DIGITAL LOGIC

LECTURE-21

Decimal Adder

- An adder for a computer that employ arithmetic circuits that accept coded decimal numbers and present results in the same code.

- A decimal adder requires a minimum of nine inputs and five outputs, since four bits are required to code each decimal digit and the circuit must have an input and output carry.

- There is a wide variety of possible decimal adder circuits, depending upon the code used to represent the decimal digits.

- Here we examine a decimal adder for the BCD code.

BCD Adder

- Consider the arithmetic addition of two decimal digits in BCD, together with an input carry from a previous stage.

- Since each input digit does not exceed 9, the output sum cannot be greater than $9 + 9 + 1 = 19$, the 1 in the sum being an input carry.

- Suppose we apply two BCD digits to a four-bit binary adder.

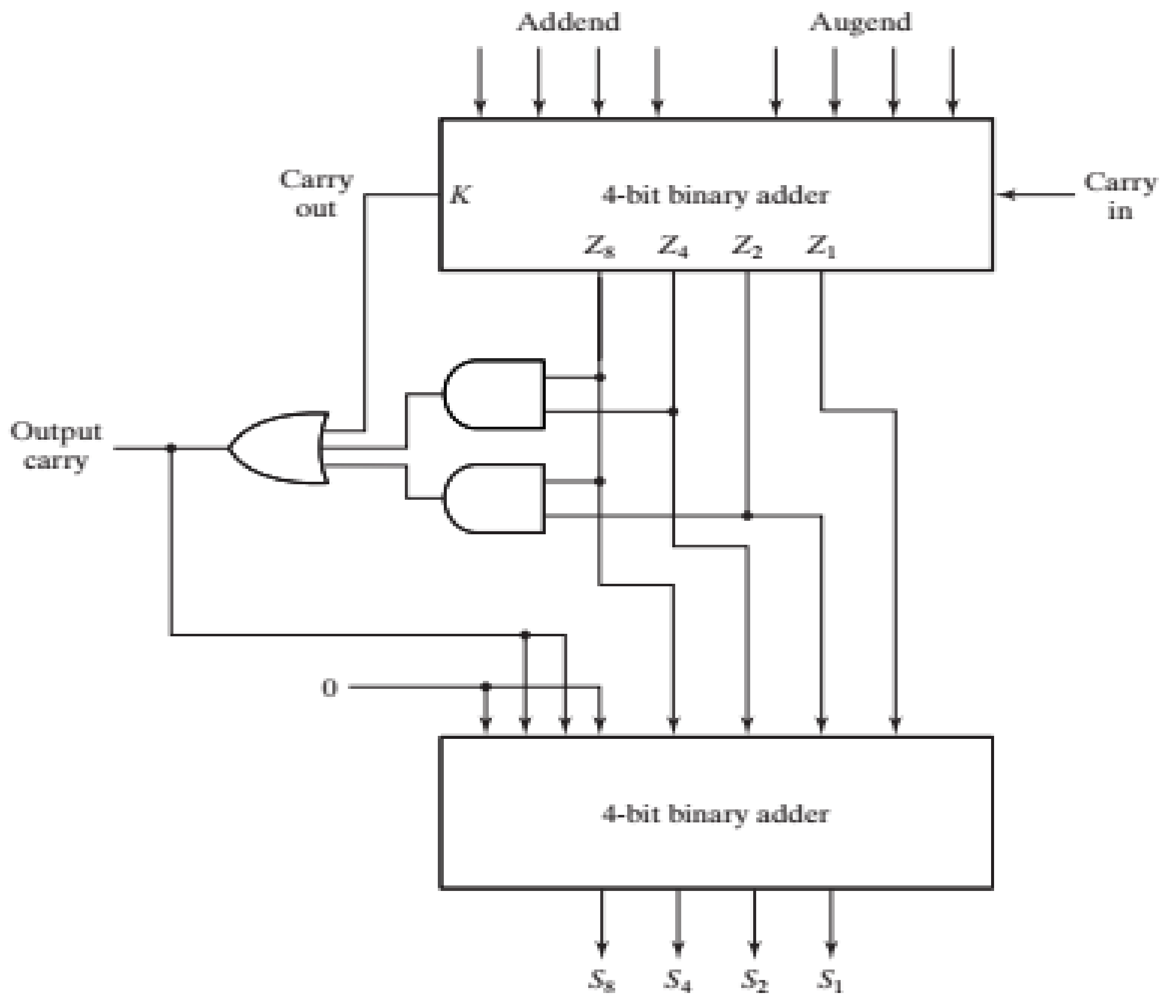
Derivation of BCD Adder

Binary Sum					BCD Sum					Decimal
K	Z_8	Z_4	Z_2	Z_1	C	S_8	S_4	S_2	S_1	
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	2
0	0	0	1	1	0	0	0	1	1	3
0	0	1	0	0	0	0	1	0	0	4
0	0	1	0	1	0	0	1	0	1	5
