SUBJECT: CIRCUIT DESIGN USING HDL LAB

TOPIC: 8:3 ENCODER USING STRUCTURAL STYLE OF MODELLING

SUBMITTED BY: ADARSH TIWARI **SUBMITTED TO: DR. VAIBHAV NEEMA SIR**

ROLL NO: 17E7004

ENROLLMENT NO: DE17196

VHDL code for 8:3 encoder using structural style of modeling

```
library ieee;
use ieee.std_logic_1164.all;

entity encoder is                                                  -- Entity declaration for 3:8_encoder
 port (i0, i1, i2, i3, i4, i5, i6, i7: in bit;
  out0, out1, out2: out bit);
end encoder;

entity or4_gate is
Port(
a_or4 : in bit;
b_or4 : in bit;
c_or4 : in bit;
d_or4 : in bit;
e_or4 : out bit);
end or4_gate;

architecture Dataflow of or4_gate is
begin
e_or4 <= a_or4 OR(b_or4 OR c_or4) OR(c_or4 OR d_or4);
end Dataflow;

architecture structure of encoder is                               -- Architecture body for 3:8_encoder

  component or4_gate is                                            -- or component declaration
  port (aor4, bor4, cor4, dor4: in bit;
   eor4: out bit);
 end component;



begin
```

```
u1: entity work .or4_gate(Dataflow) port map (a_or4 => i1, b_or4 => i3, c_or4 => i5, d_or4 => i7, e_or4 => out0);

u2: entity work .or4_gate(Dataflow) port map (a_or4 => i1, b_or4 => i3, c_or4 => i5, d_or4 => i7, e_or4 => out1);

u3: entity work .or4_gate(Dataflow) port map (a_or4 => i1, b_or4 => i3, c_or4 => i5, d_or4 => i7, e_or4 => out2);


end structure;
```

TEST BENCH

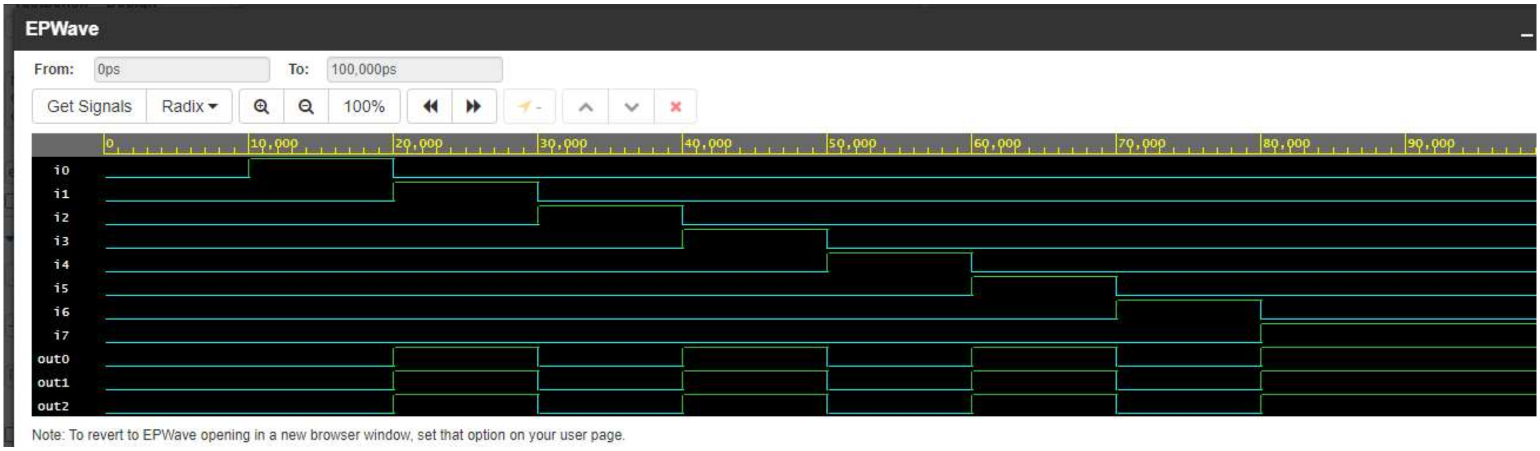
```
library IEEE;

use IEEE.std_logic_1164.all;

entity encoder_tb is

end;

architecture bench of encoder_tb is


component encoder

Port (

i0 : in bit;

i1 : in bit;

i2 : in bit;

i3 : in bit;

i4 : in bit;

i5 : in bit;

i6 : in bit;

i7 : in bit;

out0 : out bit;

out1 : out bit;

out2 : out bit);

end component;


signal i0: bit;

signal i1: bit;

signal i2: bit;

signal i3: bit;

signal i4: bit;

signal i5: bit;

signal i6: bit;
```

```vhdl
signal i7: bit;
signal out0: bit;
signal out1: bit;
signal out2: bit;

begin

uut: encoder port map ( i0 => i0,
i1 => i1,
i2 => i2,
i3 => i3,
i4 => i4,
i5 => i5,
i6 => i6,
i7 => i7,
out0 => out0,
out1 => out1,
out2 => out2 );

stimulus: process
begin

-- Initialisation code
i0 <= '0';
i1 <= '0';
i2 <= '0';
i3 <= '0';
i4 <= '0';
i5 <= '0';
i6 <= '0';
i7 <= '0';
```

```
-- test bench stimulus code

wait for 10ns;

i0 <= '1';

i1 <= '0';

i2 <= '0';

i3 <= '0';

i4 <= '0';

i5 <= '0';

i6 <= '0';

i7 <= '0';

wait for 10ns;

i0 <= '0';

i1 <= '1';

i2 <= '0';

i3 <= '0';

i4 <= '0';

i5 <= '0';

i6 <= '0';

i7 <= '0';

wait for 10ns;

i0 <= '0';

i1 <= '0';

i2 <= '1';

i3 <= '0';

i4 <= '0';

i5 <= '0';

i6 <= '0';

i7 <= '0';

wait for 10ns;

i0 <= '0';
```

```
i1 <= '0';
i2 <= '0';
i3 <= '1';
i4 <= '0';
i5 <= '0';
i6 <= '0';
i7 <= '0';
wait for 10ns;
i0 <= '0';
i1 <= '0';
i2 <= '0';
i3 <= '0';
i4 <= '1';
i5 <= '0';
i6 <= '0';
i7 <= '0';
wait for 10ns;
i0 <= '0';
i1 <= '0';
i2 <= '0';
i3 <= '0';
i4 <= '0';
i5 <= '1';
i6 <= '0';
i7 <= '0';
wait for 10ns;
i0 <= '0';
i1 <= '0';
i2 <= '0';
i3 <= '0';
i4 <= '0';
```

```
i5 <= '0';
i6 <= '1';
i7 <= '0';
wait for 10ns;
i0 <= '0';
i1 <= '0';
i2 <= '0';
i3 <= '0';
i4 <= '0';
i5 <= '0';
i6 <= '0';
i7 <= '1';
wait for 10ns;

wait;
end process;
end;
```

