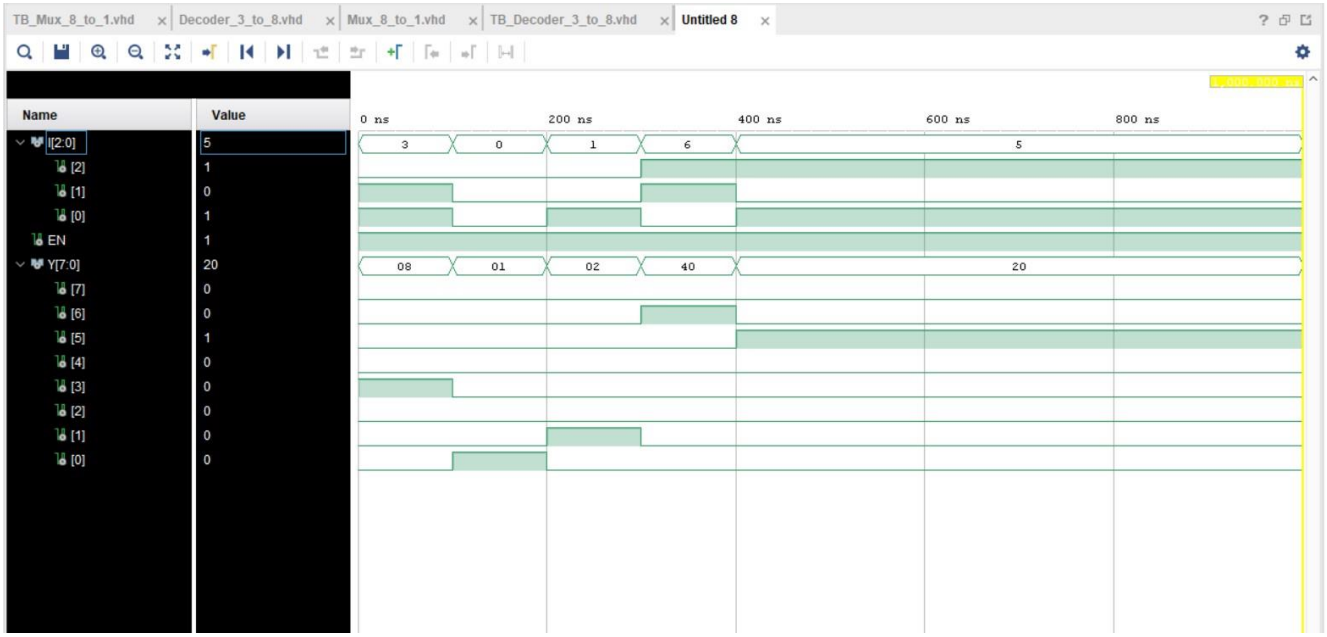


Behavioral Simulation Code for 3 to 8 Decoder (TB_Decoder_3_to_8)

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

entity TB_Decoder_3_to_8 is
--  Port ( );
end TB_Decoder_3_to_8;

architecture Behavioral of TB_Decoder_3_to_8 is
component Decoder_3_to_8 is
    Port ( I : in STD_LOGIC_VECTOR (2 downto 0);
           EN : in STD_LOGIC;
           Y : out STD_LOGIC_VECTOR (7 downto 0));
end component;

--input bus
signal I : STD_LOGIC_VECTOR(2 downto 0);
--enable pin
signal EN : STD_LOGIC;
--output bus
signal Y : STD_LOGIC_VECTOR(7 downto 0);

begin

UUT : Decoder_3_to_8 PORT MAP(
    I => I,
    EN => EN,
    Y => Y );
```

```
process
    begin
                        -- 190019 = 101 110 011 001 000 011
        EN <= '1';
        I <= "011";

        WAIT FOR 100 ns;
        I <= "000";

        WAIT FOR 100 ns;
        I <= "001";

        WAIT FOR 100 ns;
        I <= "110";

