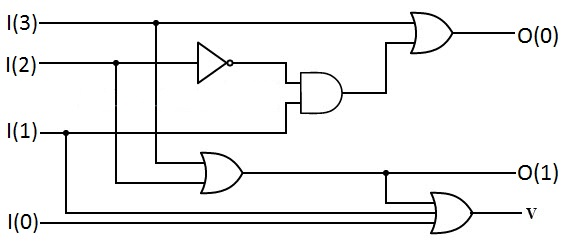
**EXPERIMENT 5**

**AIM: To design and simulate priority Encoder and Decoder using VHDL and ModelSim.**

**Theory:** A **priority encoder** is a circuit or algorithm that compresses multiple binary inputs into a smaller number of outputs. The output of a priority encoder is the binary representation of the original number starting from zero of the most significant input bit. They are often used to control interrupt requests by acting on the highest priority encoder.

**PRIORITY ENCODER**

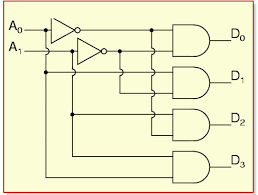
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **I(3)** | **I(2)** | **I(1)** | **I(0)** | **O(1)** | **O(0)** | **V** |
| **0** | **0** | **0** | **0** | **X** | **X** | **0** |
| **0** | **0** | **0** | **1** | **0** | **0** | **1** |
| **0** | **0** | **1** | **X** | **0** | **1** | **1** |
| **0** | **1** | **X** | **X** | **1** | **0** | **1** |
| **1** | **X** | **X** | **X** | **1** | **1** | **1** |

**TRUTH TABLE**

**DECODER:** In digital electronics, a **decoder** is a combinational logic circuit that converts binary information from the n coded inputs to a maximum of 2n unique outputs.

There are several types of binary decoders, but in all cases a decoder is an electronic circuit with multiple input and multiple output signals, which converts every unique combination of input states to a specific combination of output states. In addition to integer data inputs, some decoders also have one or more "enable" inputs. When the enable input is negated (disabled), all decoder outputs are forced to their inactive states.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A0** | **A1** | **D0** | **D1** | **D2** | **D3** |
| **0** | **0** | **1** | **0** | **0** | **0** |
| **0** | **1** | **0** | **1** | **0** | **0** |
| **1** | **0** | **0** | **0** | **1** | **0** |
| **1** | **1** | **0** | **0** | **0** | **1** |



**TRUTH TABLE DECODER 2\_4**

**VHDL CODE FOR PRIORITY ENCODER**

LIBRARY ieee;

USE ieee.std\_logic\_1164.all;

USE ieee.std\_logic\_arith.all;

ENTITY PENCODER\_4\_2 IS

port(D:in std\_logic\_vector(3 downto 0);x,y,P:out std\_logic);

END ENTITY PENCODER\_4\_2;

--

ARCHITECTURE PENCODER\_4\_2\_STRUCTURAL OF PENCODER\_4\_2 IS

component AND\_GATE

port(A,B:in std\_logic;Z: out std\_logic);

end component;

component OR\_GATE

port(A,B:in std\_logic;Z: out std\_logic);

end component;

component NOT\_GATE

port(A:in std\_logic;Z: out std\_logic);

end component;

SIGNAL DBAR:std\_logic;

SIGNAL VAR:std\_logic\_vector(1 downto 0);

BEGIN

N1: NOT\_GATE port map (D(2),DBAR);

A1: AND\_GATE port map (D(1),DBAR,VAR(0));

O1: OR\_GATE port map (VAR(0),D(3),Y);

O2: OR\_GATE port map (D(2),D(3),X);

O3: OR\_GATE port map (X,D(1),VAR(1));

O4: OR\_GATE port map (VAR(1),D(0),P);

END ARCHITECTURE PENCODER\_4\_2\_STRUCTURAL;

ARCHITECTURE PENCODER\_4\_2\_DATAFLOW OF PENCODER\_4\_2 IS

BEGIN

x<= D(2) or D(3);

y<= D(3) or (D(1) and (not D(2)));

P<= D(0) or D(1) or D(2) or D(3);

END ARCHITECTURE PENCODER\_4\_2\_DATAFLOW;

**VHDL CODE FOR DECODER**

LIBRARY ieee;

USE ieee.std\_logic\_1164.all;

USE ieee.std\_logic\_arith.all;

ENTITY DECODER\_2\_4 IS

port(A,B:in std\_logic;D: out std\_logic\_vector(3 downto 0));

END ENTITY DECODER\_2\_4;

--

ARCHITECTURE DECODER\_2\_4\_DATAFLOW OF DECODER\_2\_4 IS

BEGIN

D(0)<= (not A) nand (not B) ;

