LOGIC DESIGN EET-1021

CHAPTER 04

Lecture 22

Combinational Logic

Overview of previous lecture

- **What is Decimal adder**
- **How to construct this adder circuit**
- **What is a Magnitude Comparator circuit**

Decoder

- A *decoder is a combinational circuit that converts binary information from n input lines to a maximum of* 2^n unique output lines.

- If the *n -bit coded information has* unused combinations, the decoder may have fewer than 2^noutputs.

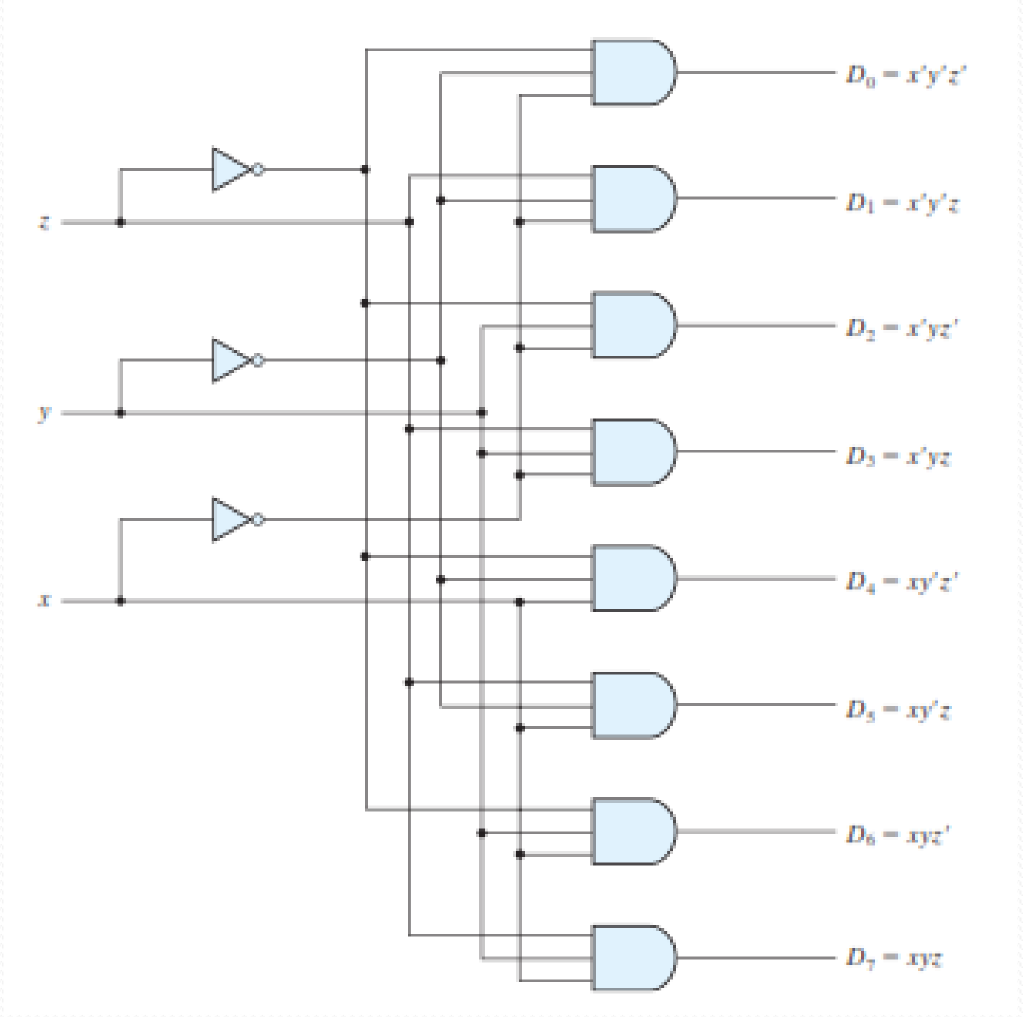
- The decoders presented here are called *n -to- m -line decoders, where* $m \leq 2^n$

- The name *decoder is also used in conjunction with* other code converters, such as a BCD-to-seven-segment decoder.

Truth Table of a Three-to-Eight-Line Decoder

For each possible input combination, there are seven outputs that are equal to 0 and only one that is equal to 1. The output whose value is equal to 1 represents the minterm equivalent of the binary number currently available in the input lines.

