Lab 2. ModelSim VHDL 1of 4 Decoder

1 of 4 Decoder with active low outputs. See the truth table below, but don't create the gates directly, instead **use two (nested) IF statements** to design the decoder. Decoder enable (i_EN) is active low. If i_EN is high, then all 4 outputs are high, i.e., turned off, because the Y outputs are active low). You should use std_logic for the i_EN pin and for select pins i_B and i_A (pin i_B is the MSB). Use a 2-bit std_logic_vector for internal signal w_BA consisting of i_B & i_A (concatenated). Use a 4 bit std_logic_vector o_Y for your output pins. Write a testbench that tests all 4 combinations of the w_BA select lines with i_EN low (enabled), then test all 4 combinations of the select lines with i_EN high (disabled).

DECODERS

Another common MSI circuit is the *decoder* (also called a *demultiplexer*). A decoder is an arrangement of gates that allows a single enable or data input to be sent to several destinations. A decoder circuit along with its MSI symbol is shown in Fig. 3-18. This is a two-line to four-line decoder (also called a one out of four). Two select inputs A and B determine which of four outputs Y_0 through Y_3 will be connected to the enable or data input.

Enable	B	A	Y_3	Y_2	Y_1	Y_0
1	X	X	1	1	1	1
0	0	0	1	1	1	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	1

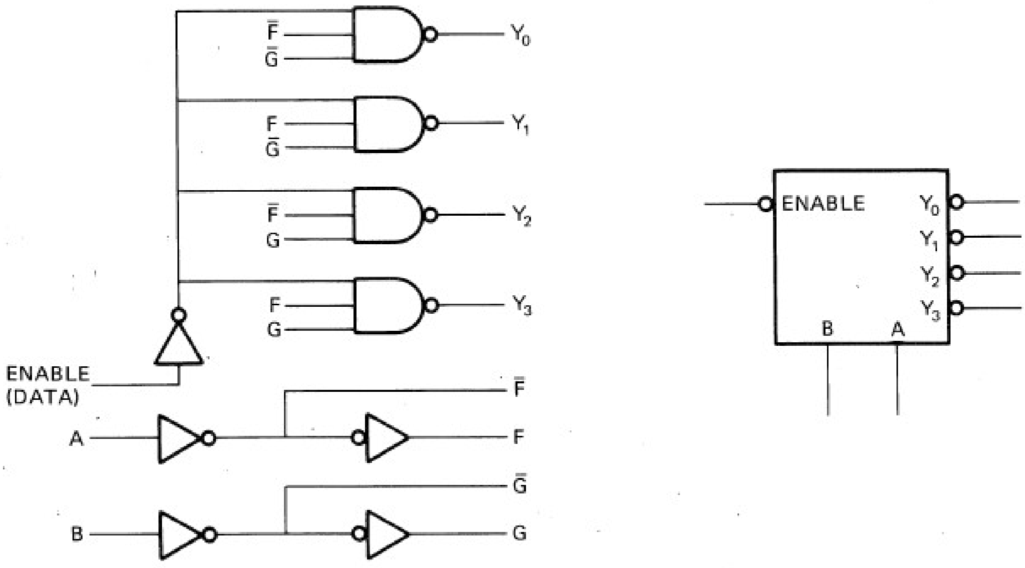

Figure 3-18 Decoder.

- For credit for this assignment, use the NANDland naming style and the coding style in my powerpoints: Submit:
  - Your decoder.vhd and decoder_tb files.
  - The ModelSim waveform from the testbench.
    (You can save an image of the waveform using Windows to SNIP a screenshot of the image).

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

entity decoder is
    port (
        i_EN : in std_logic;
            i_A  : in std_logic;
            i_B  : in std_logic;

        o_Y  : out std_logic_vector(3 downto 0)
    );
end decoder;

architecture Behavioral of decoder is
Signal w_BA : std_logic_vector(1 downto 0);
begin
w_BA <= i_B & i_A;
process(w_BA, i_EN)
begin
            if i_EN = '1' then o_Y <= "1111";
            else
                        if w_BA = "00" then o_Y <= "1110";
                        elsif w_BA = "01" then o_Y <= "1101";
                        elsif w_BA = "10" then o_Y <= "1011";
                        elsif  w_BA = "11" then o_Y <= "0111";
            end if;
end if;
end process;
end Behavioral;
```

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

entity tb_decoder is
end entity tb_decoder;

architecture behaviour of tb_decoder is

  -- Use Unit Under Test (UUT)
  component decoder
    port (
      i_EN : in  std_logic;
      i_A  : in  std_logic;
      i_B  : in  std_logic;
      o_Y  : out std_logic_vector(3 downto 0)
    );
  end component;

  -- Signals for input and output
  signal test_i_EN : std_logic := '0';
  signal test_i_A  : std_logic := '0';
  signal test_i_B  : std_logic := '0';
  signal test_o_Y  : std_logic_vector(3 downto 0);

begin

  -- Instantiate the decoder
  uut: decoder
    port map (
      i_EN => test_i_EN,
      i_A  => test_i_A,
      i_B  => test_i_B,
      o_Y  => test_o_Y
    );

  -- Stimulus process
  process
  begin
    -- Test with Enable = 1 (Output should be "1111" regardless of inputs)
    test_i_EN <= '1';
    test_i_A  <= '0';
    test_i_B  <= '0';
    wait for 50 ns;

    test_i_A  <= '1';
    wait for 50 ns;
    test_i_B  <= '1';
    wait for 50 ns;

    test_i_A  <= '0';
    wait for 50 ns;

    -- Test with Enable = 0 (Decoder active)
    test_i_EN <= '0';

    -- Apply all 4 combinations of i_A and i_B
    test_i_A <= '0';
    test_i_B <= '0';
    wait for 50 ns;

    test_i_A <= '0';
    test_i_B <= '1';
    wait for 50 ns;

    test_i_A <= '1';
    test_i_B <= '0';
    wait for 50 ns;

    test_i_A <= '1';
    test_i_B <= '1';
    wait for 50 ns;

    -- End simulation
    assert false
    report "End of simulation."
    severity failure;
  end process;
end behaviour;
```