        WAIT FOR 100 ns;
        I <= "101";

        wait; -- will wait forever
    end process;
end Behavioral;
```

Timing Diagram for 3 to 8 Decoder



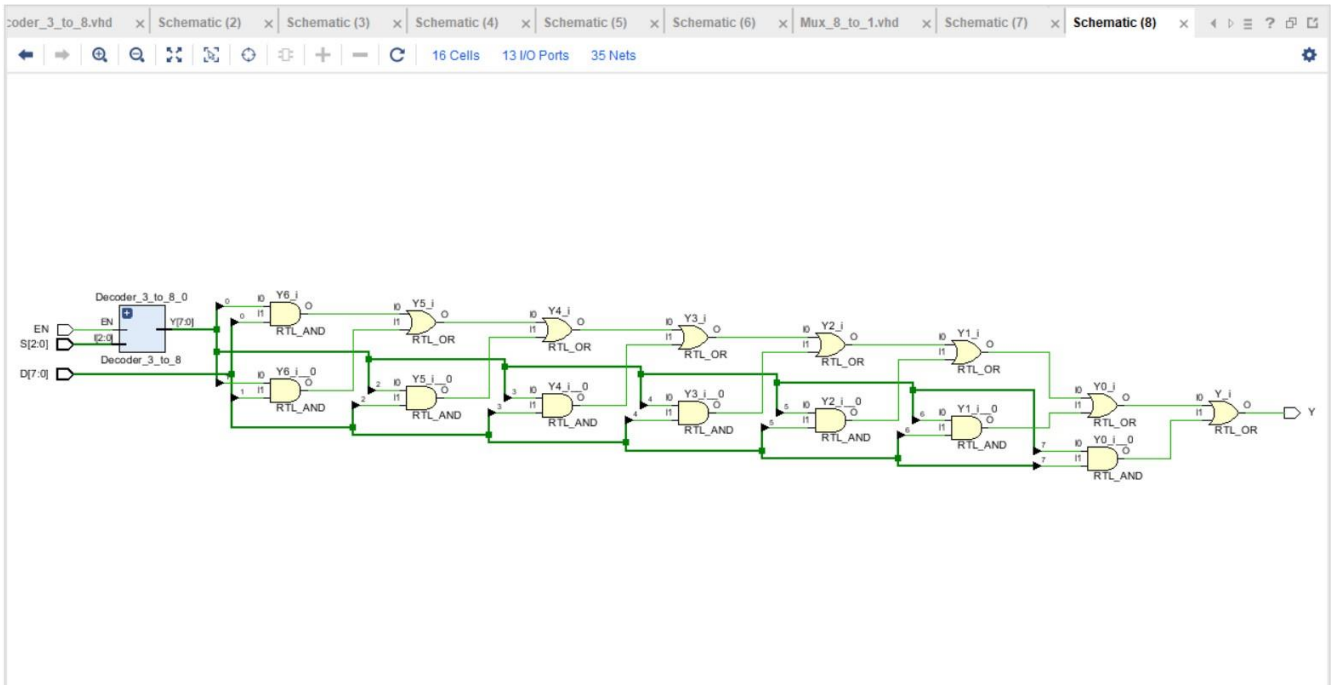
04. 8 to 1 Multiplexer

VHDL code for 8 to 1 Multiplexer (Mux_8_to_1)

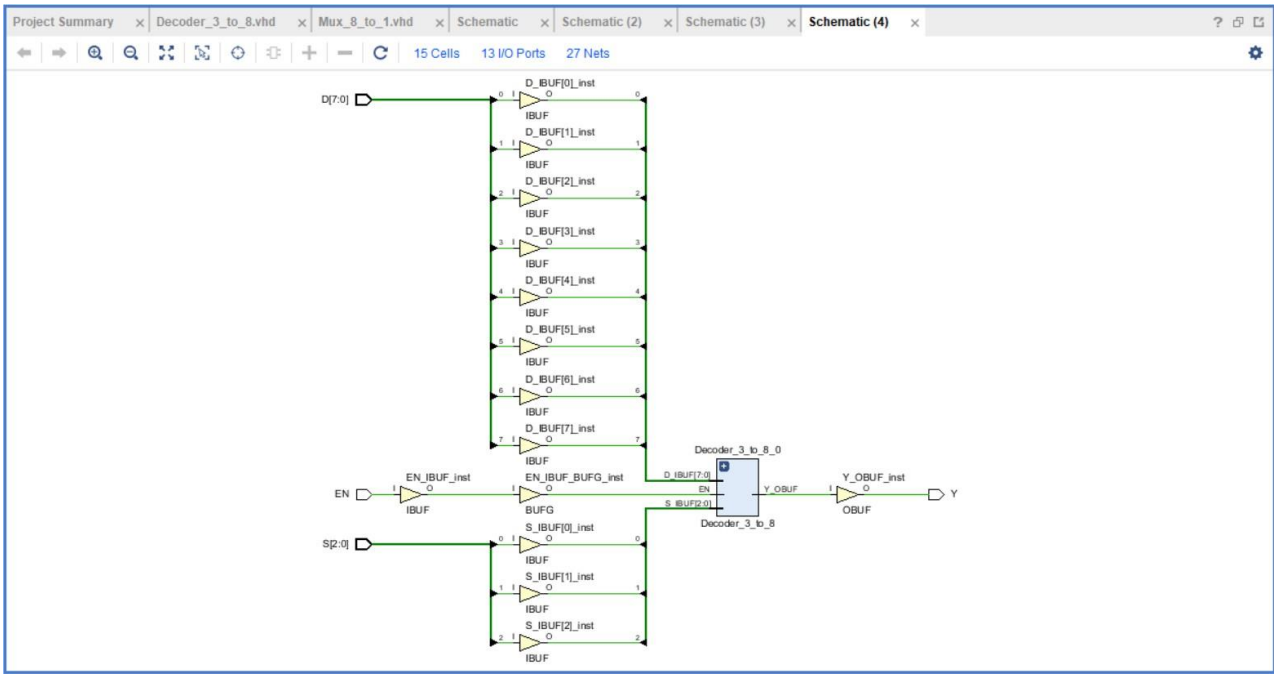
```
----------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Mux_8_to_1 is
    Port ( S : in STD_LOGIC_VECTOR (2 downto 0);
           D : in STD_LOGIC_VECTOR (7 downto 0);
           EN : in STD_LOGIC;
           Y : out STD_LOGIC);
end Mux_8_to_1;
```