Inputs			**Outputs**							
x	y	z	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1



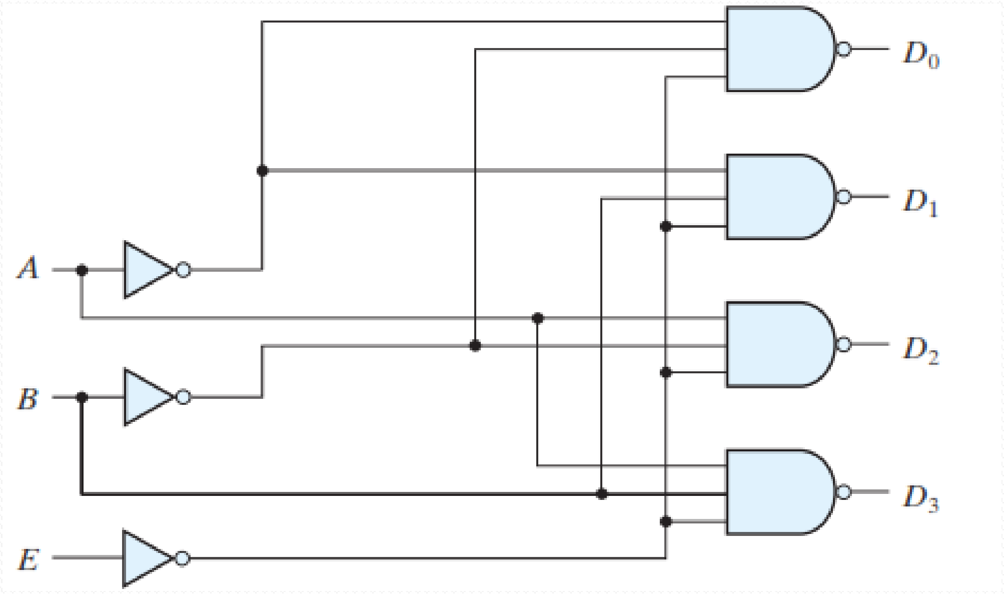
Three-to-eight-line decoder

A decoders include one or more *enable inputs to control the circuit operation. A two-to-four-line decoder with an* enable input is constructed with NAND gates.

The circuit operates with complemented outputs and a complement enable input. The decoder is enabled when *E is equal to 0 (i.e., active-low enable).*

E	A	B	D_0	D_1	D_2	D_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	1

Truth table



Logic Diagram

- A decoder with enable input can function as a *demultiplexer*— *a circuit that receives* information from a single line and directs it to one of 2^n possible output lines. The selection of a specific output is controlled by the bit combination of *n selection lines*.

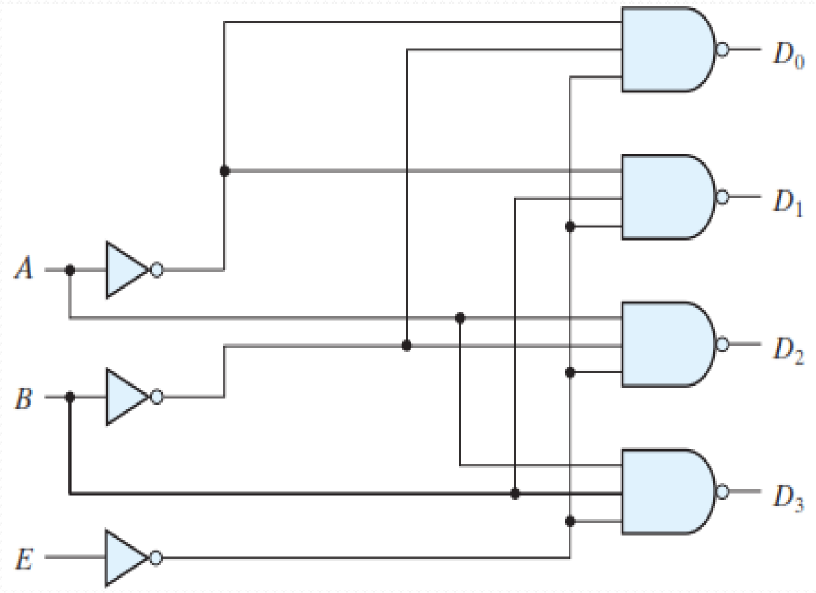
- The decoder above shown can function as a one-to-four-line demultiplexer when E *is* taken as a data input line and A *and* B *are taken as the selection inputs.*

- *The single* input variable E *has a path to all four outputs, but the input information is directed to* only one of the output lines, as specified by the binary combination of the two selection lines A *and* B .

- As decoder and demultiplexer operations are obtained from the same circuit, a decoder with an enable input is referred to as a *decoder – demultiplexer* .

HDL for 2 to 4 line Decoder:

```
// Gate-level description of two-to-four-line decoder
  module  decoder_2x4_gates (D, A, B, enable);
 output  [0: 3] D;
 input   A, B, enable;
wire   A_not,B_not, enable_not;
not
G1  (A_not, A),
G2  (B_not, B),
G3  (enable_not, enable);
 nand
 G4  (D[0], A_not, B_not, enable_not),
G5  (D[1], A_not, B, enable_not),
 G6  (D[2], A, B_not, enable_not),
 G7  (D[3], A, B, enable_not);
endmodule
```



