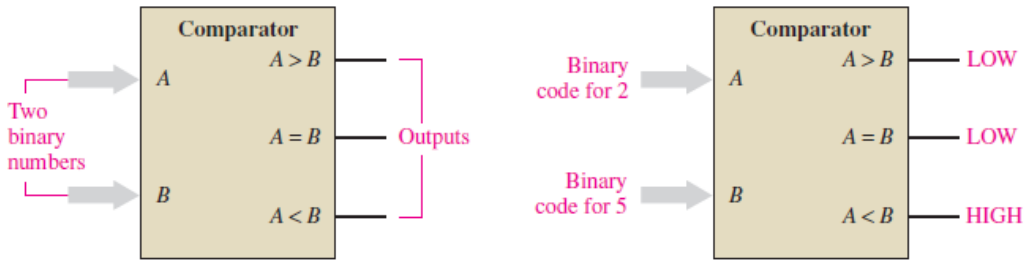
The Comparison Function

Magnitude comparison is performed by a logic circuit called a **comparator**, covered in A **comparator** compares two quantities and indicates whether or not they are equal.

For example, suppose you have two numbers and wish to know if they are equal or not equal and, if not equal,
which is greater. The comparison function is represented in



(a) Basic magnitude comparator

(b) Example: A is less than B $(2 < 5)$ as indicated by the HIGH output $(A < B)$

FIGURE 1 The comparison function.

Figure 1. One number in binary form (represented by logic levels) is applied to input A, and the other number in binary form (represented by logic levels) is applied to input B. The outputs indicate the relationship of the two numbers by producing a HIGH level on the proper output line.

Suppose that a binary representation of the number 2 is applied to input
A and a binary representation of the number 5 is applied to input B.
A HIGH level will appear on the A < B (A is less than B) output,
indicating the relationship between the two numbers (2 is less than 5).
The wide arrows represent a group of parallel lines on which the bits are transferred.

you should be able to

Use the exclusive-NOR gate as a basic comparator

Analyze the internal logic of a magnitude comparator that has both equality and inequality outputs Apply the 74HC85 comparator to compare the magnitudes of two 4-bit numbers

Cascade 74HC85s to expand a comparator to eight or more bits

Equality

the exclusive-NOR(**XNOR**) gate can be used as a basic comparator
because its output is a 0 if the two input bits are not equal and a 1 if the input bits are equal

Figure 2 , shows the exclusive-NOR gate as a 2-bit comparator.

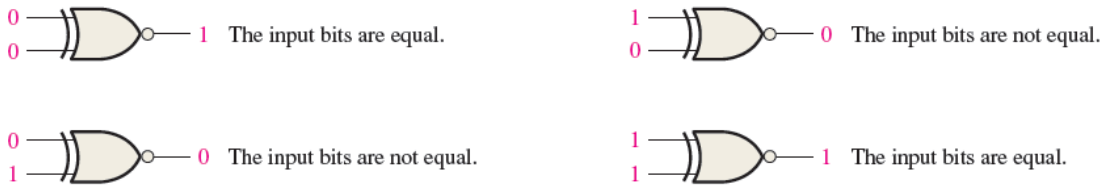
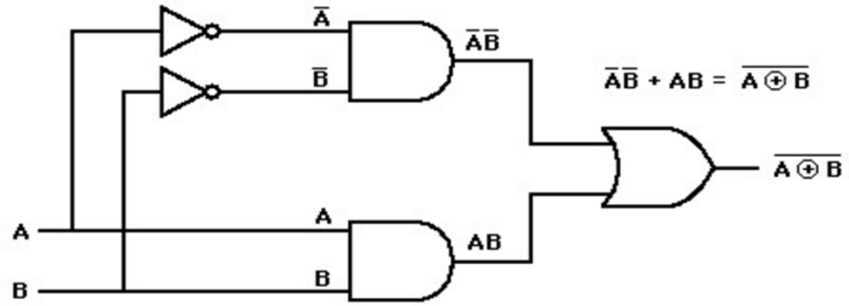


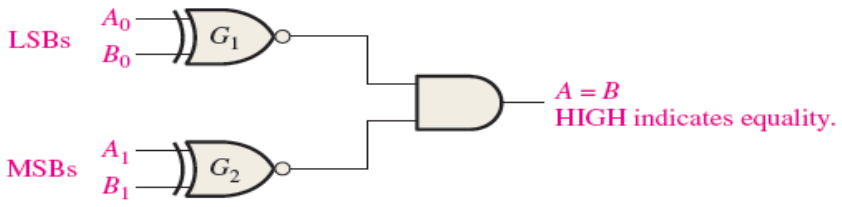
FIGURE 2 Basic comparator operation.



Truth table of XNOR gate

Inputs		Output
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

In order to compare binary numbers containing two bits each,
An- additional exclusive NOR gate is necessary.
The two least significant bits (LSBs) of the two numbers are compared by gate G1, and the two most significant bits (MSBs) are compared by gate G2, as shown in Figure 3.
If the two numbers are equal, their corresponding bits are the sam
and the output of each exclusive-NOR gate is a 1. If the corresponding sets of bits are not equal, a 0 occurs on that exclusive-NOR gate output.