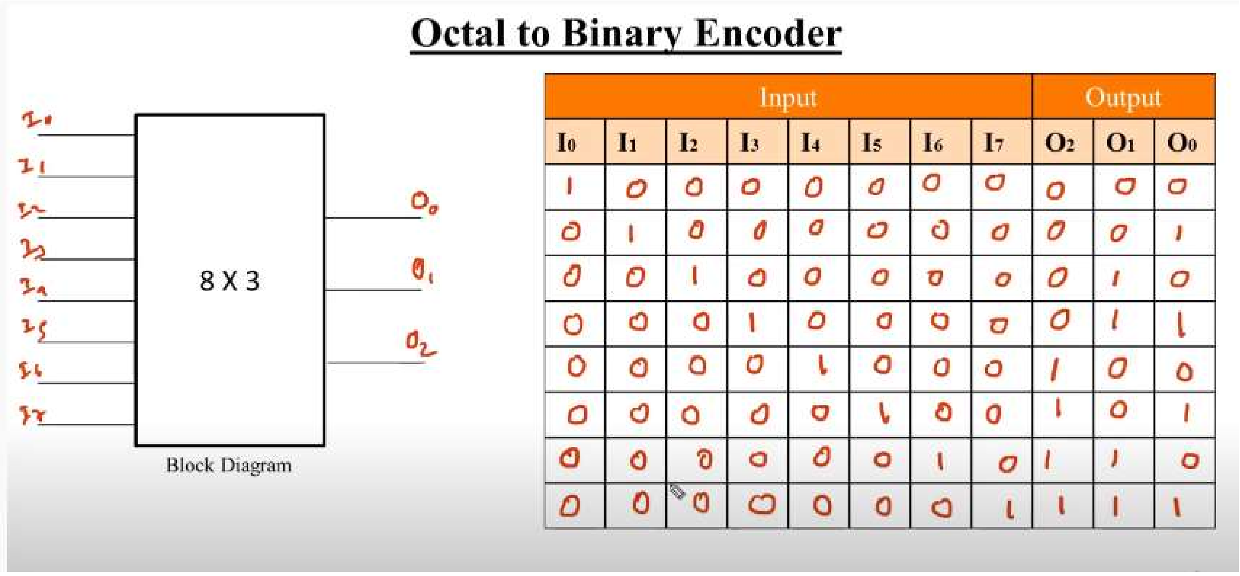


From: 0ps

To: 100,000ps

Note: To revert to EPWave opening in a new browser window, set that option on your user page.

8:3 ENCODER



Octal to Binary Encoder

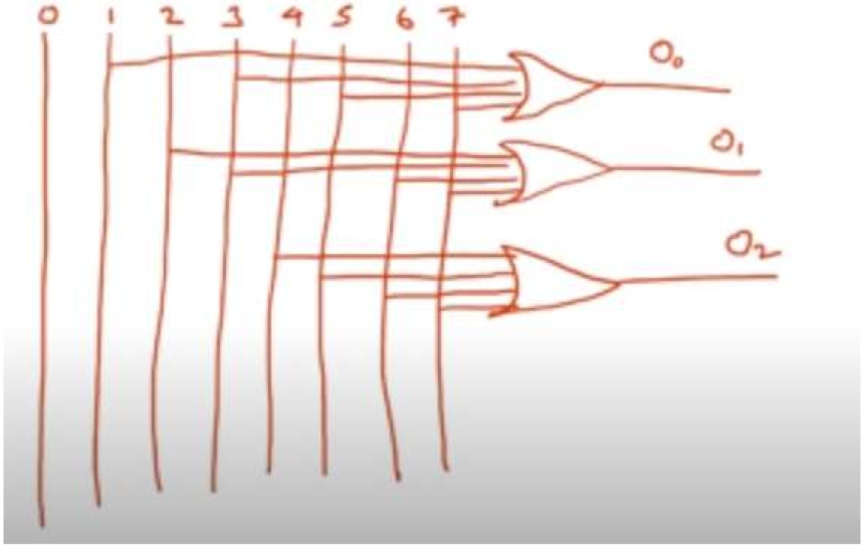
Block Diagram

Input								Output		
I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	O_2	O_1	O_0
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

O0: I1+I3+I5+I7

O1: I2+I3+I6+I7

O2: I4+I5+I6+I7



0
1
2
3
4
5
6
7
O0
O1
O2