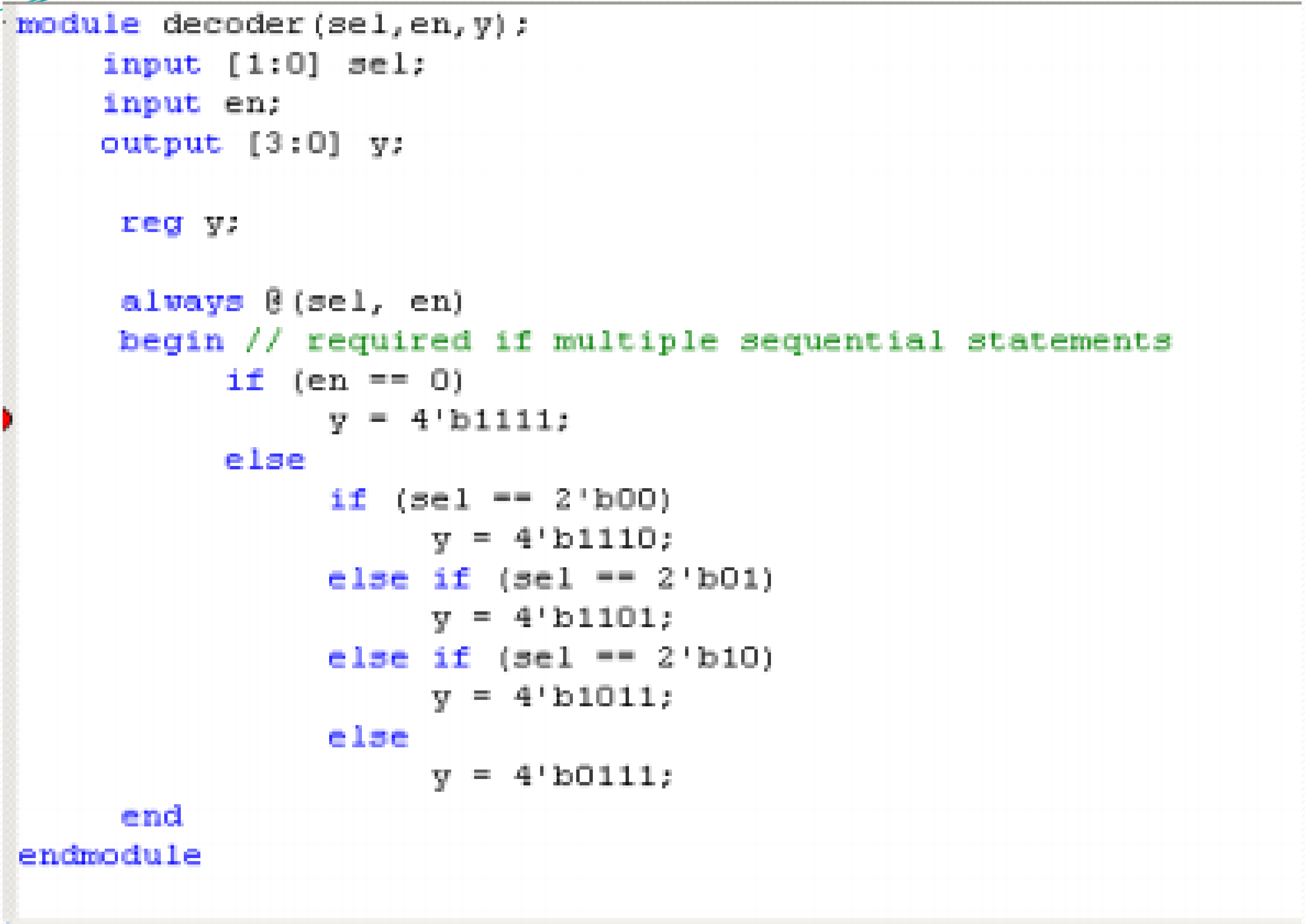
Dataflow Modeling

```
//Dataflow description of a 2-to-4-line decoder
module decoder_df (A,B,E,D);
   input A,B,E;
   output [0:3] D;
   assign D[0] = ~(~A & ~B & ~E),
          D[1] = ~(~A & B & ~E),
          D[2] = ~(A & ~B & ~E),
          D[3] = ~(A & B & ~E);
endmodule
```

A 2-to-1 line multiplexer with data inputs A and B, select input S, and output Y is described with the continuous assignment

```
assign Y = (A & S) | (B & ~S)
```