0	0	1	1	0	0	0	1	1	0	6
0	0	1	1	1	0	0	1	1	1	7
0	1	0	0	0	0	1	0	0	0	8
0	1	0	0	1	0	1	0	0	1	9
0	1	0	1	0	1	0	0	0	0	10
0	1	0	1	1	1	0	0	0	1	11
0	1	1	0	0	1	0	0	1	0	12
0	1	1	0	1	1	0	0	1	1	13
0	1	1	1	0	1	0	1	0	0	14
0	1	1	1	1	1	0	1	0	1	15
1	0	0	0	0	1	0	1	1	0	16
1	0	0	0	1	1	0	1	1	1	17
1	0	0	1	0	1	1	0	0	0	18
1	0	0	1	1	1	1	0	0	1	19

- The logic circuit that detects the necessary correction can be derived from the entries in the table. It is obvious that a correction is needed when the binary sum has an output carry $K = 1$.

- The condition for a correction and an output carry can be expressed by the Boolean function

$$C = K + Z_8Z_4 + Z_8Z_2$$



Block diagram of a BCD adder

Magnitude Comparator

- A *magnitude comparator* *is a* combinational circuit that compares two numbers A *and* B *and determines their relative* magnitudes.

- The outcome of the comparison is specified by three binary variables that indicate whether $A > B$, $A = B$ *or* $A < B$.

- Consider two numbers, A *and* B , *with four digits* each. Write the coefficients of the numbers in descending order of significance: $A = A_3\, A_2\, A_1\, A_0$