General format: Binary number $A \rightarrow A_1A_0$
Binary number $B \rightarrow B_1B_0$

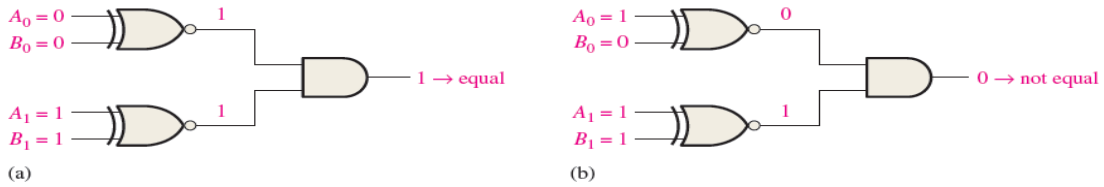
FIGURE 3 Logic diagram for equality comparison of two 2-bit numbers.

In order to produce a single output indicating an equality or inequality of two numbers, an AND gate can be combined with XNOR gates,

as shown in Figure 3 The output of each exclusive-NOR gate is applied to the AND gate input. When the two input bits for each exclusive-NOR are equal, the corresponding bits of the numbers are equal, producing
a 1 on both inputs to the AND gate and thus a 1 on the output. When the two numbers are not equal, one or both sets of corresponding bits are unequal, and a 0 appears on at least one input to the AND gate to produce a 0 on its output. Thus, the output of the AND gate indicates equality (1) or inequality (0) of the two numbers

EXAMPLE- 1 Apply each of the following sets of binary numbers to the comparator inputs, and determine the output by following the logic levels through the circuit

(a) 10 and 10 , (b) 11 and 10



Inequality

In addition to the equality output, fixed-function comparators can provide additional outputs that indicate which of the two binary numbers being compared is the larger. That is,
there is an output that indicates when number *A* is greater than number *B* ($A > B$) and an output that indicates when number *A* is less than number *B* ($A < B$), as shown in the logic symbol for a 4-bit comparator in Figure 4
To determine an inequality of binary numbers *A* and *B*, you first examine the highest order bit in each number. The following conditions are possible:
1. If $A_3 = 1$ and $B_3 = 0$, number *A* is greater than number *B*.
2. If $A_3 = 0$ and $B_3 = 1$, number *A* is less than number *B*.
3. If $A_3 = B3$, then you must examine the next lower bit position for an inequality.
These three operations are valid for each bit position in the numbers. The general procedure used in a comparator is to check for an inequality in a bit position, starting with the highest-order bits (MSBs). When such an inequality is found, the relationship of the two numbers is established, and any other inequalities
in lower-order bit positions must be
ignored because it is possible for an opposite indication to occur; *the highest-order indication must take precedence.*

Magnitude Comparator – a Magnitude Comparator is a digital comparator which has three output terminals, one each for equality, A = B greater than, A > B and less than A < B, shows the inner logic circuit for 1 bit magnitude comparator

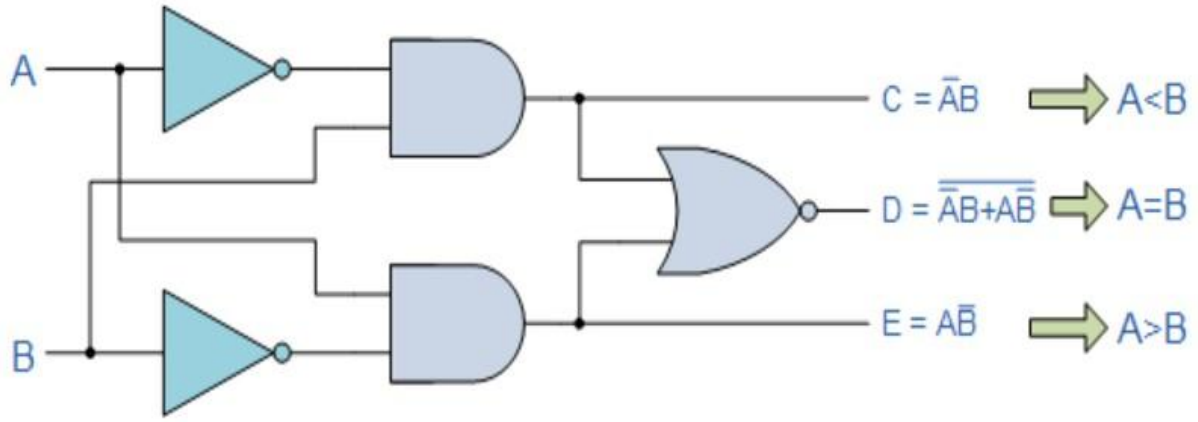


Figure the inner logic circuit for **1** bit magnitude comparator

Inputs		Outputs		
B	A	A > B	A = B	A < B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