D(1)<= (not A) nand B ;

D(2)<= A nand (not B) ;

D(3)<= A nand B ;

END ARCHITECTURE DECODER\_2\_4\_DATAFLOW;

ARCHITECTURE DECODER\_2\_4\_STRUCTURAL OF DECODER\_2\_4 IS

component NAND\_GATE

port(A,B:in std\_logic;Z: out std\_logic);

end component;

component NOT\_GATE

port(A:in std\_logic;Z: out std\_logic);

end component;

SIGNAL ABAR,BBAR:std\_logic;

BEGIN

N1: NOT\_GATE port map(A,ABAR);

N2: NOT\_GATE port map(B,BBAR);

NA1: NAND\_GATE port map(ABAR,BBAR,D(0));

NA2: NAND\_GATE port map(ABAR,B,D(1));

NA3: NAND\_GATE port map(A,BBAR,D(2));

NA4: NAND\_GATE port map(A,B,D(3));

END ARCHITECTURE DECODER\_2\_4\_STRUCTURAL;

ARCHITECTURE DECODER\_2\_4\_BEHAV OF DECODER\_2\_4 IS

BEGIN

process (A,B) is

begin

if (A ='0' and B = '0') then

D <= "1110";

elsif (A ='1' and B = '0') then

D <= "1011";

elsif (A ='0' and B = '1') then

D <= "1101";

else

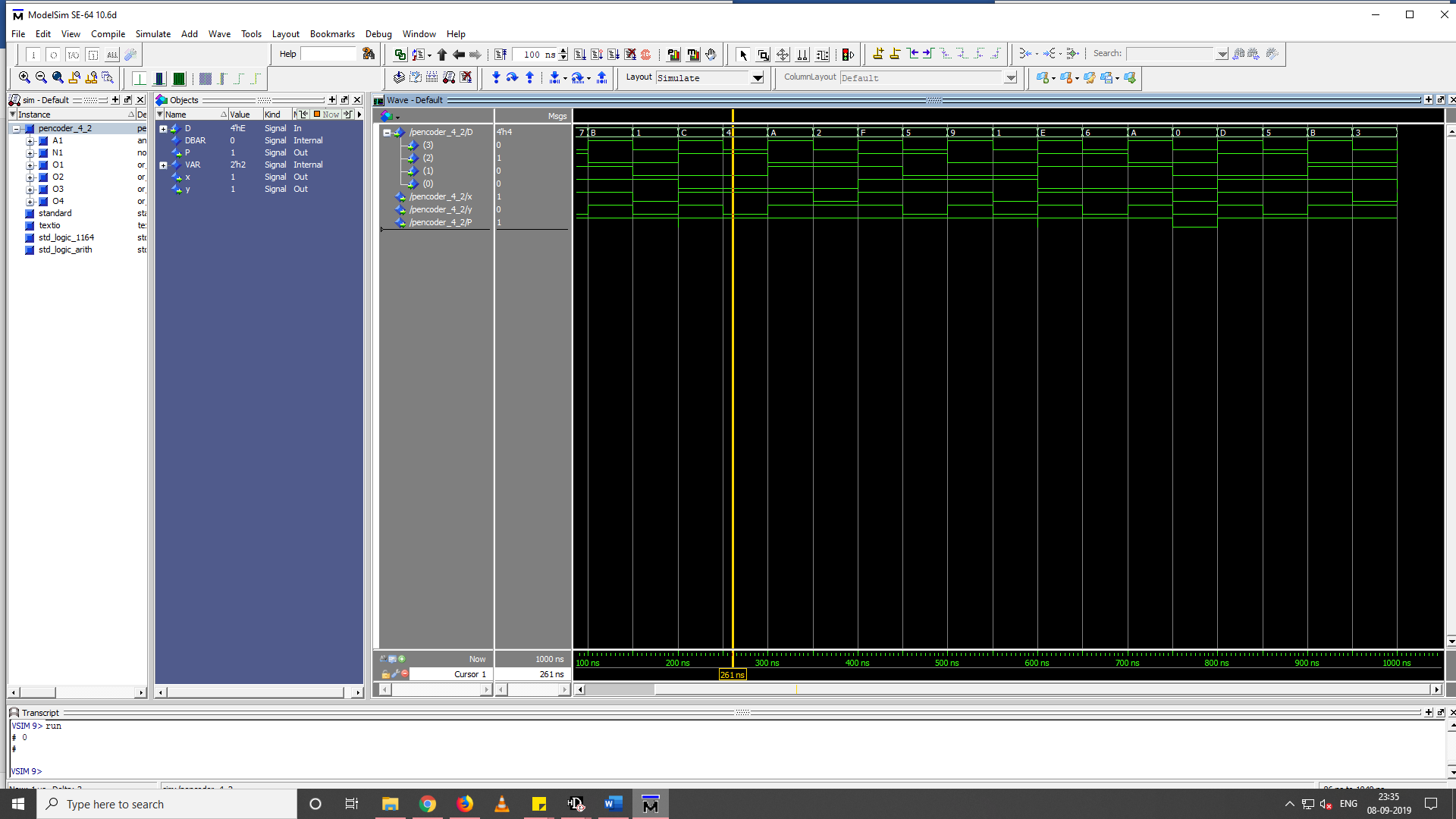
D <= "0111";

end if;