$$B = B_3 B_2 B_1 B_0$$

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 $A_0 = B_0$

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

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

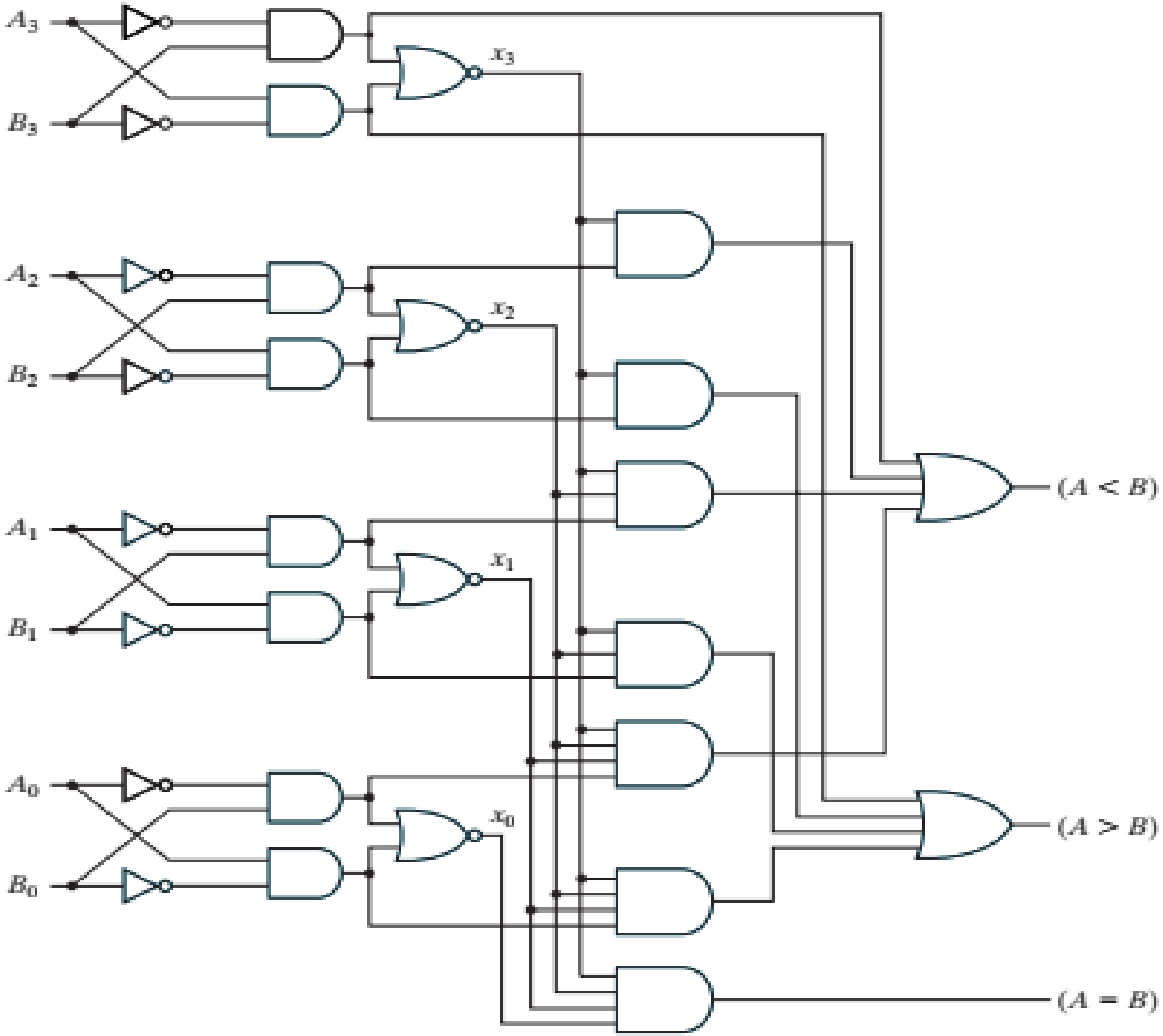
$$(A = B) = x_3 x_2 x_1 x_0$$

To determine whether A is greater or less than B , we inspect the relative magnitudes of pairs of significant digits, starting from the most significant position.

$$(A > B) = A_3B_3' + x_3A_2B_2' + x_3x_2A_1B_1' + x_3x_2x_1A_0B_0'$$

$$(A < B) = A_3'B_3 + x_3A_2'B_2 + x_3x_2A_1'B_1' + x_3x_2x_1A'n_0B_0'$$

The symbols *$A > B$ and $A < B$ are binary output variables that are equal to 1* when *$A > B$ and $A < B$, respectively.*



Four-bit magnitude comparator

Example

Design a combinational circuit that compares two 4-bit numbers to check if they are equal. The circuit output is equal to 1 if the two numbers are equal and 0 otherwise.

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

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

HDL for 4 bits Magnitude Comparator

```
// Dataflow description of a four-bit comparator
module  mag_compare
(  output    A_lt_B, A_eq_B, A_gt_B,
 input  [3: 0]   A, B);
 assign  A_lt_B  = (A <  B);
 assign  A_gt_B  = (A > B);
assign  A_eq_B  = (A = =  B);
endmodule
```

THANK YOU