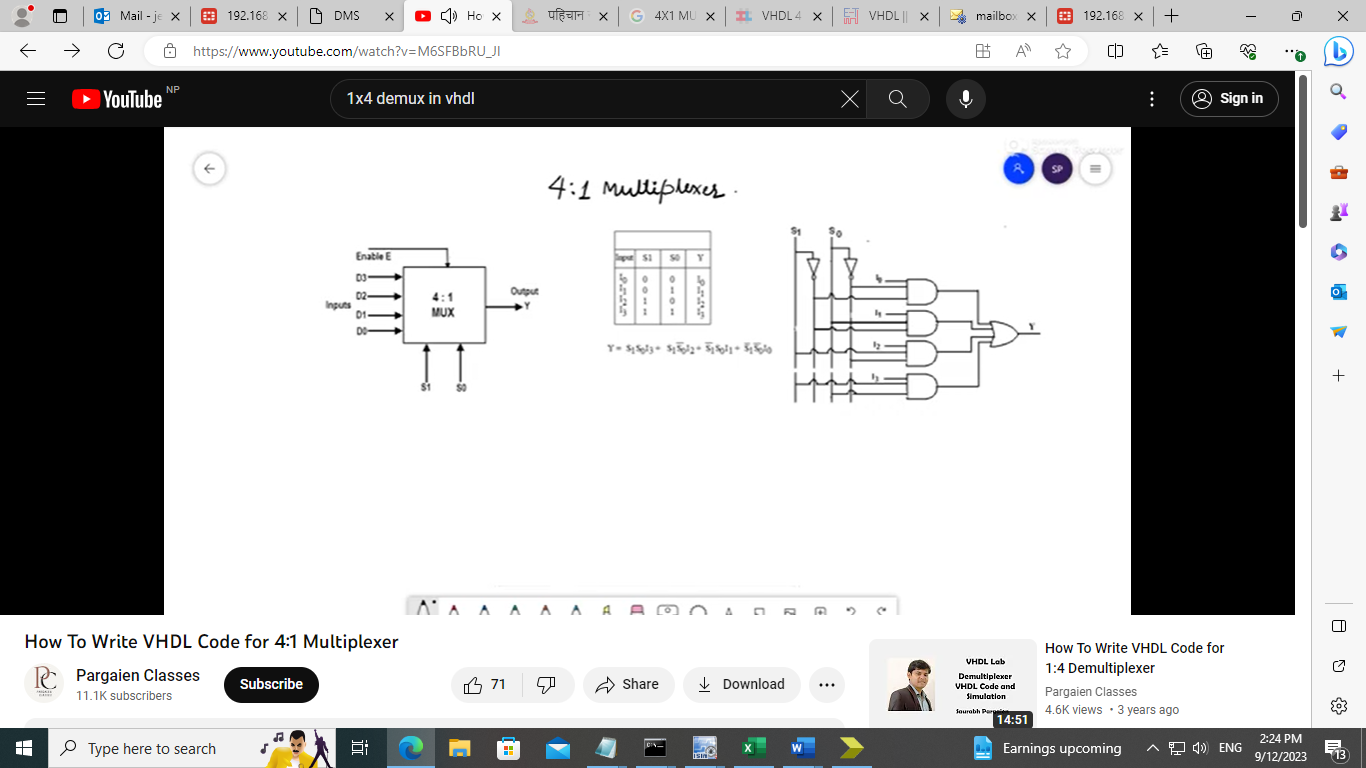
**Mux, Demux, Encoder and Decoder Lab**

1. **4x1 mux**



library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity mux\_four is

**Port ( i : in STD\_LOGIC\_VECTOR (3 downto 0);**

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

**y : out STD\_LOGIC);**

end mux\_four;

architecture Behavioral of mux\_four is

begin

**with s select**

**y <= i(0) when "00",**

**i(1) when "01",**

**i(2) when "10",**

**i(3) when others;**

end Behavioral;

