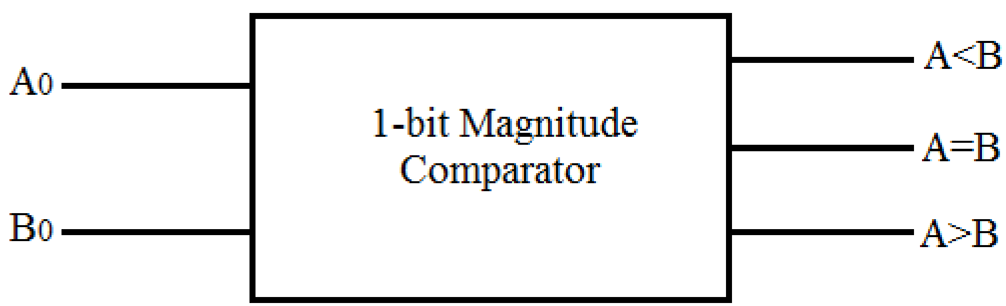
Magnitude Comparator

- It is a combinational circuit that compares two numbers and determines their relative magnitude.
- Consider the two numbers as A & B.
- Hence, it consists of three outputs that indicates:
  - A > B
  - A = B
  - A< B
- The circuit for comparing two *n -bit numbers has* 2^{2n} *entries in the* truth table and becomes too cumbersome.
- A comparator circuit possesses a certain amount of regularity.
- Digital functions that possess an inherent well-defined regularity can usually be designed by means of an algorithm
  - a procedure which specifies a finite set of steps that, if followed, give the solution to a problem.

Magnitude Comparator

- Consider two numbers, $A(A_0)$ and $B(B_0)$ with a single bit each.

Block Diagram:



Truth Table:

A_0	B_0	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

Functions:

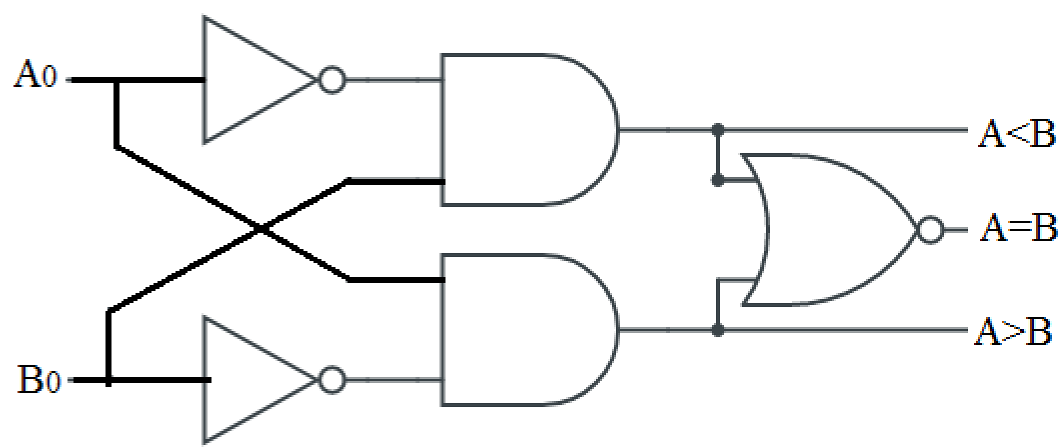
$(A<B)= A_0'B_0$

$(A>B)=A_0B_0'$

$(A=B)=A_0'B_0'+A_0B_0$

$=(A_0'B_0+A_0B_0')'$

Circuit Diagram:



Magnitude Comparator

- Consider two numbers, *A and B , with four digits* each(Write the coefficients of the numbers in descending order of significance)

$$A = A_3\, A_2\, A_1\, A_0$$
$$B = B_3 B_2 B_1 B_0$$

- A=B
  - The two numbers are equal if all pairs of significant digits are equal:
    - $A_3 = B_3$, $A_2 = B_2$, $A_1 = B_1$, *and* $A_0 = B_0$.
  - *When the numbers are binary, the digits are either 1 or 0, and the equality of* each pair of bits can be expressed logically with an exclusive-NOR function as:

$$x_i = A_i B_i + A_i' B_i' \quad \text{for } i = 0, 1, 2, 3$$

  - For equality to exist, all x_i variables must be equal to 1, a condition that dictates an AND *operation of all variables:*

$$(A = B) = x_3 x_2 x_1 x_0$$

- The binary variable (A = B) is equal to 1 only if all pairs of digits of the two numbers are equal