```
architecture Behavioral of Mux_8_to_1 is
component Decoder_3_to_8
    port (
    I: in std_logic_vector(2 downto 0);
    EN: in std_logic;
    Y: out std_logic_vector(7 downto 0));
end component;

SIGNAL Y0 : std_logic_vector (7 downto 0);

begin
    Decoder_3_to_8_0 : decoder_3_to_8
        port map (
        I(2 downto 0) => S(2 downto 0),
        EN => EN,
        Y(7 downto 0) => Y0(7 downto 0));

    Y <= (Y0(0) and D(0)) or (Y0(1) and D(1)) or (Y0(2) and D(2)) or
(Y0(3) and D(3)) or (Y0(4) and D(4)) or (Y0(5) and D(5)) or (Y0(6) and
D(6)) or (Y0(7) and D(7));

end Behavioral;
```

Elaborated Design View for 8 to 1 Multiplexer (RTL Schematic view)



Schematic Representation of the 8 to 1 Multiplexer

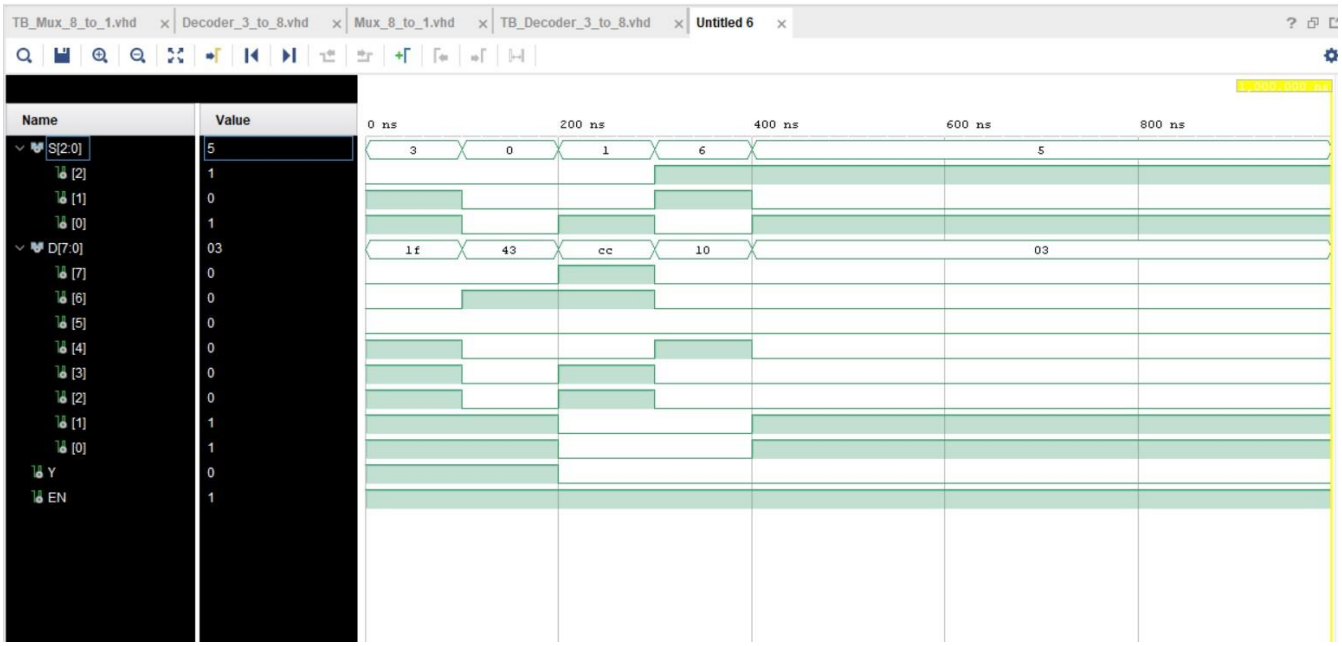


Behavioral Simulation Code for 8 to 1 Multiplexer (TB_Mux_8_to_1)

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

entity TB_Mux_8_to_1 is
--  Port ( );
end TB_Mux_8_to_1;

architecture Behavioral of TB_Mux_8_to_1 is
component Mux_8_to_1
        port(S : in std_logic_vector(2 downto 0);
             D : in std_logic_vector(7 downto 0);
             Y : out std_logic;
             EN: in std_logic);
end component;

signal S : std_logic_vector(2 downto 0);
signal D : std_logic_vector(7 downto 0);
signal Y : std_logic;
signal EN: std_logic;

begin
uut: Mux_8_to_1 port map(
 D => D,
 S => S,
 Y => Y,
 EN => EN );
```

```
process

begin
    EN <= '1';                  -- 190019 = 101 110 011 001 000 011

    D <= "00011111";
    S <= "011";

    wait for 100 ns;
    D <= "01000011";
    S <= "000";

    wait for 100 ns;
    D <= "11001100";
    S <= "001";

    wait for 100 ns;
    D <= "00010000";
    S <= "110";

    wait for 100 ns;
    D <= "00000011";