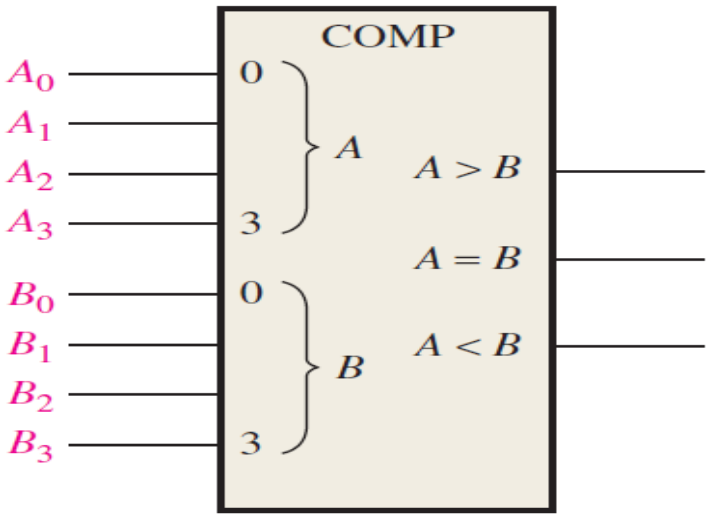
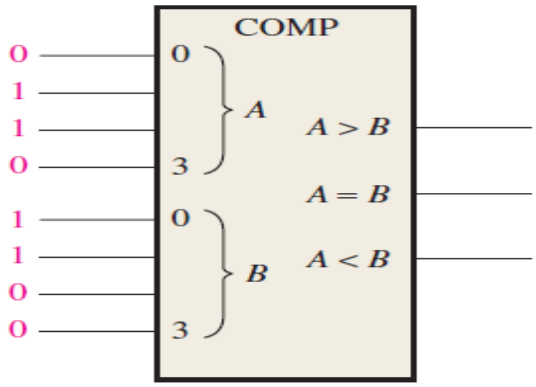


FIGURE 4 Logic symbol for a 4-bit comparator with inequality indication.

EXAMPLE 2

Determine the A = B, A > B, and A < B outputs for the input numbers shown on the comparator



Solution

The number on the A inputs is 0110 and the number on the B inputs is 0011. The A > B
output is HIGH and the other outputs are LOW

Related Problem

What are the comparator outputs when $A_3A_2A_1A_0$ = 1001 and $B_3B_2B_1B_0$ = 1010?

Implementation: 4-Bit Magnitude Comparator

Fixed-Function Device The 74HC85/74LS85 pin diagram and logic symbol are shown in Figure 5.

Notice that this device has all the inputs and outputs of the generalized comparator previously discussed and, in addition, has three cascading inputs: **A < B, A = B, A > B.**

These inputs allow several comparators to be cascaded for comparison of any number of bits greater than four. To expand the comparator the **A < B, A = B**, and **A > B** outputs of the lower-order comparator are connected to the corresponding cascading inputs of the next higher-order comparator. The lowest order comparator must have a HIGH on the **A = B** input and LOWs on the **A< B** and **A >B** inputs.

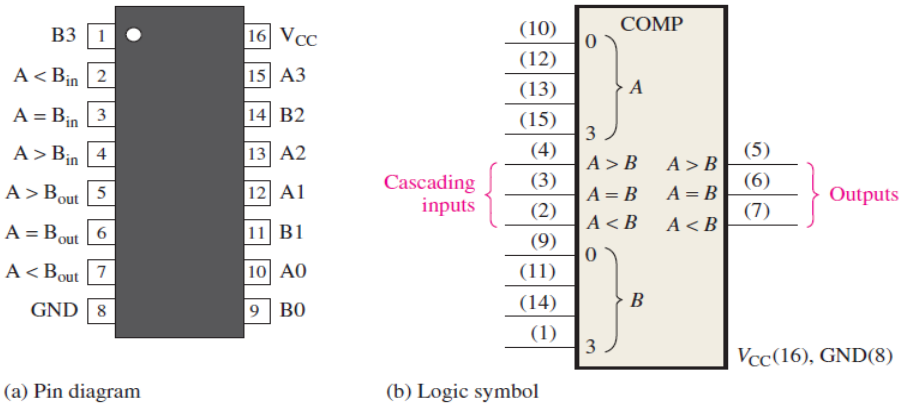


FIGURE 5 The 74HC85/74LS85 4-bit magnitude comparator.

EXAMPLE 3

Use 74HC85 comparators to compare the magnitudes of two 8-bit numbers. Show the comparators with proper interconnections

Solution

Two 74HC85s are required to compare two 8-bit numbers. They are connected as shown in Figure 6 in a cascaded arrangement.

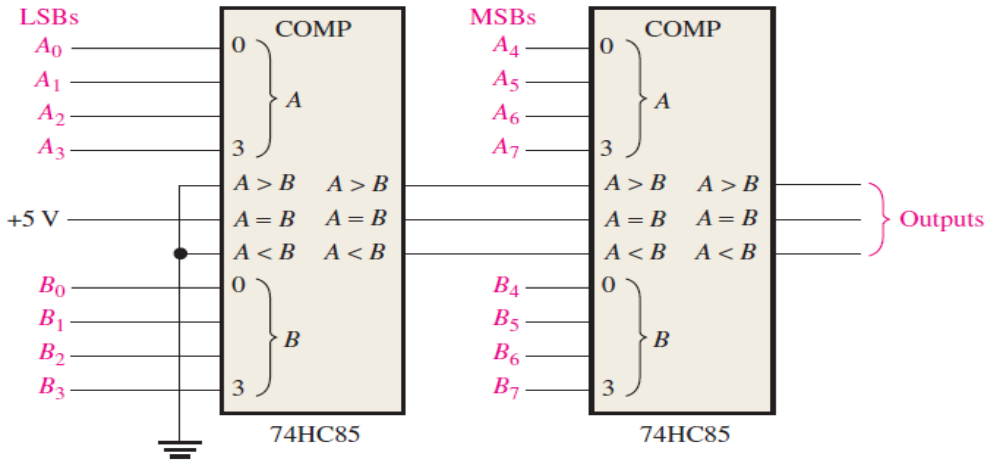


FIGURE 6– An 8-bit magnitude comparator using two 74HC85s.

