The material covered in this class will be as follows:

⇒Overflow.

⇒BCD adder.

⇒Binary multiplier.

⇒Magnitude comparator.

Overflow

Overflow is defined as the situation when two N-digit numbers are added and the sum occupies (N+1) digits. This situation occurs when adding binary numbers as follows:

1. An end carry is generated when adding two N-bit unsigned numbers.
2. The carry-in and carry-out bits are different when adding two N-bit signed binary numbers.

Example: Suppose we are adding +70 and +80 using 8-bits in signed 2's complement form.

+70 → 01000110 and +80 → 01010000

The carry 0 1

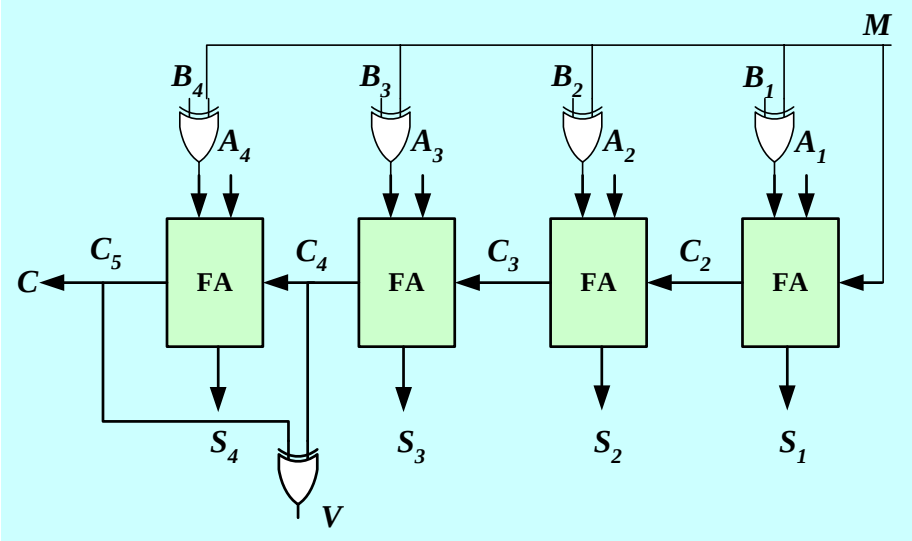
+70	0 1 0 0 0 1 1 0
+80	0 1 0 1 0 0 0 0
+150	1 0 0 1 0 1 1 0

The same overflow occurs when adding (-70) + (-80)

The carry 1 0

-70	1 0 1 1 1 0 1 0
-80	1 0 1 1 0 0 0 0
-150	0 1 1 0 1 0 1 0

An overflow can be detected by observing the carry-in and the carry-out of the sign bit. If we apply them to an exclusive-or gate then the output is one when overflow occurs.

