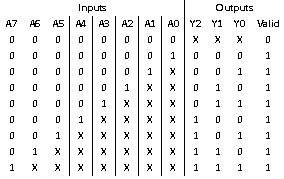
**Encoder in Digital Logic**

It has maximum of **2^n input lines** and **‘n’ output lines**, hence it encodes the information from 2^n inputs into an n-bit code. It will produce a binary code equivalent to the input, which is active High. Therefore, the encoder encodes 2^n input lines with ‘n’ bits.

### 8 : 3 Encoder (Octal to Binary) –

The 8 to 3 Encoder or octal to Binary encoder consists of **8 inputs**: Y7 to Y0 and **3 outputs** : A2, A1 & A0. Each input line corresponds to each octal digit and three outputs generate corresponding binary code.

The figure below shows the logic symbol of octal to binary encoder:



Verilog code

// Code your design here

module encoder8\_3(en, a\_in, y\_op,v);

input en;

input [7:0] a\_in;

output reg v;

output [2:0] y\_op;

reg [2:0] y\_op;

always @ (a\_in,en)

begin

if(en==0 )

begin

y\_op =3'bxxx;

v=1'b0;

end

else

begin

v=1'b1;

case (a\_in)

8'b00000001: y\_op = 3'b000;

8'b00000010: y\_op = 3'b001;

8'b00000100: y\_op = 3'b010;

8'b00001000: y\_op = 3'b011;

8'b00010000: y\_op = 3'b100;

8'b00100000: y\_op = 3'b101;

8'b01000000: y\_op = 3'b110;

8'b10000000: y\_op = 3'b111;

default: y\_op =3'bzzz;

endcase

end

end

endmodule

Test bench

// Code your testbench here

// or browse Examples

module encodertest\_tb;

// Inputs

reg en;

reg [7:0] a\_in;

// Outputs

wire v;

wire [2:0] y\_op;

// Instantiate the Unit Under Test (UUT)

encoder8\_3 uut(en,a\_in,y\_op,v);

initial begin

$dumpfile("dump.vcd");

$dumpvars(1);

a\_in = 8'b00000000;

en = 1'b0;

#10 en =1'b1; a\_in = 8'b00000001;

#10 a\_in = 8'b00000010;

#10 a\_in = 8'b00000100;

#10 a\_in = 8'b00001000;

#10 a\_in = 8'b00010000;

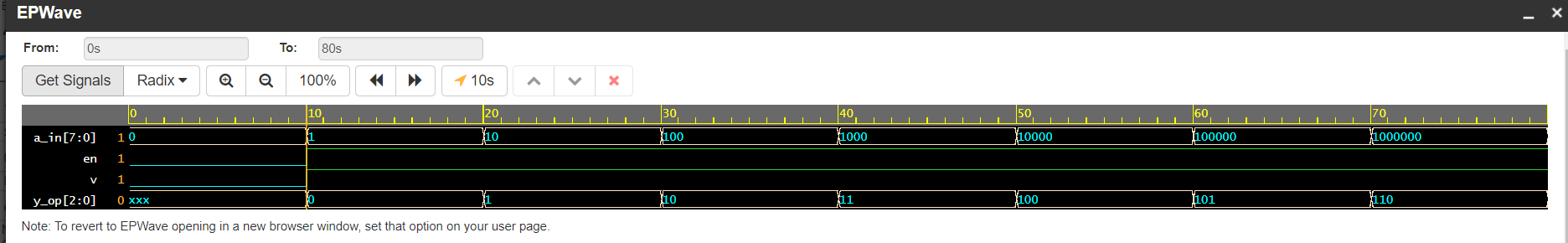
#10 a\_in = 8'b00100000;

#10 a\_in = 8'b01000000;

#10 a\_in = 8'b10000000;

end

endmodule



## **What is Priority Encoder?**

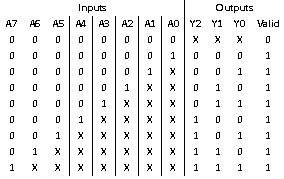
The priority encoder is a combinational logic circuit that contains 2^n input lines and n output lines and represents the highest priority input among all the input lines. When multiple input lines are active high at the same time, then the input that has the highest priority is considered first to generate the output.

It is used to solve the issues in binary encoders, which generate wrong output when more than one input line is active high. If more than one input line is active high(1) at the same time, then this encoder prioritizes every input level and allocates the priority level to each input.

The output of this encoder corresponds to the input that has the highest priority. To obtain the output, only the input with the highest priority is considered by ignoring all other input lines. This is a type of binary encoder or an ordinary encoder with a priority function. The input that has the larger magnitude or highest priority is encoded first rather than other input lines. Hence, the generated output is based on the priority assigned to the inputs.

### 8 to 3 Priority Encoder

This kind of encoder is also named an 8-bit or Octal to Binary priority encoder. This type of encoder consists of 8 inputs and 3 outputs. When multiple inputs are active high at the same time, the input with the highest priority is considered to represent the output.



module prio\_enco(en, a\_in, y\_op,v);

input en;

input [7:0] a\_in;

output reg v;

output [2:0] y\_op;

reg [2:0] y\_op;

always @ (a\_in,en)

begin

if(en==0 )

begin

y\_op =3'bxxx;

v=1'b0;

end

else

begin

v=1'b1;

case (a\_in)

8'b00000001: y\_op =3'b000;

8'b0000001x: y\_op= 3'b001;

8'b000001xx: y\_op= 3'b010;

8'b00001xxx: y\_op= 3'b011;

8'b0001xxxx: y\_op= 3'b100;

8'b001xxxxx: y\_op= 3'b101;

8'b01xxxxxx: y\_op= 3'b110;

8'b1xxxxxxx: y\_op= 3'b111;

default: y\_op=3'bzzz;

endcase

end

end

endmodule

Test bench

// Code your testbench here

// or browse Examples

module encoder\_prio\_test\_tb;

// Inputs

reg en;

reg [7:0] a\_in;

// Outputs

wire v;

wire [2:0] y\_op;

// Instantiate the Unit Under Test (UUT)

prio\_enco uut(en,a\_in,y\_op,v);

initial begin

$dumpfile("dump.vcd");

$dumpvars(1);

a\_in = 8'b00000000;

en = 1'b0;

#10 en =1'b1; a\_in = 8'b00000001;

#10 a\_in = 8'b00000010;

#10 a\_in = 8'b00000100;

#10 a\_in = 8'b00001000;

#10 a\_in = 8'b00010000;

#10 a\_in = 8'b00100000;

#10 a\_in = 8'b01000000;

#10 a\_in = 8'b10000000;

end

endmodule