Not :XOR=($A \oplus B$)

Example

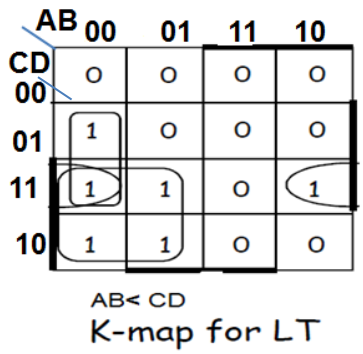
If we have 4-bit in input 2-bit for (**A** and **B**)

2-BIT for (**C** and **D**)

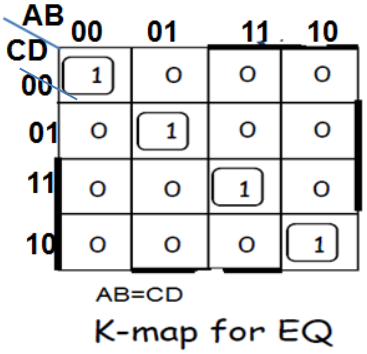
Solution:

1- Truth table of 4-Bit Comparator

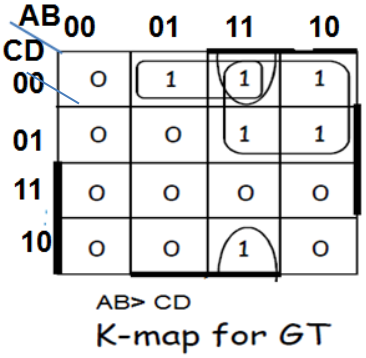
A	B	C	D	G	E	L
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	0



K-map for LT



K-map for EQ



K-map for GT

LT = A' B' D + A' C + B' C D (AB< CD)

EQ = A'B'C'D' + A'BC'D + ABCD + AB'CD' = (A xnor C) • (B xnor D)
(AB=CD)

GT = B C' D' + A C' + A B D' (AB> CD)

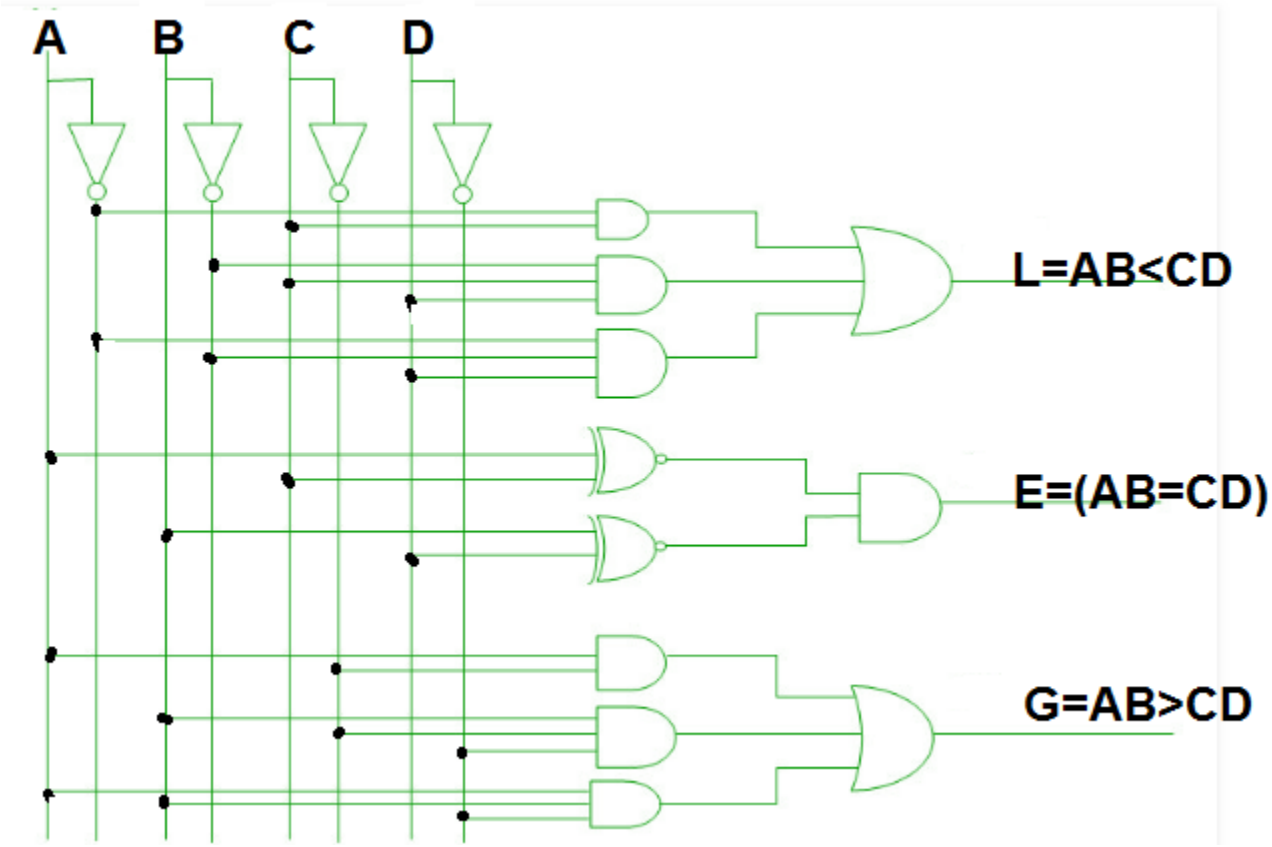
two alternative
implementations of EQ without XOR…….