Behavioral Model

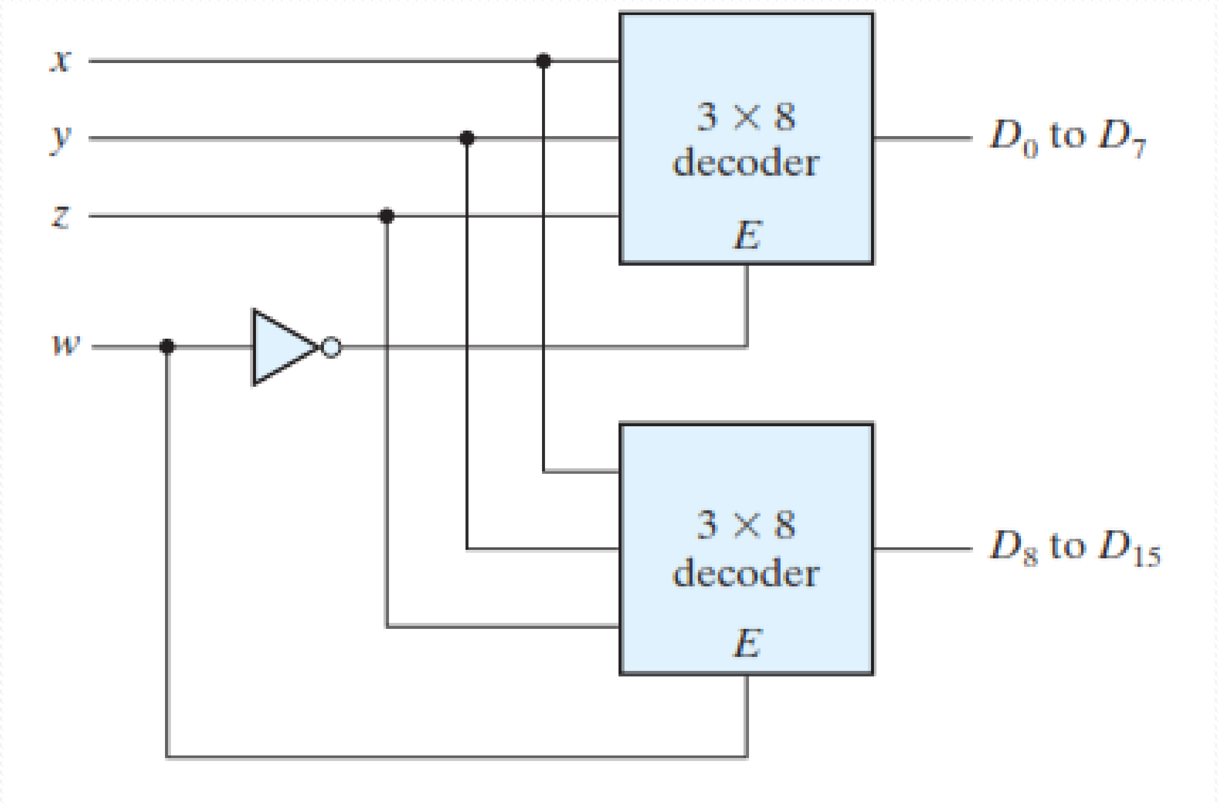


```
module decoder(sel,en,y);
     input [1:0] sel;
     input en;
     output [3:0] y;

      reg y;

      always @(sel, en)
      begin // required if multiple sequential statements
           if (en == 0)
                y = 4'b1111;
           else
                if (sel == 2'b00)
                     y = 4'b1110;
                else if (sel == 2'b01)
                     y = 4'b1101;
                else if (sel == 2'b10)
                     y = 4'b1011;
                else
                     y = 4'b0111;
      end
endmodule
```

Decoders with enable inputs can be connected together to form a larger decoder circuit



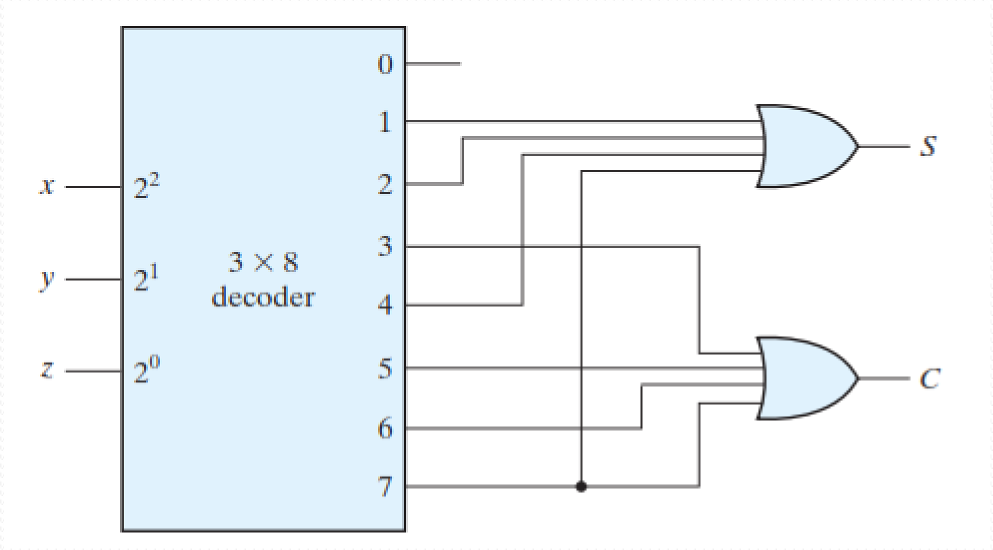
4×16 decoder constructed with two 3×8 decoders

Combinational Logic Implementation

Since any Boolean function can be expressed in sum-of-minterms form, a decoder that generates the minterms of the function, together with an external OR gate that forms their logical sum, provides a hardware implementation of the function. In this way, any combinational circuit with n *inputs and* m *outputs can be implemented with an* n*-to-*2^n -line decoder and m *OR* gates.

$$S(x, y, z) = \Sigma(1, 2, 4, 7)$$

$$C(x, y, z) = \Sigma(3, 5, 6, 7)$$



Implementation of a full adder with a decoder

Encoder

- An encoder is a digital circuit that performs the inverse operation of a decoder. An encoder has 2^n(or fewer) input lines and *n* *output lines*. *The output lines, as an aggregate,* generate the binary code corresponding to the input value.