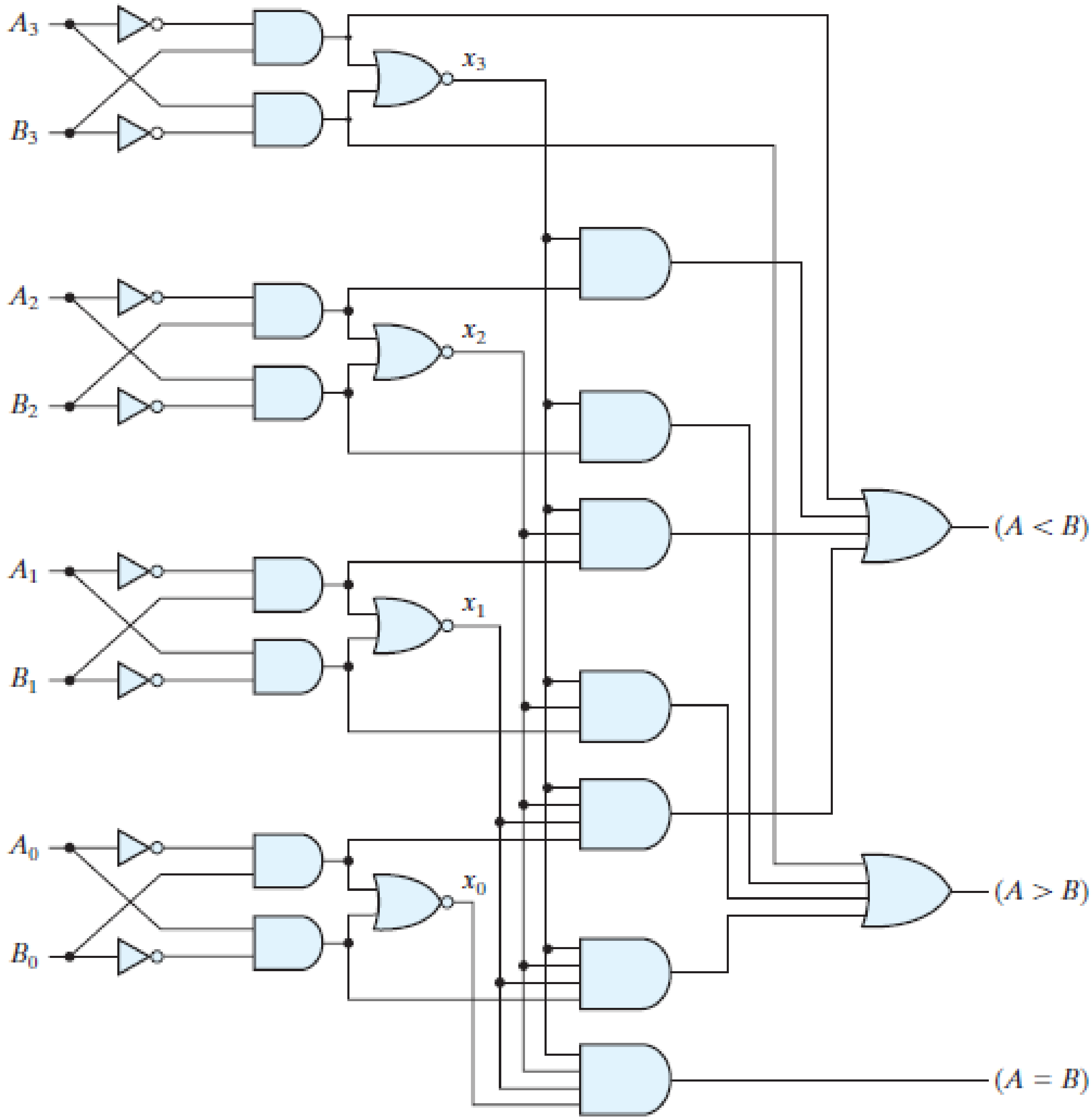
Magnitude Comparator

- <u>A>B and A<B</u>
  - we inspect the relative magnitudes of pairs of significant digits, starting from the most significant position.
  - If the two digits of a pair are equal, we compare the next lower significant pair of digits.
  - The comparison continues until a pair of unequal digits is reached.
    - If the corresponding digit of A is 1 and that of B is 0, we conclude that A>B.
    - If the corresponding digit of A is 0 and that of B is 1, we have A<B.
  - The sequential comparison can be expressed logically by the two Boolean fun