with XOR

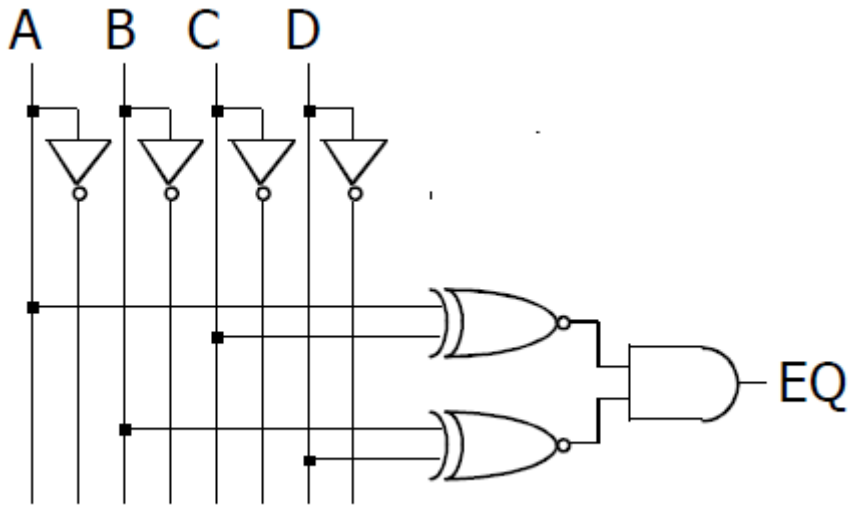
EQ = $\overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}B\overline{C}D + ABCD + A\overline{B}C\overline{D}$=

=$\overline{A}\,\overline{C}$ (BD +$\overline{B}\,\overline{D}$)+AC(BD+$\overline{B}\,\overline{D}$)=($\overline{A}\,\overline{C}$ +AC) (BD+$\overline{B}\,\overline{D}$)

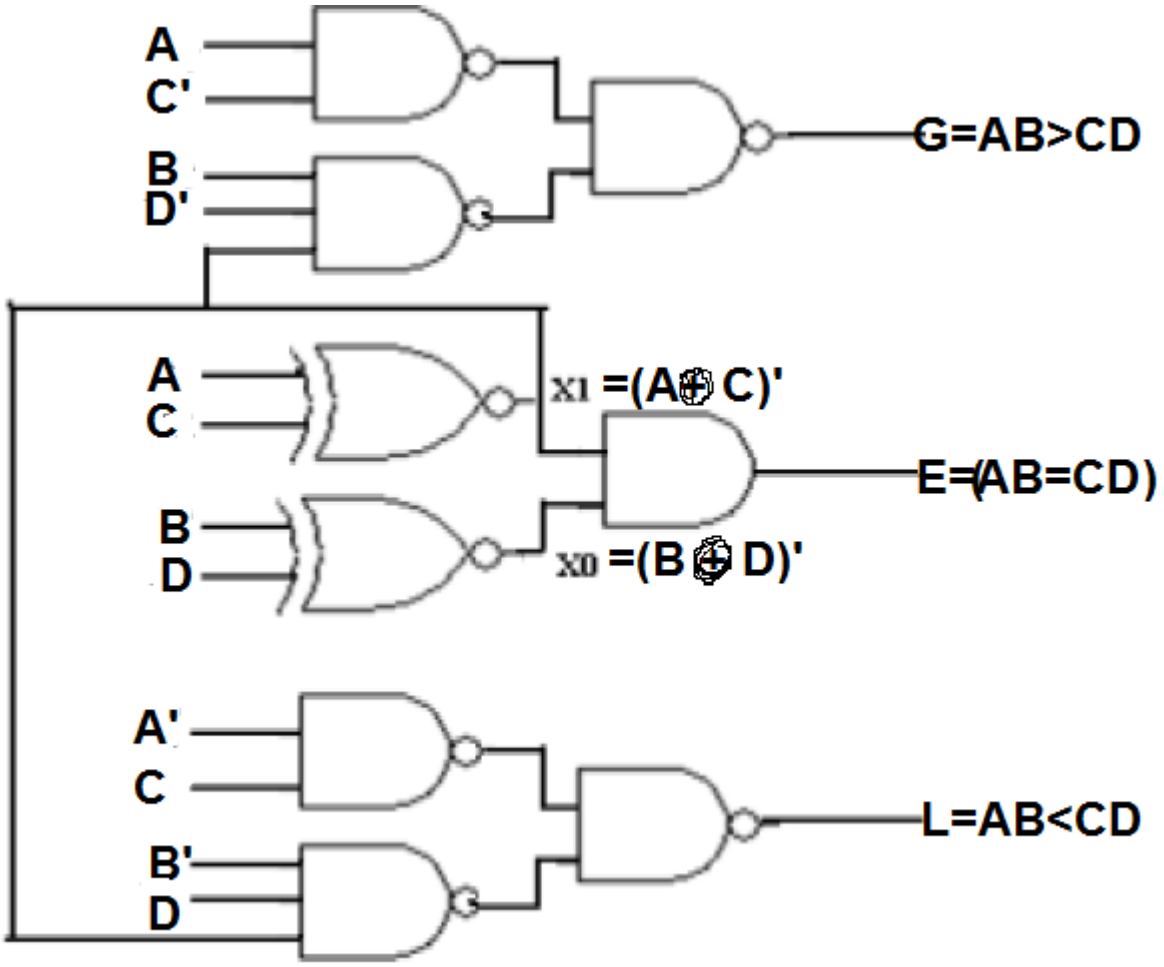
EQ= $\overline{(A \oplus C)}\ \overline{(B \oplus D)}$