- Let us consider an example of an encoder i.e the **octal-to-binary encoder**

It has eight inputs (one for each of the octal digits) and three outputs that generate the corresponding binary number. It is assumed that only one input has a value of 1 at any given time.

Inputs								Outputs		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x	y	z
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

Truth Table of an Octal-to-Binary Encoder

The encoder can be implemented with OR gates whose inputs are determined directly from the truth table. These conditions can be expressed by the following Boolean output functions:

$$
\begin{aligned}
z &= D_1 + D_3 + D_5 + D_7 \\
y &= D_2 + D_3 + D_6 + D_7 \\
x &= D_4 + D_5 + D_6 + D_7
\end{aligned}
$$

The encoder can be implemented with three OR gates.

Limitation

- The encoder defined above has the limitation that only one input can be active at any given time. If two inputs are active simultaneously, the output produces an undefined combination.

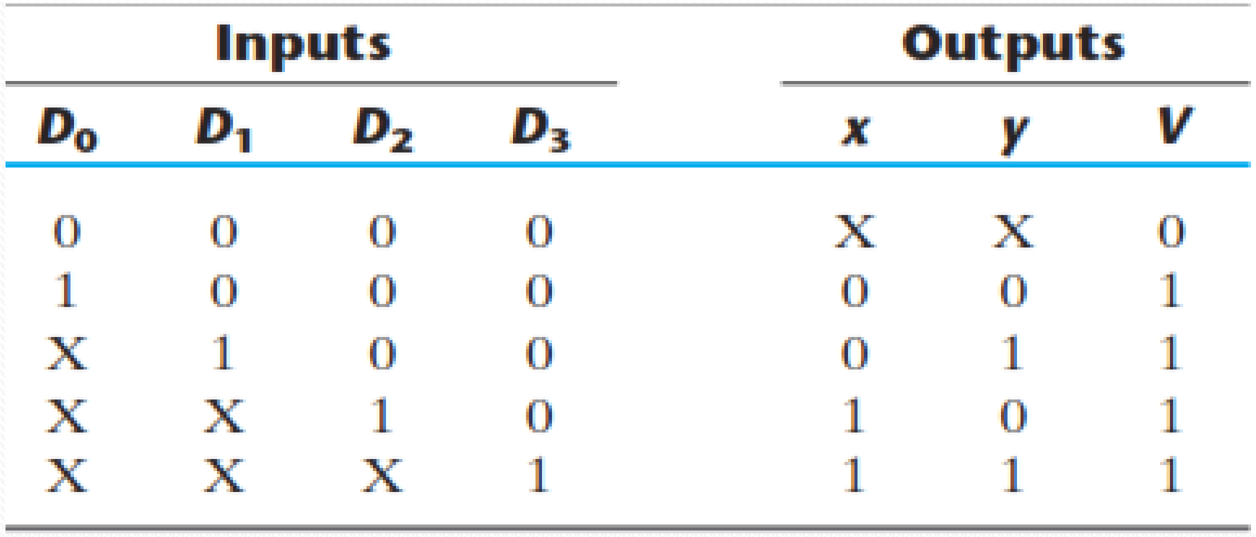
- To resolve this ambiguity, encoder circuits must establish an input priority to ensure that only one input is encoded.

- Another ambiguity in the octal-to-binary encoder is that an output with all 0's is generated when all the inputs are 0; but this output is the same as when D is equal to 1. The discrepancy can be resolved by providing one more output to indicate whether at least one input is equal to 1.

Priority Encoder

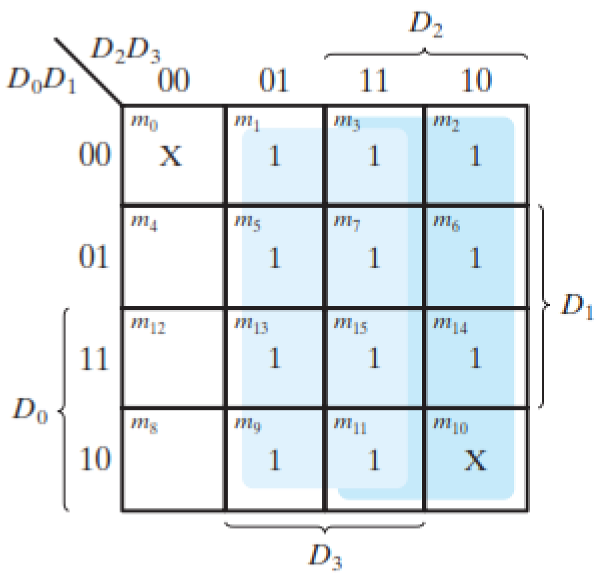
- A priority encoder is an encoder circuit that includes the priority function.
- The truth table of a four-input priority encoder is given below.

