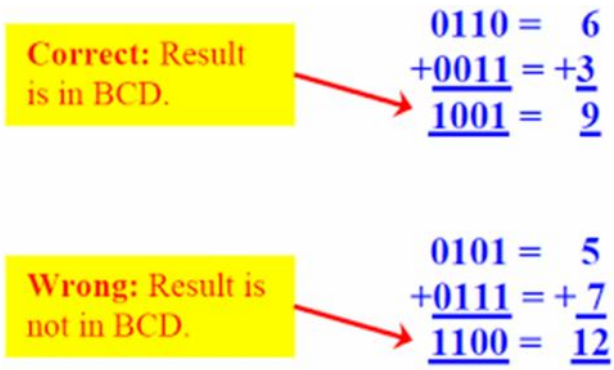
BCD Adder

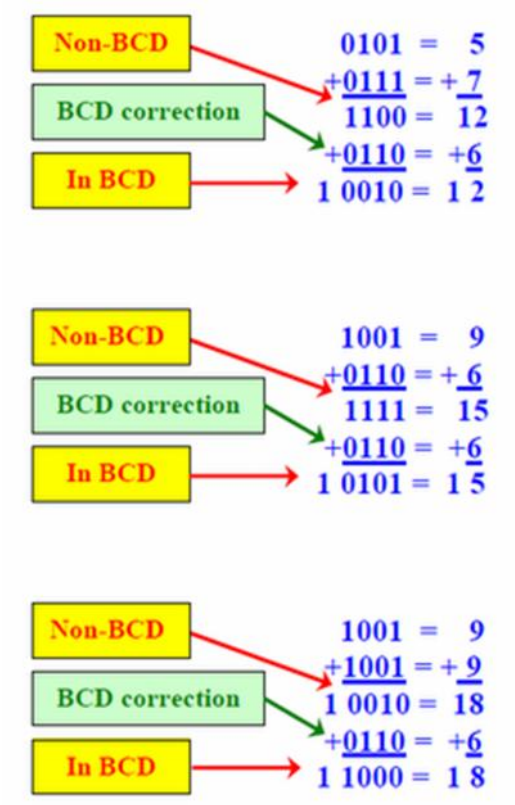
If two BCD digits are added then their sum result will not always be in BCD.
Consider the two given examples.



In the first example, result is in BCD while in the second example it is not in BCD.
Four bits are needed to represent all BCD digits (0 – 9). But with four bits we can represent up to 16 values (0000 through 1111). The extra six values (1010 through 1111) are not valid BCD digits. Whenever the sum result is > 9, it will not be in BCD and will require correction to get a valid BCD result.

Z_3	Z_2	Z_1	Z_0		F
0	0	0	0		0
0	0	0	1		0
0	0	1	0		0
0	0	1	1		0
0	1	0	0		0
0	1	0	1		0
0	1	1	0		0
0	1	1	1		0
1	0	0	0		0
1	0	0	1		0
1	0	1	0		1
1	0	1	1		1
1	1	0	0		1
1	1	0	1		1
1	1	1	0		1
1	1	1	1		1

Correction is done through the addition of 6 to the result to skip the six invalid values as shown in the truth table by yellow color. Consider the given examples of non-BCD sum result and its correction.



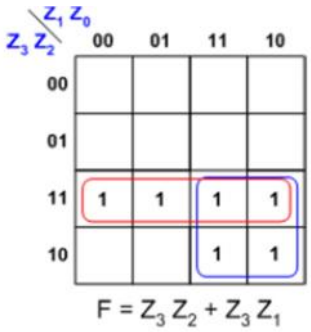
A BCD adder is a circuit that adds two BCD digits in parallel and produces a sum BCD digit and a carry out bit.

The maximum sum result of a BCD input adder can be 19. As maximum number in BCD is 9 and may be there will be a carry from previous stage also, so 9 + 9 + 1 = 19

The following truth table shows all the possible sum results when two BCD digits are added.

Dec	CO	Z_3	Z_2	Z_1	Z_0		F
0	0	0	0	0	0		0
1	0	0	0	0	1		0
2	0	0	0	1	0		0
3	0	0	0	1	1		0
4	0	0	1	0	0		0
5	0	0	1	0	1		0
6	0	0	1	1	0		0
7	0	0	1	1	1		0
8	0	1	0	0	0		0
9	0	1	0	0	1		0
10	0	1	0	1	0		1
11	0	1	0	1	1		1
12	0	1	1	0	0		1
13	0	1	1	0	1		1
14	0	1	1	1	0		1
15	0	1	1	1	1		1
16	1	0	0	0	0		1
17	1	0	0	0	1		1
18	1	0	0	1	0		1
19	1	0	0	1	1		1

The logic circuit that checks the necessary BCD correction can be derived by detecting the condition where the resulting binary sum is 01010 through 10011 (decimal 10 through 19).
It can be done by considering the shown truth table, in which the function F is true when the digit is not a valid BCD digit. It can be simplified using a 5-variable K-map.
But detecting values 1010 through 1111 (decimal 10 through 15) can also be done by using a 4-variable K-map as shown in the figure.

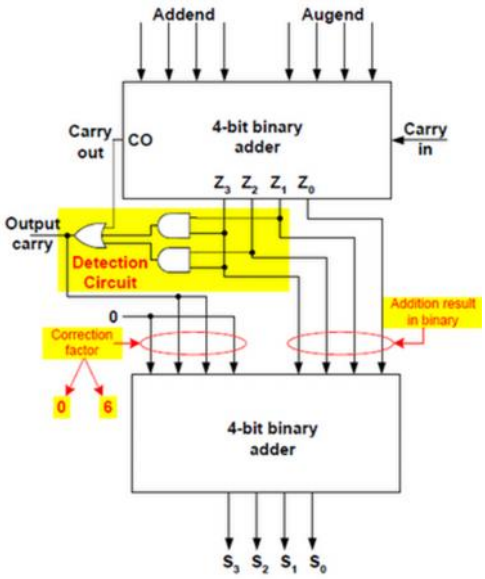


Values greater than 1111, i.e., from 10000 through 10011 (decimal 16 through 19) can be detected by the carry out (CO) which equals 1 only for these output values. So, F = CO = 1 for these values.
Hence, F is true when CO is true OR when ($Z_3 Z_2 + Z_3 Z_1$) is true.

Thus, the correction step (adding 0110) is performed if the following function equals 1:

$$F = CO + Z_3 Z_2 + Z_3 Z_1$$

The circuit of the BCD adder will be as shown in the figure.



The two BCD digits, together with the input carry, are first added in the top 4-bit binary adder to produce the binary sum. The bottom 4-bit binary adder is used to add the correction factor to the binary result of the top binary adder.

Note:

1. When the Output carry is equal to zero, the correction factor equals zero.
2. When the Output carry is equal to one, the correction factor is 0110.

The output carry generated from the bottom binary adder is ignored, since it supplies information already available at the output-carry terminal.

A decimal parallel adder that adds n decimal digits needs n BCD adder stages. The output carry from one stage must be connected to the input carry of the next higher-order stage.