# Decoders & Encoders

## Decoder

In a digital system, discrete quantities of information are represented with binary codes. A binary code of *n* bits can represent up to $2^n$ distinct elements of the coded information. A ***decoder*** is a **combinational circuit that converts *n* bits of binary information of input lines to a maximum of $2^n$ unique output lines.** Usually decoders are designated as an *n to m lines decoder*, where *n* is the number of input lines and *m* (=$2^n$) is the number of output lines.

Decoders have a wide variety of applications in digital systems such as data demultiplexing, digital display, digital to analog converting, memory addressing, etc. A 3-to-8 line decoder is illustrated in Figure 1.

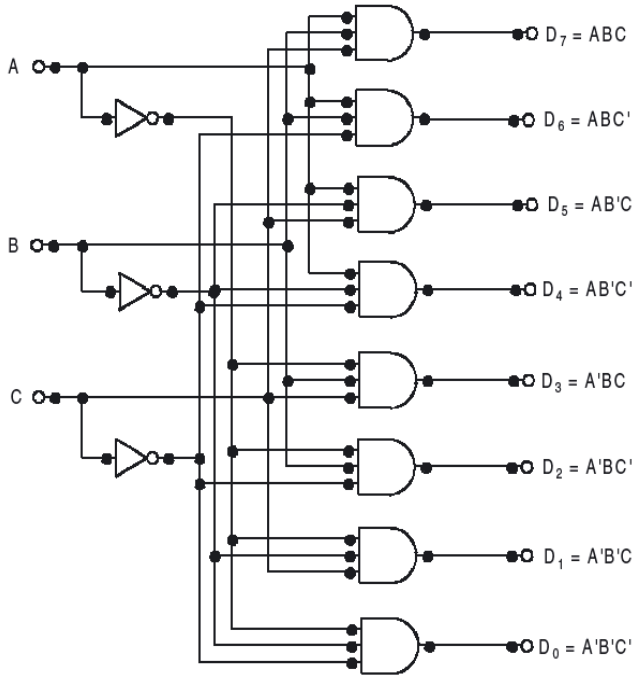



Figure (1)

| *Input variables* | | | *Outputs* | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| *A* | *B* | *C* | $D_0$ | $D_1$ | $D_2$ | $D_3$ | $D_4$ | $D_5$ | $D_6$ | $D_7$ |
| 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 |

The 3-to-8 line decoder consists of three input variables and eight output lines. Note that each of the output lines represents one of the minterms generated from three variables. The internal combinational circuit is realized with the help of INVERTER gates and AND gates. The operation of the decoder circuit may be further illustrated from the input output relationship as given in the above table. Note that the output variables are mutually exclusive to each other, as only one output is possible to be logic 1 at any one time.

## ***Some Applications of Decoders***

As we have seen that decoders give multiple outputs equivalent to the minterms corresponding to the input variables, it is obvious that any Boolean expression in the sum of the products form can be very easily implemented with the help of decoders. It is not necessary to obtain the minimized expression through simplifying procedures like a Karnaugh map, or tabulation method, or any other procedure. It is sufficient to inspect the minterm contents of a function from the truth table, or the canonical form of sum of the products of a Boolean expression and selected minterms obtained from the *output lines of a decoder may be simply OR-gated to derive the required function*. The following examples will demonstrate this.

***Example 1. Implement the function F (A,B,C) = Σ (1,3,5,6).***

***Solution.*** Since the above function has three input variables, a 3-to-8 line decoder may be employed. It is in the sum of the products of the minterms *m*1, *m*3, *m*5, and *m*6, and so decoder output D1, D3, D5, and D6 may be OR-gated to achieve the desired function. The combinational circuit of the above functions is shown in Figure 2.

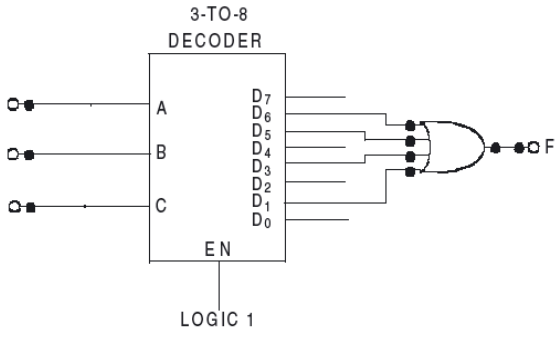


Figure (2)

***Example 2. Design a full adder circuit with decoder IC.***

***Solution.*** We have seen earlier that full adder circuits are implemented with logic gates. This can be very easily implemented with the help of a decoder IC. If we observe the truth table of a full adder. In respect to minterms, the Boolean expression of sum output S and carry output C can be written as:

S = X′A′B + X′AB′ + XA′B′ + XAB  and  C = X′AB + XA′B + XAB′ + XAB.