$$(A > B) = A_3B'_3 + x_3A_2B'_2 + x_3x_2A_1B'_1 + x_3x_2x_1A_0B'_0$$

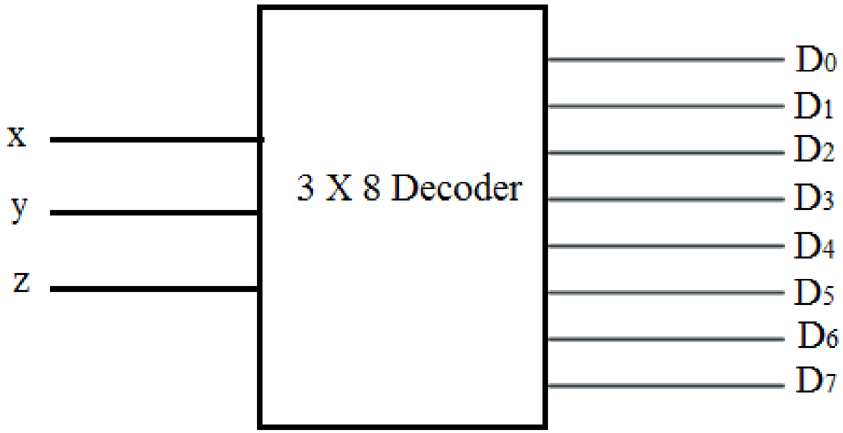
$$(A < B) = A'_3B_3 + x_3A'_2B_2 + x_3x_2A'_1B'_1 + x_3x_2x_1A'n_0B'_0$$

- The symbols (A>B) and (A<B) are binary output variables that are equal to 1 when A>B and A<B, respectively.


A3
B3
x3
A2
B2
x2
A1
B1
x1
A0
B0
x0
(A < B)
(A > B)
(A = B)

Decoder

- A decoder is a combinational circuit that converts binary information from n input lines to the maximum 2^n unique outputs
- If the decoder consists of three inputs the maximum number of output lines is eight.
- <u>Block Diagram:</u>