2-WITH XNOR gate



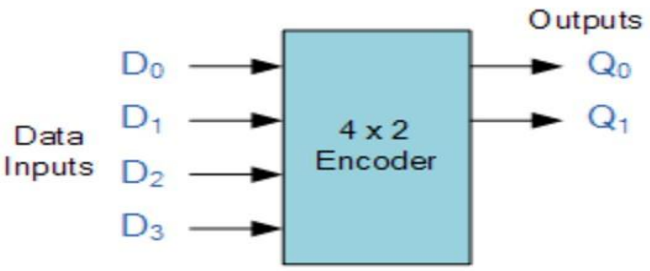
Based on the simplified Boolean functions for the three outputs G, E and L, the logic diagram of the 2-bit magnitude comparator is shown below:


A
C'
B
D'
G=AB>CD
A
C
X1 =(A⊕C)'
E=(AB=CD)
B
D
X0 =(B⊕D)'
A'
C
B'
D
L=AB<CD

Encoder

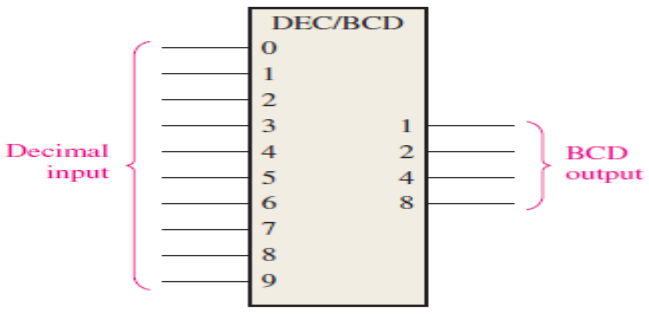
A Digital Encoder more commonly called a Binary Encoder takes ALL its data inputs one at a time and then converts them into a single encoded output.
a binary encoder, is a multi-input combinational logic circuit that converts the logic level "1" data at its inputs into an equivalent binary code at its output.
Generally, digital encoders produce outputs of 2-bit, 3-bit or 4-bit codes depending upon the number of data input lines. An "n-bit" binary encoder has 2^n input lines and n-bit output lines with common types that include 4-to-2, 8-to-3 and 16-to-4 line configurations. (fig) shows 4 to 2 encoder with its truth table

4 to 2 Encoder with its truth table



Inputs				Outputs	
D_3	D_2	D_1	D_0	Q_1	Q_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1
0	0	0	0	x	x

One of the main disadvantages of standard digital encoders is that they can generate the wrong output code when there is more than one input present at logic level "1". For example, if we make inputs D_1 and D_2 HIGH at logic "1" both at the same time, the resulting output is neither at "01" or at "10" but will be at "11" which is an output binary number that is different to the actual input present. Also, an output code of all logic "0"s can be generated when all of its inputs are at "0" OR when input D_0 is equal



Logic symbol for a decimal-to-BCD encoder.

The BCD (8421) code is listed in Table. From this table you can determine the relationship between each BCD bit and the decimal digits in order to analyze the logic. For instance, the most significant bit of the BCD code, *A*3, is always a 1 for decimal digit 8 or9. An OR expression for bit *A*3 in terms of the decimal digits can therefore be written as $A3 = 8 + 9$ Bit *A*2 is always a 1 for decimal digit 4, 5, 6 or 7 and can be expressed as an OR function as follows:

$A2 = 4 + 5 + 6 + 7$

Bit *A*1 is always a 1 for decimal digit 2, 3, 6, or 7 and can be expressed as

$A1 = 2 + 3 + 6 + 7$

*A*0 is always a 1 for decimal digit 1, 3, 5, 7, or 9. The expression for *A*0 is

$A0 = 1 + 3 + 5 + 7 + 9$

TABLE

Decimal Digit	BCD Code A_3	A_2	A_1	A_0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Now let's implement the logic circuitry required for encoding each decimal digit to a BCD code by using the logic expressions just developed. It is simply a matter of OR ing the appropriate decimal digit input lines to form each BCD output. The basic encoder logic resulting from these expressions is shown in Figure