**Test bench**

-- Stimulus process

stim\_proc: process

begin

**i <= "1010";**

**s <= "00"; wait for 100 ns;**

**s <= "01"; wait for 100 ns;**

**s <= "10"; wait for 100 ns;**

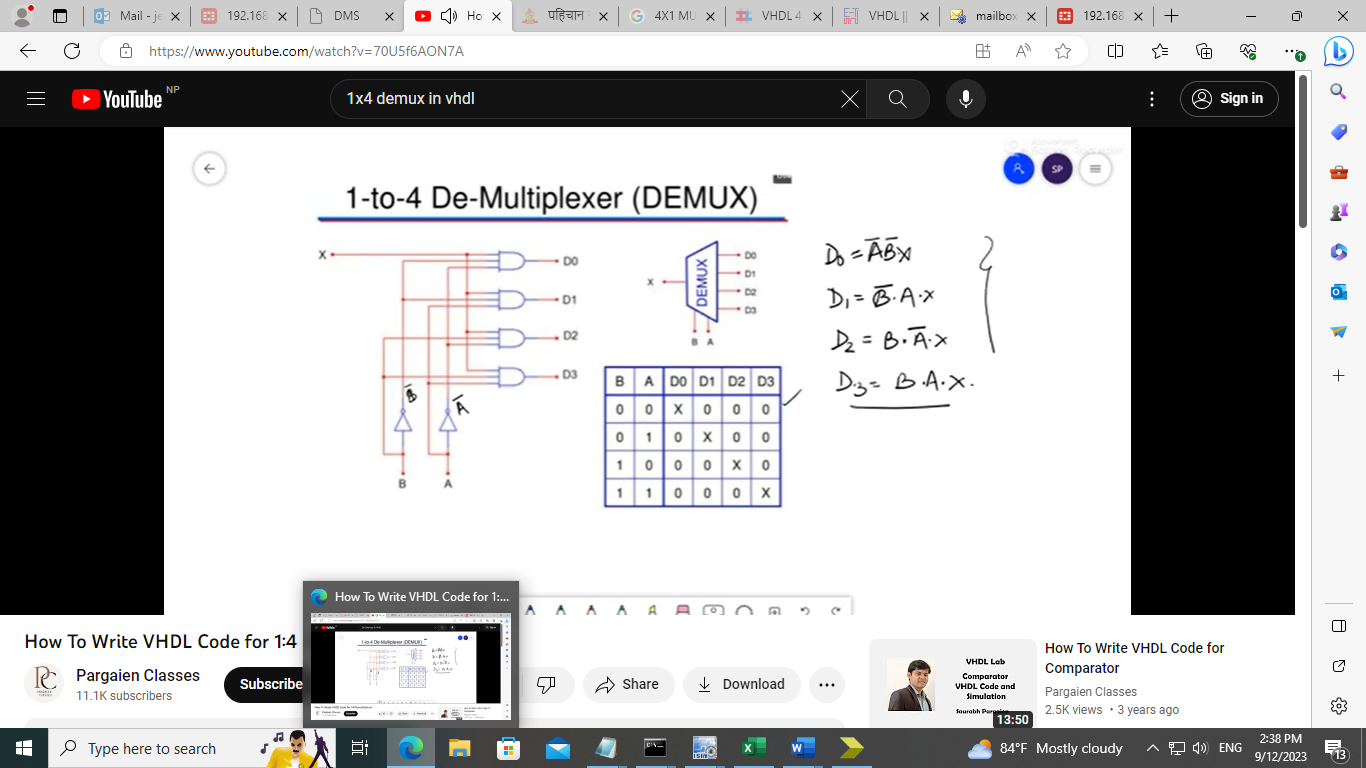
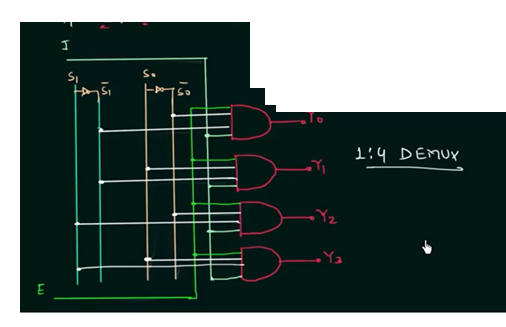
**s <= "11"; wait for 100 ns;**