    S <= "101";

    wait;

end process;
end Behavioral;
```

Timing Diagram for 8 to 1 Multiplexer



05. XDC (Xilinx Design Constraints) VHDL Code

```
#inputs
set_property PACKAGE_PIN V17 [get_ports {D[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[0]}]
set_property PACKAGE_PIN V16 [get_ports {D[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[1]}]
set_property PACKAGE_PIN W16 [get_ports {D[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[2]}]
set_property PACKAGE_PIN W17 [get_ports {D[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[3]}]
set_property PACKAGE_PIN W15 [get_ports {D[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[4]}]
set_property PACKAGE_PIN V15 [get_ports {D[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[5]}]
set_property PACKAGE_PIN W14 [get_ports {D[6]}]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports {D[6]}]
set_property PACKAGE_PIN W13 [get_ports {D[7]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[7]}]

set_property PACKAGE_PIN U1 [get_ports {S[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {S[0]}]
set_property PACKAGE_PIN T1 [get_ports {S[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {S[1]}]
set_property PACKAGE_PIN R2 [get_ports {S[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {S[2]}]

#output
set_property PACKAGE_PIN U16 [get_ports {Y}]
set_property IOSTANDARD LVCMOS33 [get_ports {Y}]

#push button
set_property PACKAGE_PIN U18 [get_ports EN]
set_property IOSTANDARD LVCMOS33 [get_ports EN]
```

06. Conclusions

At the end of the lab, I could design decoders and multiplexers and obtained strong knowledge of developing components from very basic ones. And I obtained a sound knowledge of how decodes and multiplexers function.