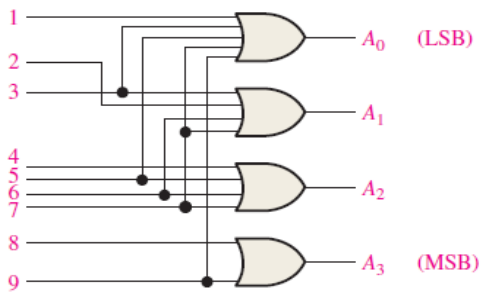
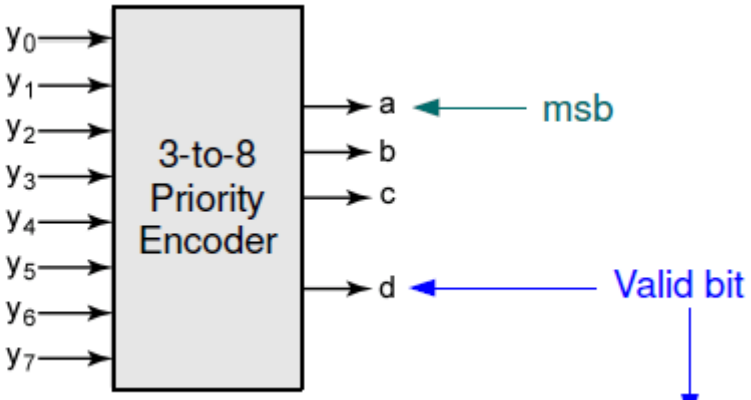


FIGURE Basic logic diagram of a decimal-to-BCD encoder. A 0-digit input is not needed because the BCD outputs are all LOW when there are no HIGH inputs.

Priority Encoders

- If more than one input is active, the higher-order input has priority over the lower-order input.
  - The higher value is encoded on the output
- A valid indicator, d, is included to indicate whether or not the output is valid.
  - Output is invalid when no inputs are active
- d = 0
  - Output is valid when at least one input is active
- d = 1

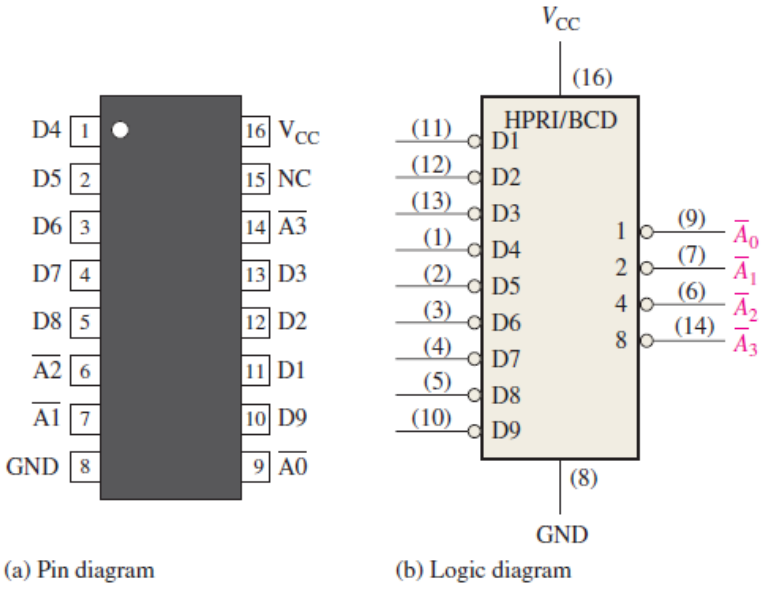
Why is the valid indicator needed?



y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	a	b	c	d
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
X	1	0	0	0	0	0	0	0	0	1	1
X	X	1	0	0	0	0	0	0	1	0	1
X	X	X	1	0	0	0	0	0	1	1	1
X	X	X	X	1	0	0	0	1	0	0	1
X	X	X	X	X	1	0	0	1	0	1	1
X	X	X	X	X	X	1	0	1	1	0	1
X	X	X	X	X	X	X	1	1	1	1	1

Fixed-Function Device The 74HC147 is a priority encoder with active-LOW inputs (0)
for decimal digits 1 through 9 and active-LOW BCD outputs as indicated in the logic symbol
in Figure. A BCD zero output is represented when none of the inputs is active.

The device pin numbers are in parentheses.

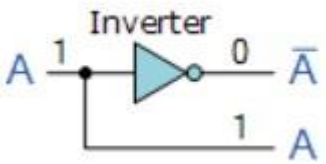


(a) Pin diagram (b) Logic diagram

FIGURE The 74HC147 decimal-to-BCD encoder (HPRI means highest value input has priority.