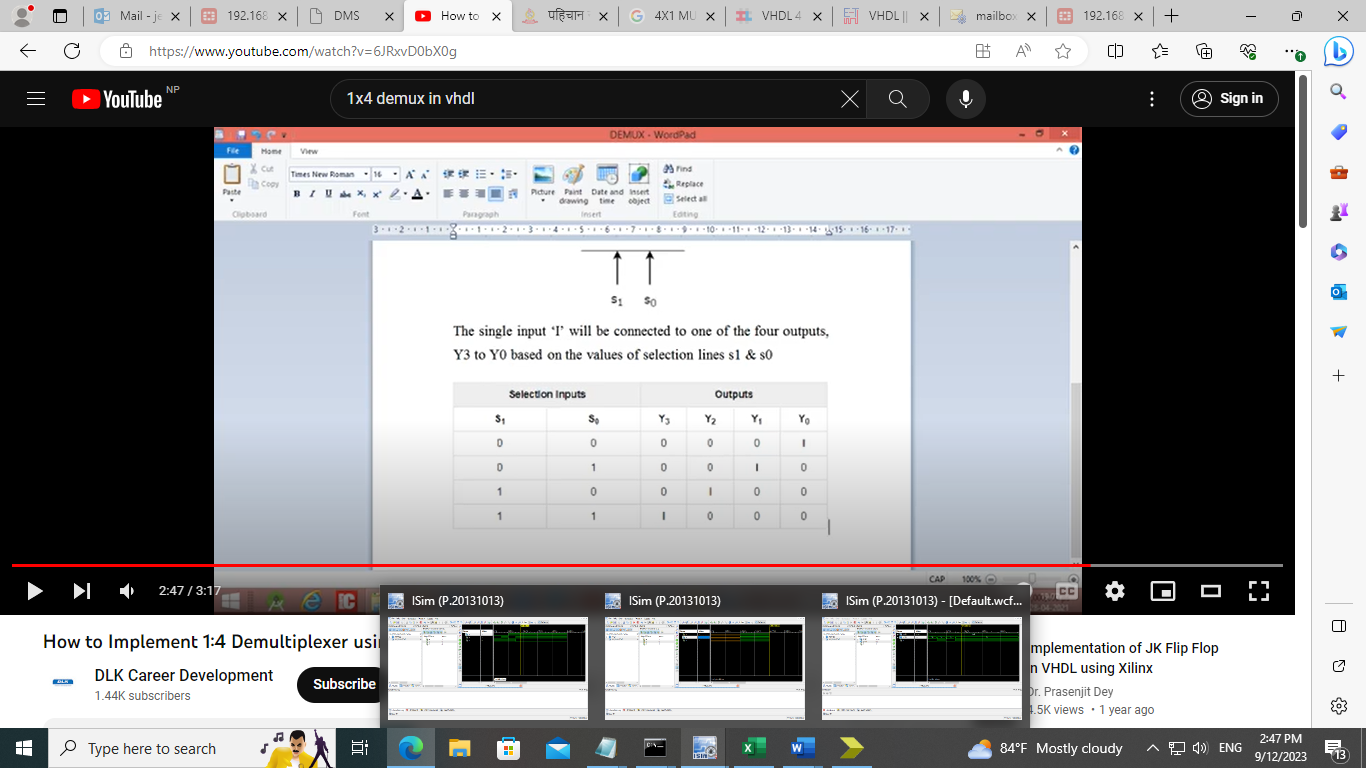
wait;

end process;

END;