Truth Table of a Priority Encoder

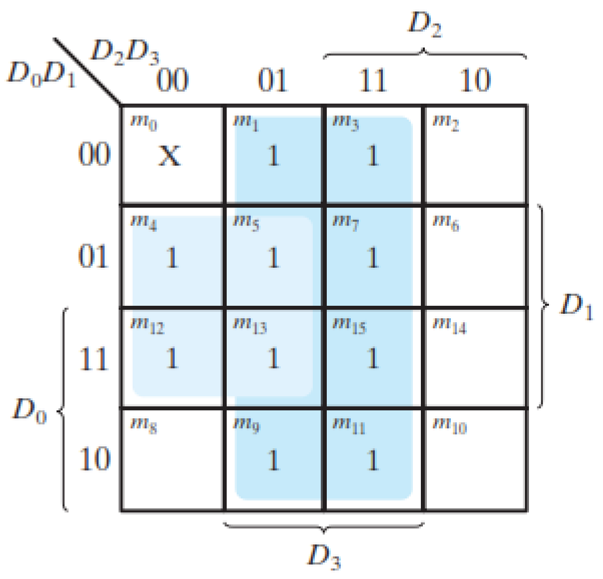


Inputs				Outputs		
D_0	D_1	D_2	D_3	x	y	V
0	0	0	0	X	X	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1

In addition to the two outputs x and y , the circuit has a third output designated by V ; this is a valid bit indicator that is set to 1 when one or more inputs are equal to 1. Note that whereas X 's in output columns represent don't-care conditions, the X 's in the input columns are useful for representing a truth table in condensed form