The above expression can be realized in Figure 3.

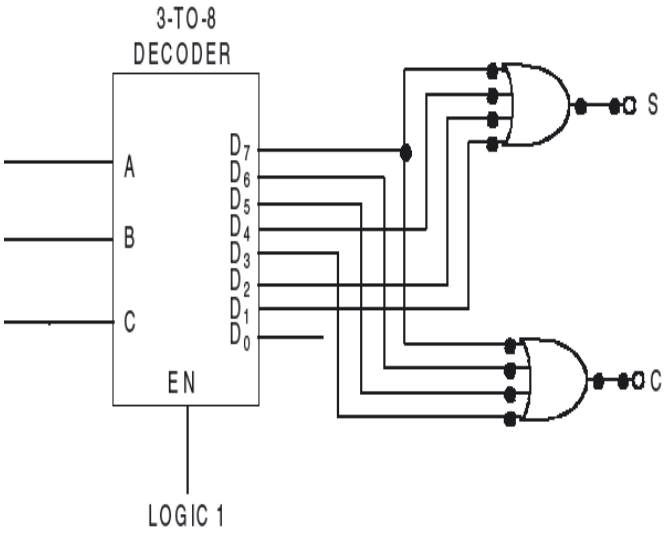
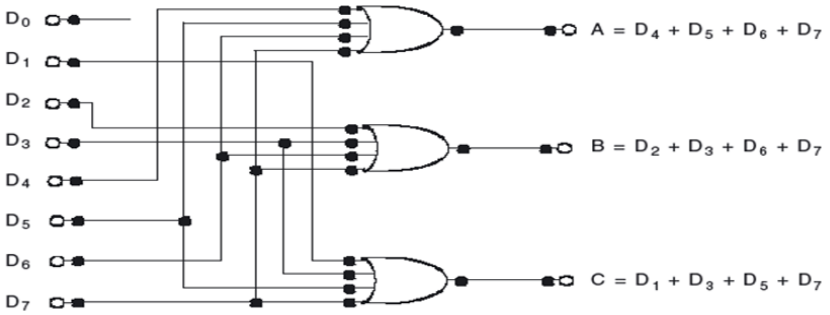



Figure (3)

## ***Encoders***

An ***encoder*** is a combinational network that performs the reverse operation of the decoder. An encoder has $2^n$ or less numbers of inputs and *n* output lines. The output lines of an encoder generate the binary code for the $2^n$ input variables. Figure 4 illustrates an eight inputs/three outputs encoder. It may also be referred to as an octal-to-binary encoder where binary codes are generated at outputs according to the input conditions. The truth table is given bellow.

Figure (4)




| *Inputs* | | | | | | | | *Outputs* | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $D_0$ | $D_1$ | $D_2$ | $D_3$ | $D_4$ | $D_5$ | $D_6$ | $D_7$ | *A* | *B* | *C* |
| 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 |

The encoder in Figure 4 assumes that only one input line is activated to logic 1 at any particular time, otherwise the other circuit has no meaning. It may be noted that for eight inputs there are a possible $2^8 = 256$ combinations, but only eight input combinations are useful and the rest are don't-care conditions. It may also be noted that $D_0$ input is not connected to any of the gates. All the binary outputs A, B, and C must be all 0s in this case. All 0s output may also be obtained if all input variables $D_0$ to $D_7$ are logic 0. This is the main discrepancy of this circuit. This discrepancy can be eliminated by introducing another output indicating the fact that all the inputs are not logic 0.

However, this type of encoder is not available in an IC package because it is not easy to implement with OR gates and not much of the gates are used. The type of encoder available in IC package is called a *priority encoder*. These encoders establish an input priority to ensure that only highest priority input is encoded. As an example, if both $D_2$ and $D_4$ inputs are logic 1 simultaneously, then output will be according to $D_4$ only *i.e.*, output is 100.