1. **1x4 Demux**





entity demux\_lab is

Port ( I : in STD\_LOGIC;

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

O : out STD\_LOGIC\_VECTOR (3 downto 0));

end demux\_lab;

architecture Behavioral of demux\_lab is

begin

**O(0) <= (I AND NOT S(0) AND NOT S(1));**

**O(1) <= (I AND NOT S(0) AND S(1));**

**O(2) <= (I AND S(0) AND NOT S(1));**

**O(3) <= (I AND S(0) AND S(1));**

end Behavioral;

**Test bench for 1x4 Demux**

I <= '1';

S(0) <= '0';

S(1) <= '0';

wait for 100 ns;

S(0) <= '0';

S(1) <= '1';

wait for 100 ns;

S(0) <= '1';

S(1) <= '0';

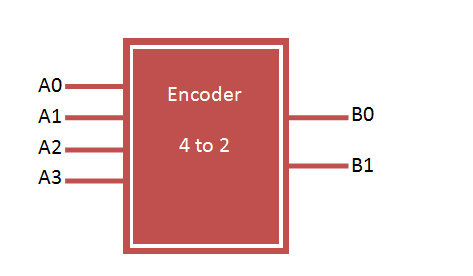
wait for 100 ns;

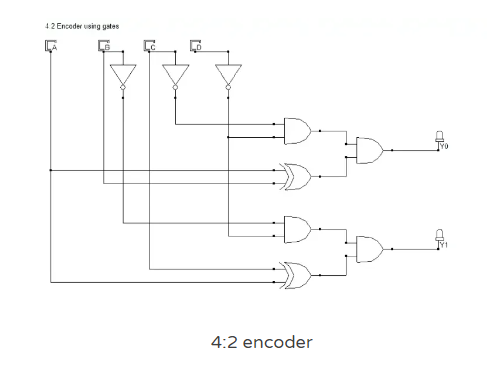
S(0) <= '1';

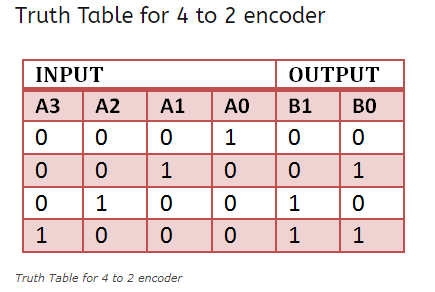
S(1) <= '1';

wait for 100 ns;

1. **VHDL Code for 4 to 2 Encoder**







entity encoder\_4to2 is

Port ( a : in STD\_LOGIC\_VECTOR (3 downto 0);

b : out STD\_LOGIC\_VECTOR (1 downto 0));

end encoder\_4to2;

architecture Behavioral of encoder\_4to2 is

begin

**process(a)**

**begin**

**case a is**

**when "0001" => b <= "00";**

**when "0010" => b <= "01";**

**when "0100" => b <= "10";**

**when "1000" => b <= "11";**

**when others => b <= "ZZ";**

**end case;**

**end process;**

end Behavioral;

**Test bench 4 to 2 Encoder**

stim\_proc: process

begin

**a <= "0000";**

**wait for 100 ns;**

**a <= "0001";**

**wait for 100 ns;**

**a <= "0010";**

**wait for 100 ns;**

**a <= "0100";**

**wait for 100 ns;**

**a <= "1000";**

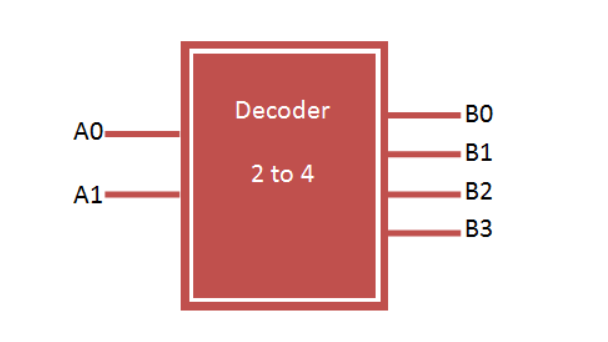
**wait for 100 ns;**

wait;

end process;

