Practical No 5
Implement Adder and Subtractor Arithmetic circuits

Aim:

a) Design and implement Half adder and Full adder.
b) Design and implement binary subtractor.
c) Design and implement BCD adder.
d) Design and implement BCD subtractor.
e) Design and implement XS – 3 adder.
f) Design and implement XS – 3 subtractor.

Theory:

a) **Design and implement Half adder and Full adder.**

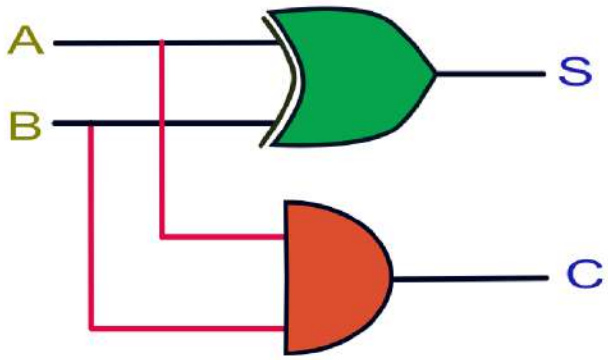
Half-Adder:
A half-adder is an arithmetic circuit block that can be used to add two one-bit numbers. Such a circuit thus has two inputs and two outputs (sum and carry).
The following is the truth table representing the working of a half-adder

Inputs		Outputs	
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

From the above truth table we write the Boolean expression for sum and carry as follows

$$S = A \cdot \overline{B} + \overline{A} \cdot B = A \oplus B$$
$$C = A \cdot B$$

Hence to implement the Sum (S) we need an EXOR gate and for Carry (C) we need a AND gate



The above circuit can be used to design and verify the working of half-adder

Full Adder:

A full-adder is a combinational circuit which has 3-inputs and 2-outputs. It adds 3-bits at a time A, B and C_{in}, where C_{in} is a carry generated by previous addition. The outputs are Sum(S) and Carry (CY)

The following is the truth table of a full-adder

Inputs			Outputs	
A	B	C_{in}	S	CY
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

In order to implement the full-adder, we need to design the circuit using K-maps, we draw the K-maps separately for Sum(S) and Carry(CY) and then solve the K-maps to get the required output

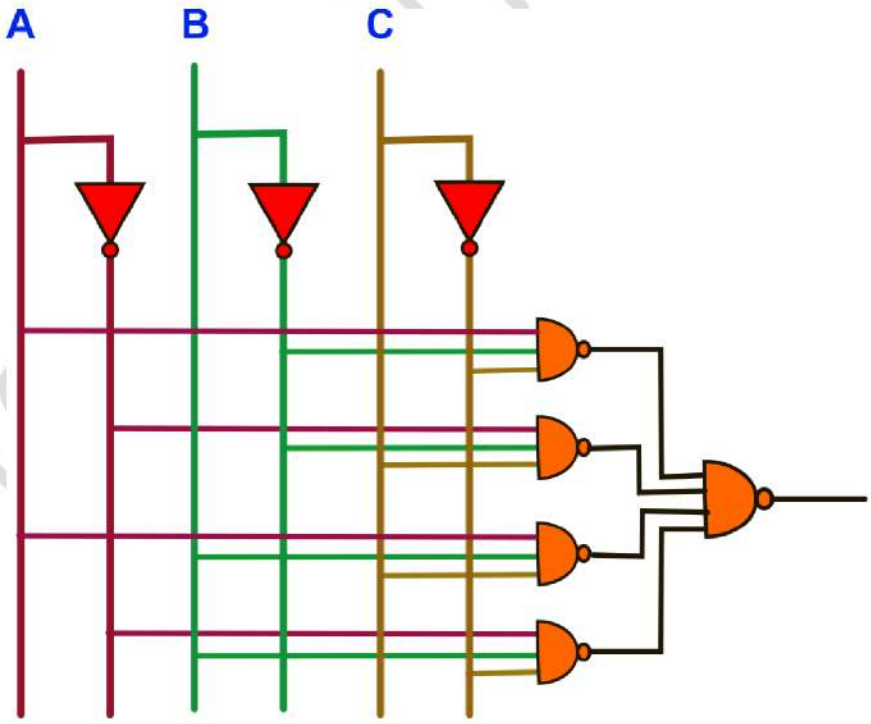
K – Map for Sum(S):