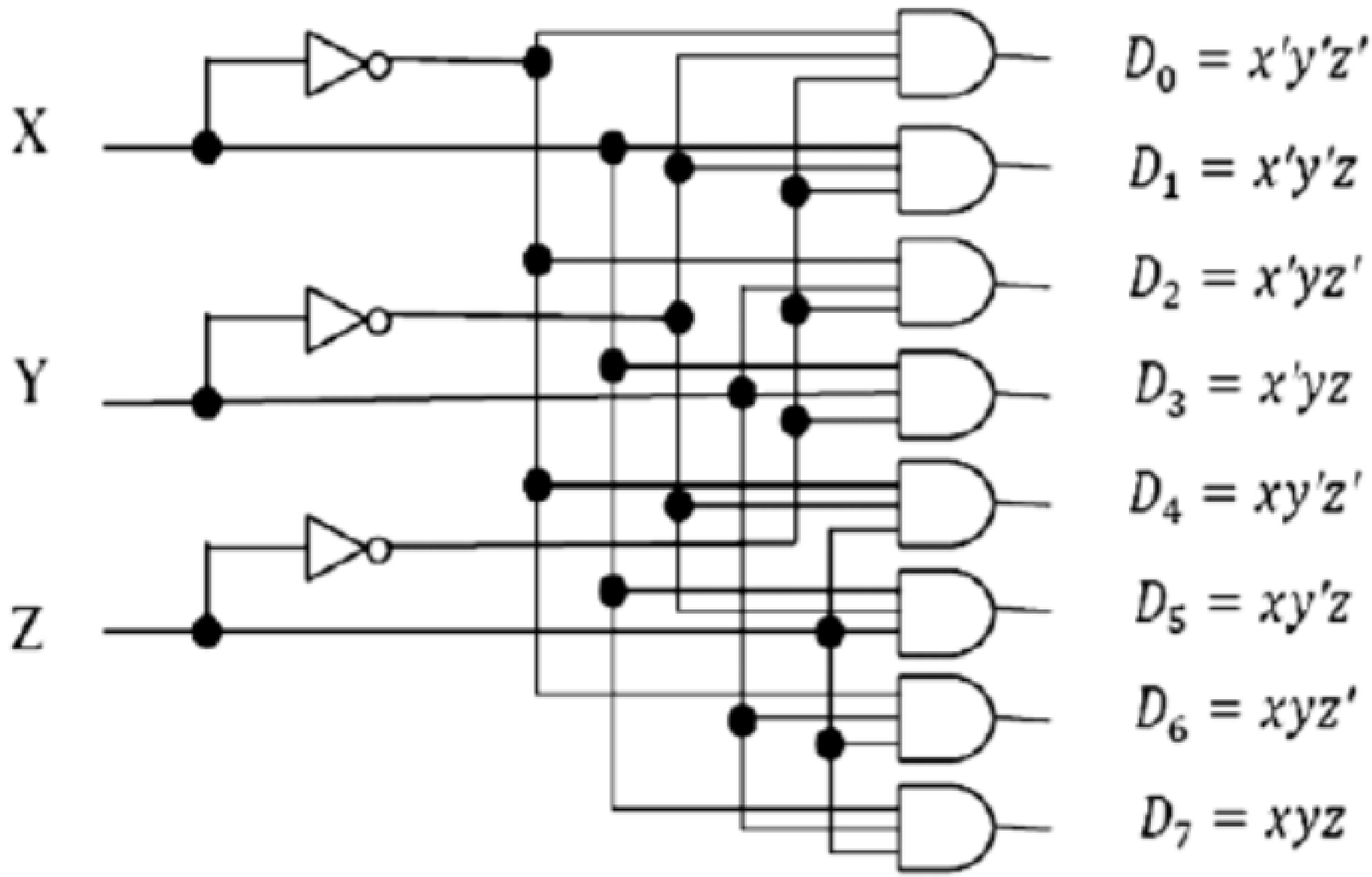


- <u>Truth Table:</u>

x	y	z	D0	D1	D2	D3	D4	D5	D6	D7
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

Decoder

- <u>Circuit Diagram:</u>



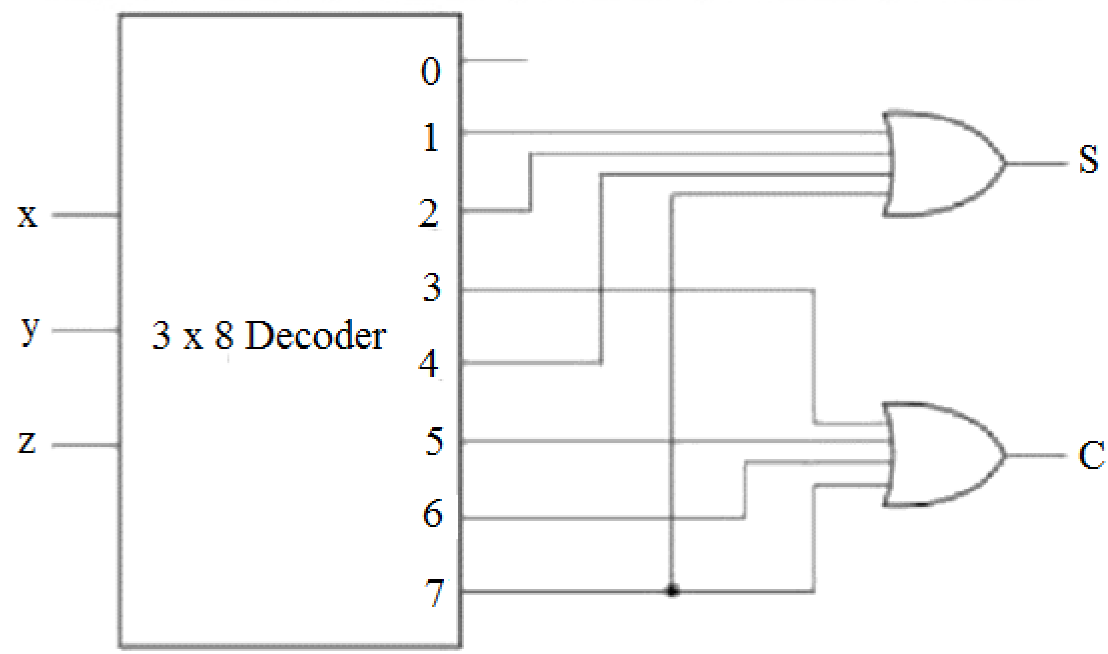
Decoder

- Any combinational circuit with n inputs and m outputs can be implemented with an n-to-2^n line decoder and m OR gates.
    - For this the Boolean functions for the circuit must be expressed in sum of minterms form
    - Decoder is chosen that generates all the minterms of the n input variables
    - The inputs to each OR gate are selected from the decoder outputs according to the minterm list in each function.

Decoder

- Implement a full adder circuit with a decoder.
  - Truth table:

x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



  - Minterms form:
    - $S(x,y,z) = \sum(1,2,4,7)$
    - $C(x,y,z) = \sum(3,5,6,7)$

Active-Low Decoder

x	y	z	D0	D1	D2	D3	D4	D5	D6	D7
0	0	0	0	1	1	1	1	1	1	1
0	0	1	1	0	1	1	1	1	1	1
0	1	0	1	1	0	1	1	1	1	1
0	1	1	1	1	1	0	1	1	1	1
1	0	0	1	1	1	1	0	1	1	1
1	0	1	1	1	1	1	1	0	1	1
1	1	0	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	0