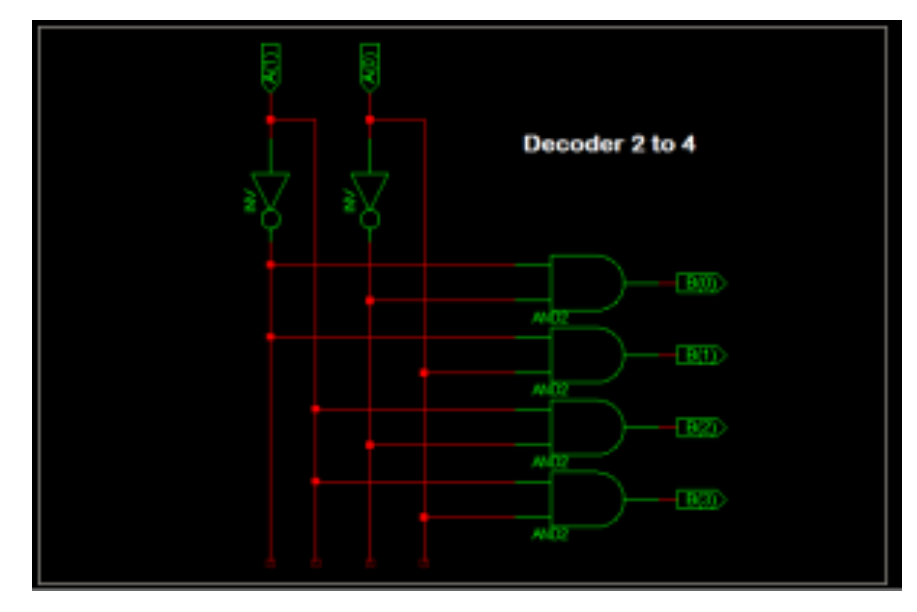
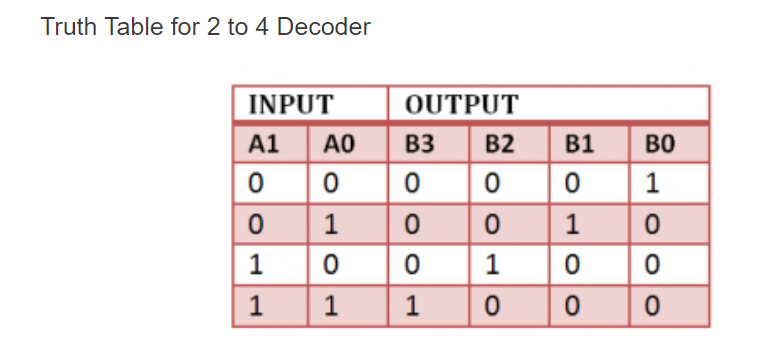
END;

# VHDL Code for 2 to 4 decoder



2 to 4 decoder has 2 input lines and 4 output lines. Decoder can be easily constructed using basic logic gates.

### 2 to 4 Decoder design using logic gates

entity decoder\_2to4 is

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

b : out STD\_LOGIC\_VECTOR (3 downto 0));

end decoder\_2to4;

architecture Behavioral of decoder\_2to4 is

begin

**process(a)**

**begin**

**case a is**

**when "00" => b <= "0001";**

**when "01" => b <= "0010";**

**when "10" => b <= "0100";**

**when "11" => b <= "1000";**

**when others => b <= "ZZZZ";**

**end case;**

**end process;**

end Behavioral;

**Test bench** for 2 to 4 decoder

-- Stimulus process

stim\_proc: process

begin

**a <= "00"; wait for 100 ns;**

**a <= "01"; wait for 100 ns;**

**a <= "10"; wait for 100 ns;**

**a <= "11"; wait for 100 ns;**

wait;

end process;

END;