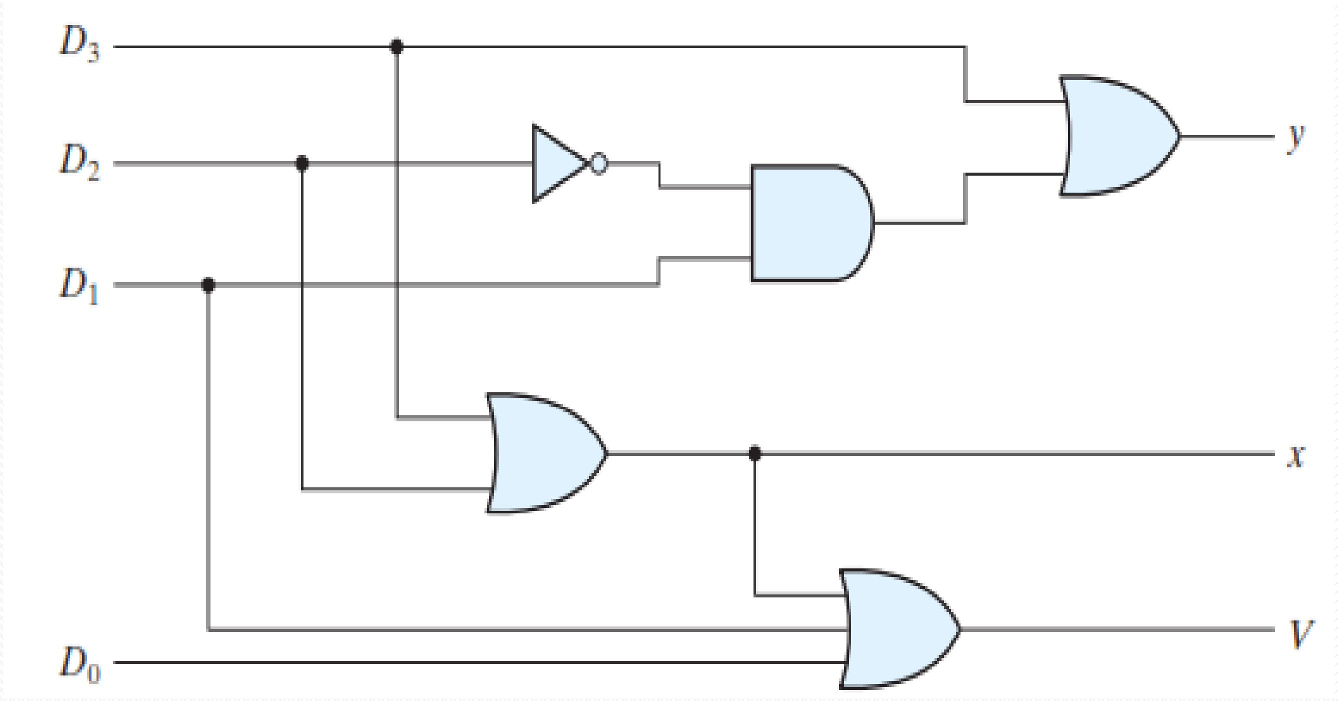


$x = D_2 + D_3$



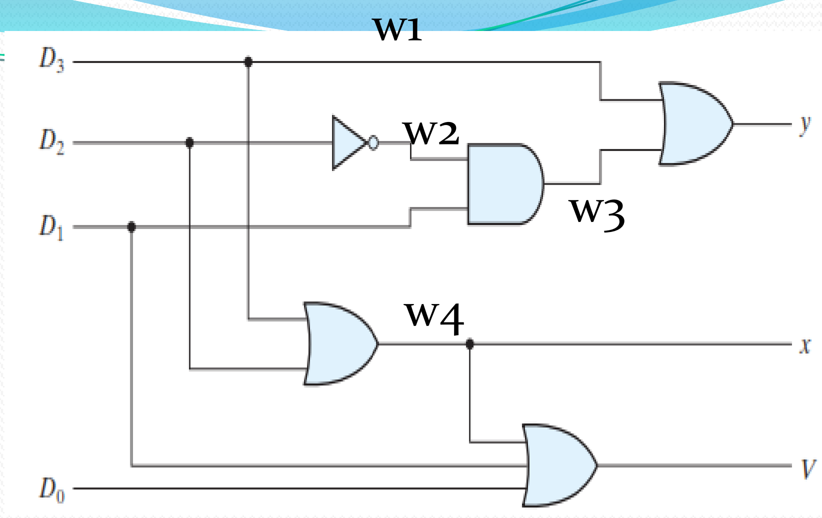
$y = D_3 + D_1D'_2$

$$x = D_2 + D_3$$
$$y = D_3 + D_1\,D'_2$$
$$V = D_0 + D_1 + D_2 + D_3$$



Four-input priority encoder

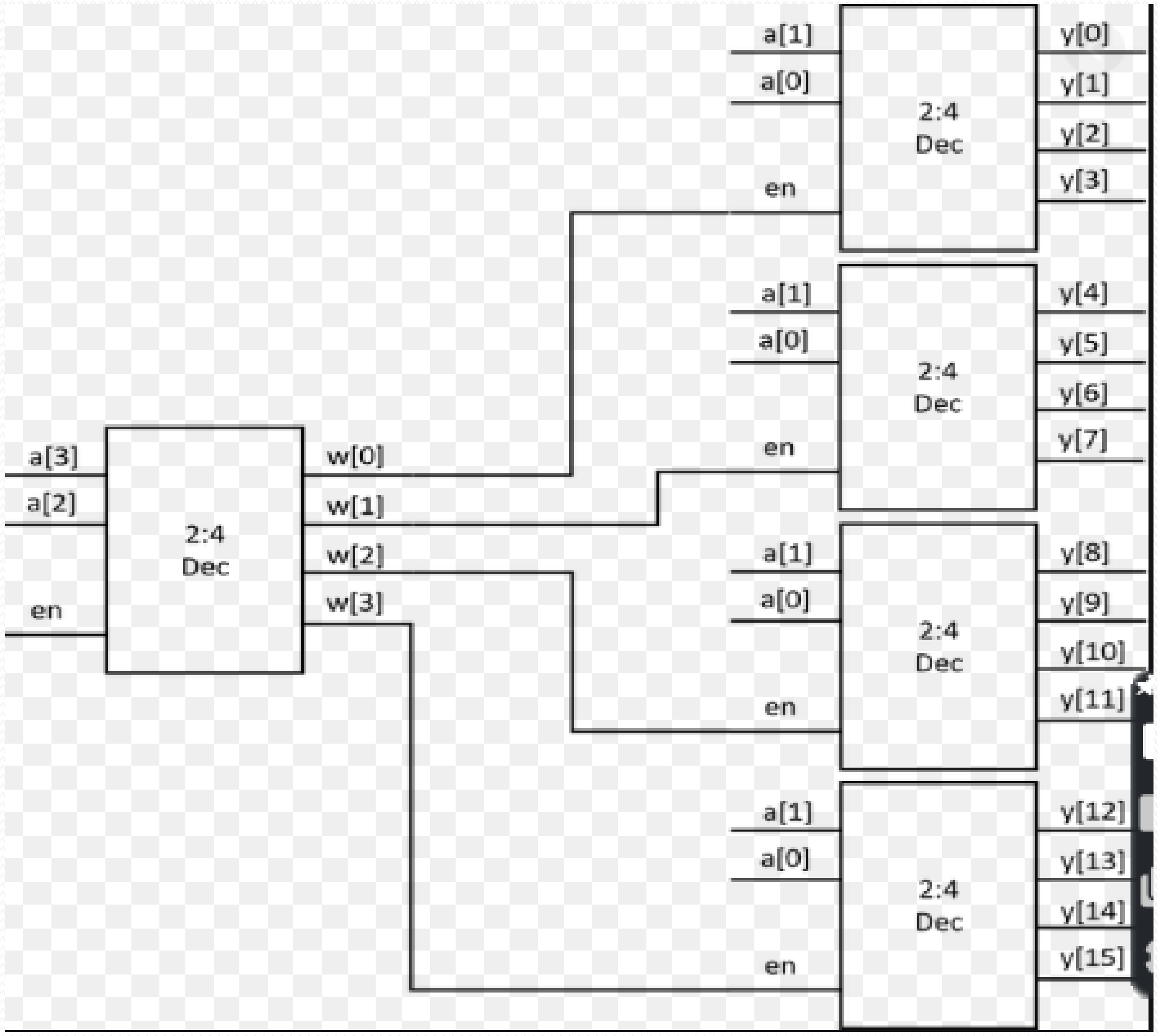
HDL for Priority Encoder



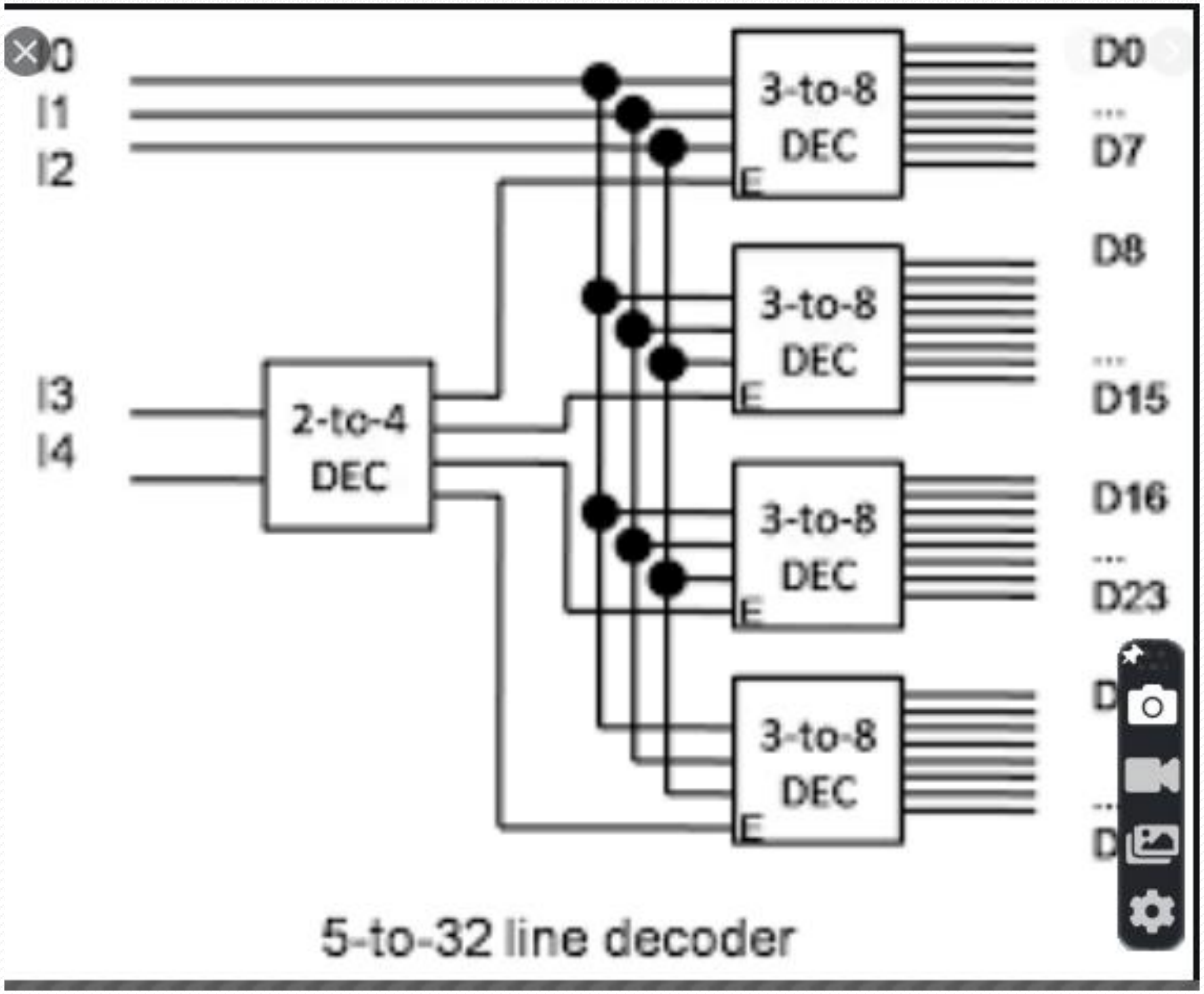
```
module PriorityEncoder_4Bit(
    input [0:3] D,
    output [1:0] Y,
    output V
    );
 reg [1:0] Y;
 reg V;
 always @(D)
 begin
Y[1] <= D[2] | D[3];
Y[0] <= D[3] | D[1] & ~D[2];
V = D[0] | D[1] | D[2] | D[3];
end
endmodule
```

Example

Construct a 4-to-16-line decoder with five 2-to-4-line decoders with enable.



Construct a 5-to-32-line decoder with four 3-to-8-line decoders with enable and a 2-to4-line decoder. Use block diagrams for the components.



THANK YOU