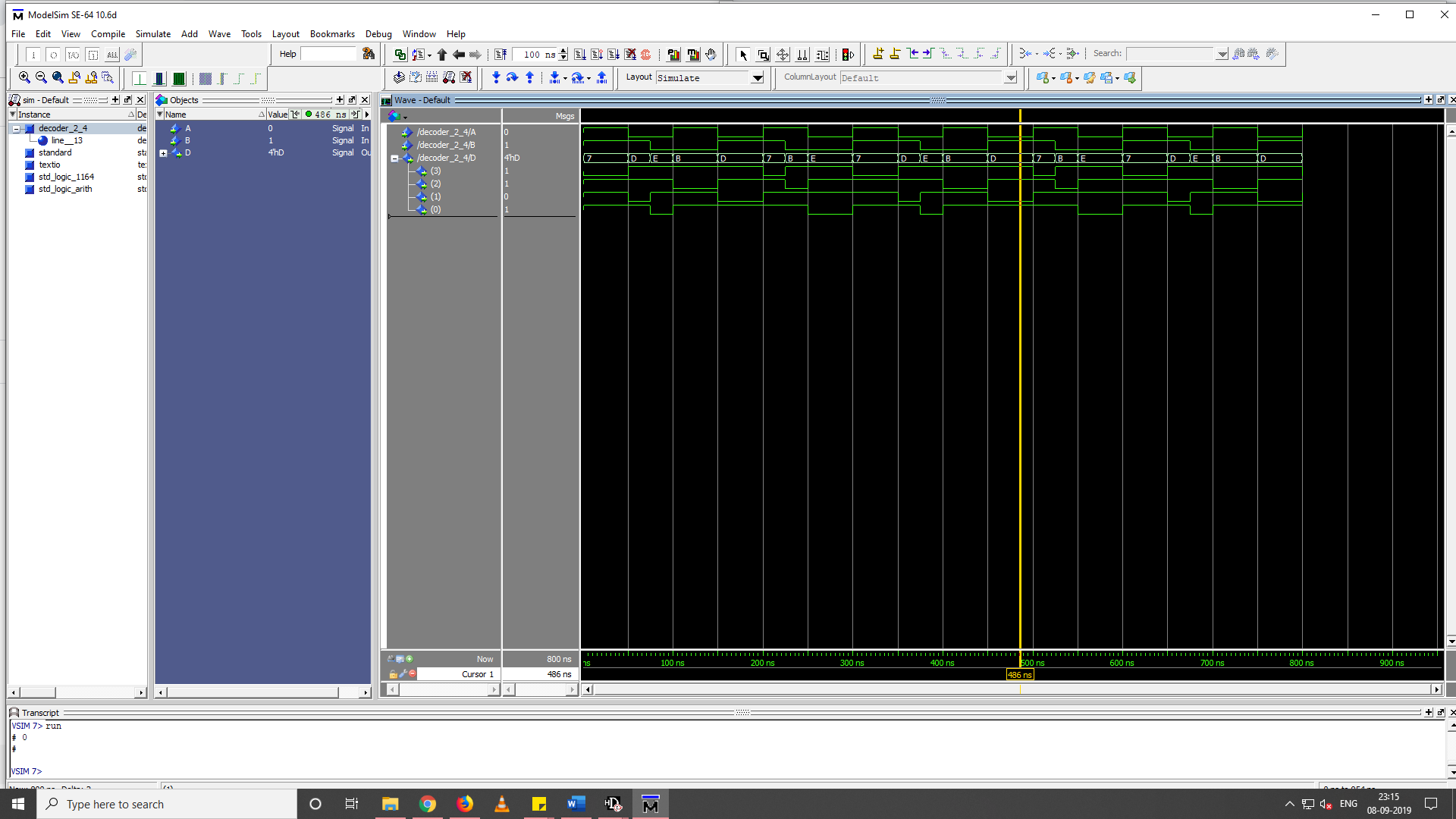
end process;

END ARCHITECTURE DECODER\_2\_4\_BEHAV;

**RESULTS:**



**PRIORITY ENCODER 4\_2**



**DECODER 4\_2**