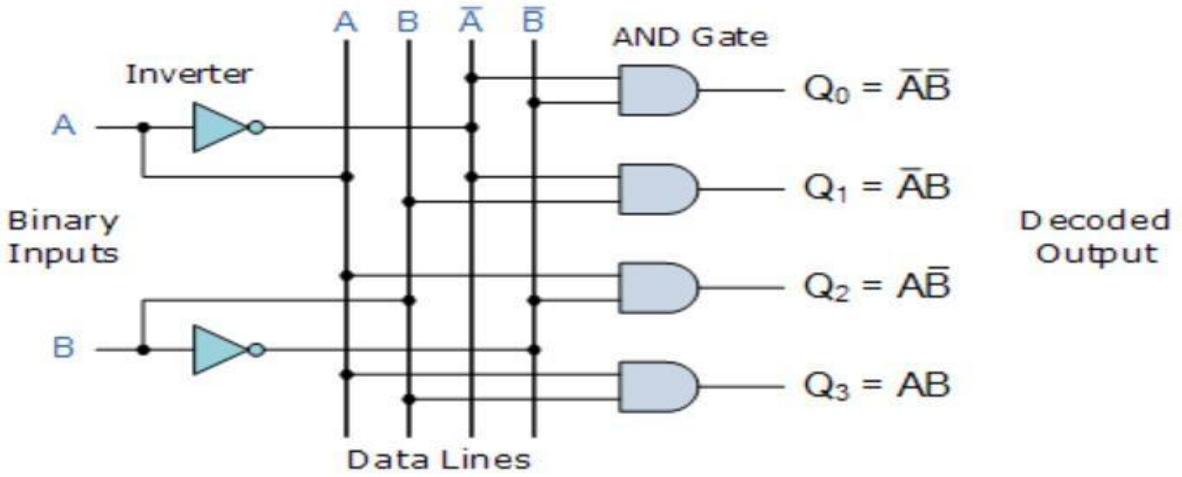
Decoders

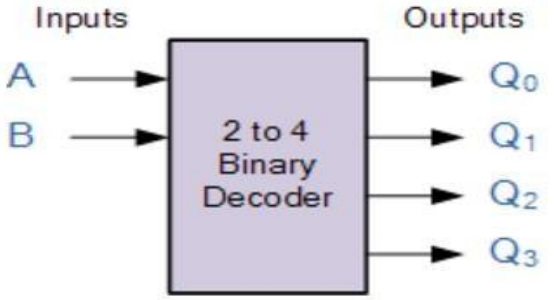
Decoders are another type of Digital Logic device that has inputs of 2-bit, 3-bit or 4-bit codes depending upon the number of data input lines, so a decoder that has a set of two or more bits will be defined as having an n- bit code and therefore it will be possible to represent 2^n possible values So for example, an inverter NOT-gate can be classed as a 1-to-2 binary decoder as 1-input and 2-outputs (2^1) is possible because with an input A it can produce two outputs A and A'(not-A) as shown in below .



A standard combinational logic decoder is an **n-to-m** decoder, where $m \leq 2^n$, and whose output, Q is dependent only on its present input states. In other words, a binary decoder looks at its current inputs, determines which binary code or binary number is present at its inputs and selects the appropriate output that corresponds to that binary input. Commonly available BCD-to-Decimal decoders include the TTL 7442 or the CMOS 4028 Generally a decoders output code normally has more bits than its input code and practical "binary decoder" circuits include, 2-to-4 3-to-8 and 4- to-16 line configurations

An **Example** of a 2-to-4 line decoder along with its truth table is given below





Truth Table

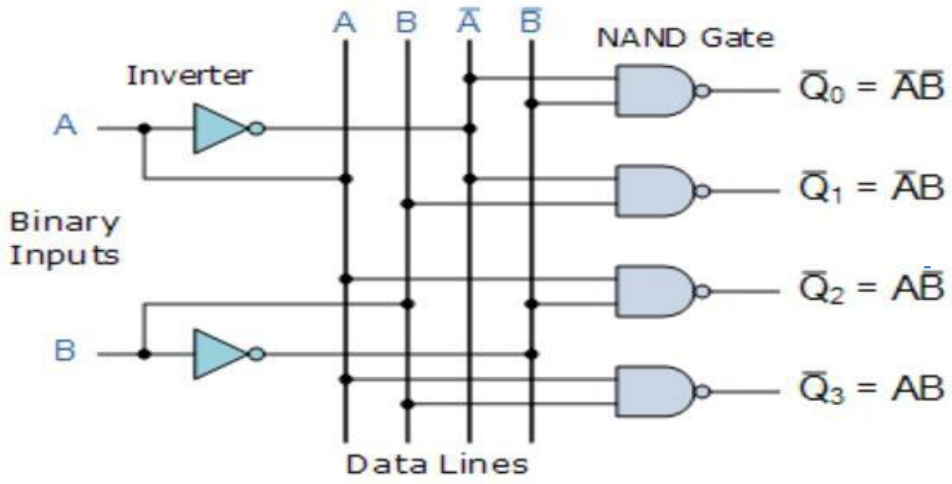
A	B	Q_0	Q_1	Q_2	Q_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

In this simple example a 2-to-4 line binary decoder consists of an array of four AND gates. The 2 binary inputs labeled A and B are decoded into one of 4 outputs, hence the description of 2-to-4 binary decoder.

The binary inputs A and B determine which output line from Q0 to Q3 is “HIGH” at logic level “1” while the remaining outputs are held “LOW” at logic “0” so only one output can be active (HIGH) at any one time. Therefore, whichever output line is “HIGH” identifies the binary code present at the input, in other words it “de-codes” the binary input.
An alternative way of looking at the decoder circuit is to regard inputs A, B and C as address signals. Each combination of A, B or C defines a unique memory address. some binary decoders are constructed using NAND gates instead of AND gates for their decoded output, since NAND gates are cheaper to produce than AND’s as they require fewer transistors to implement within their design. The use of NAND gates as the decoding element, results in an active-“LOW” output while the rest will be “HIGH”. As a NAND gate produces the AND operation with an inverted output, the NAND decoder looks like as in (fig) with its inverted T.T

Electronic Systems Simulators

محاكيات الانظمة الالكترونية BY –KERIM .D.A



Truth Table

A	B	Q_0	Q_1	Q_2	Q_3
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0