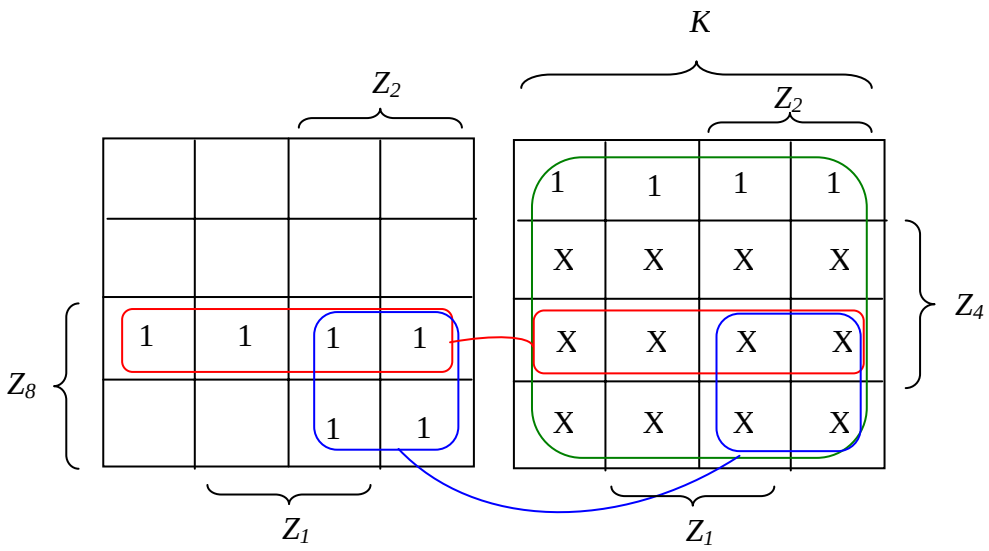


Decimal Adder

This is also called a BCD adder. In order to add two decimal digits with a possible carry in of one, then the maximum sum is 19. The following table shows the sum when performed in binary and compared to the sum when performed in BCD. In both cases five outputs are needed.

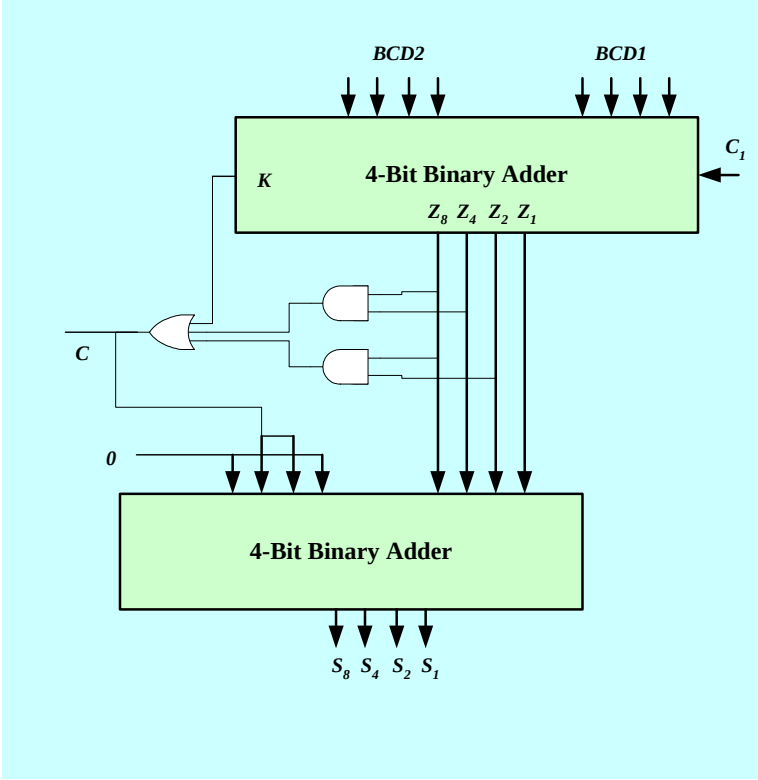
Dec.	Binary sum					BCD sum				
	K	**Z_8**	**Z_4**	**Z_2**	**Z_1**	**C**	**S_8**	**S_4**	**S_2**	**S_1**
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	1
2	0	0	0	1	0	0	0	0	1	0
3	0	0	0	1	1	0	0	0	1	1
4	0	0	1	0	0	0	0	1	0	0
5	0	0	1	0	1	0	0	1	0	1
6	0	0	1	1	0	0	0	1	1	0
7	0	0	1	1	1	0	0	1	1	1
8	0	1	0	0	0	0	1	0	0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

Inspecting the table reveals that a correction in the sum is needed when the sum is greater than 9. The correction is adding 6 to the sum. The BCD adder will then consist of the 4-bit binary adder. A second 4-bit binary adder is needed to add 6 to the sum when it is greater than 9. The required logic circuit needed to detect if correction is needed can be obtained by inspecting the table. A second method is to find a simplified expression for the carry out C of the five variables K, Z8, Z4, Z2, and Z1. Minterms m20 to m31 are considered don't care conditions.



$$\therefore C = K + Z_8 Z_4 + Z_8 Z_2 .$$

The logic circuit will be as follows:



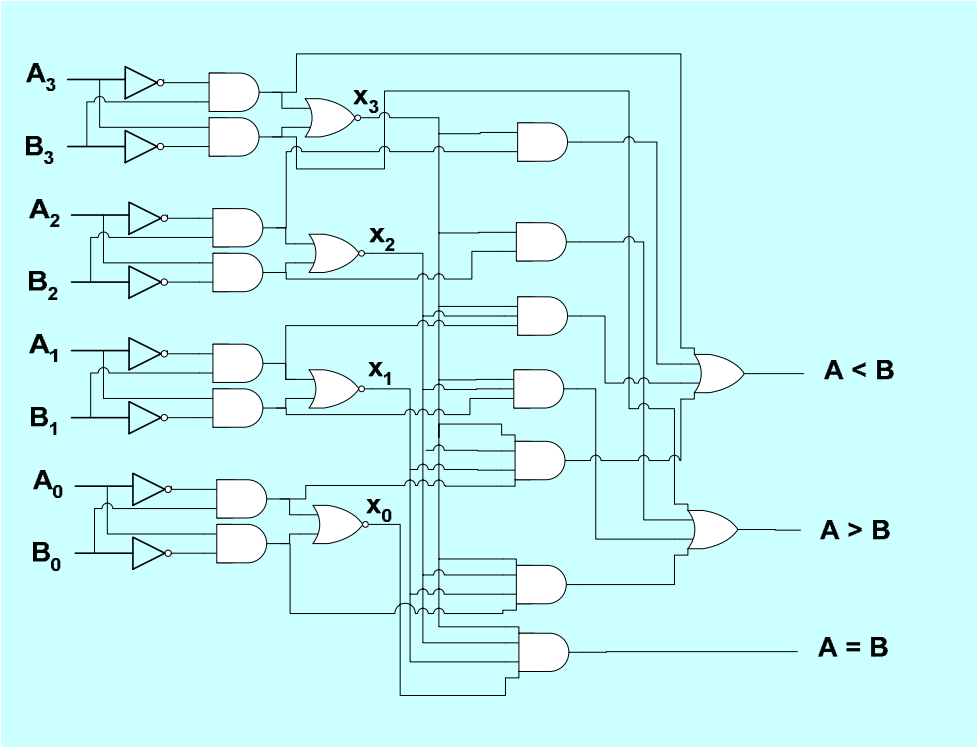
Magnitude Comparator

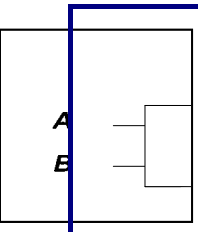
To compare two 4-bit numbers ($A_3\ A_2\ A_1\ A_0$) and ($B_3\ B_2\ B_1\ B_0$), we have to design a circuit with eight inputs and three outputs. The outputs are:

- A = B
- A > B
- A < B

A better method to design this circuit is to follow the systematic way of comparison, where we compare each pair of bits starting from the most significant bit. If all pairs are equal then A=B. If we find a difference in the compared bits (i.e. one is 1 and the other is 0), then the number containing the 1 is larger. This leads to the following three Boolean functions

1. $(A=B) = x_3 x_2 x_1 x_0$, where $x_i = A_i B_i + A_i' B_i'$
2. $(A > B) = A_3 B_3' + x_3 A_2 B_2' + x_3 x_2 A_1 B_1' + x_3 x_2 x_1 A_0 B_0'$
3. $(A < B) = A'_3 B_3 + x_3 A_2' B_2 + x_3 x_2 A_1' B_1 + x_3 x_2 x_1 A_0' B_0$





Binary Multiplier

To multiply two 2-bit binary numbers B1 B0 and A1 A0, we may use half adders and AND gates.

$$
\begin{array}{cccc}
 & & B_1 & B_0 \\
 & & A_1 & A_0 \\
\hline
 & & A_0B_1 & A_0B_0 \\
 & A_1B_1 & A_1B_0 & \\
\hline
C_3 & C_2 & C_1 & C_0
\end{array}
$$