	$\overline{A}.\overline{B}$	$\overline{A}.B$	$A.B$	$A.\overline{B}$
$\overline{C}$	0	1	0	1
C	1	0	1	0

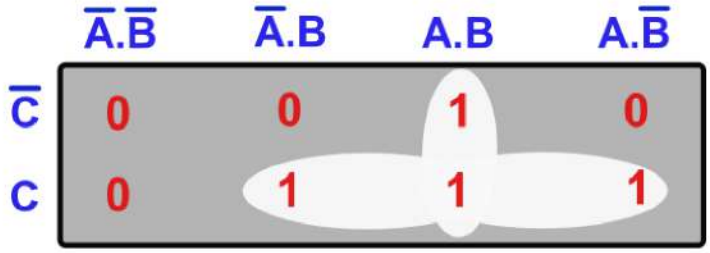
Hence the equation for the Sum from the K-map is

$$S = A.\overline{B}.\overline{C} + \overline{A}.\overline{B}.C + A.B.C + \overline{A}.B.\overline{C}$$

Therefore for realising the Sum circuit of the Full-adder we require the following



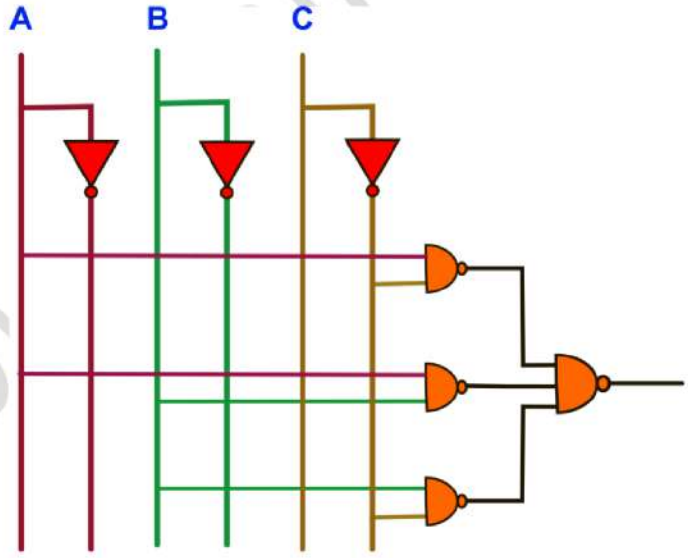
K – Map for Carry (CY):



From the above K-map we see that we get 3-pairs, and since a pair eliminates a variable hence equation for Carry is as follows

$$CY = A.B + B.C + A.C$$

Therefore for realizing the Carry circuit of the Full-adder we require the following



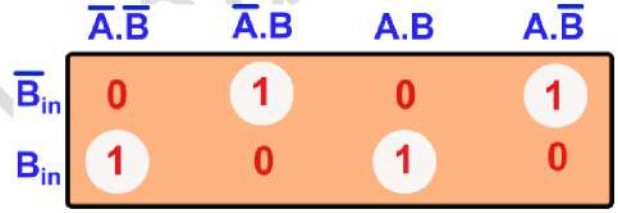
b) **Design and implement binary subtractor.**
A full-subtractor is a combinational circuit which has 3-inputs and 2-outputs. The third input is the borrow generated from previous subtraction.
The outputs are Difference (D) and Borrow out (B_O)

The following is the truth table of a full-subtractor

Inputs			Outputs	
A	**B**	**B_{in}**	**D**	**B_O**
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

In order to implement the full-subtractor, we need to design the circuit using K-maps, we draw the K-maps separately for Difference (D) and Borrow out(B_O) and then solve the K-maps to get the required output

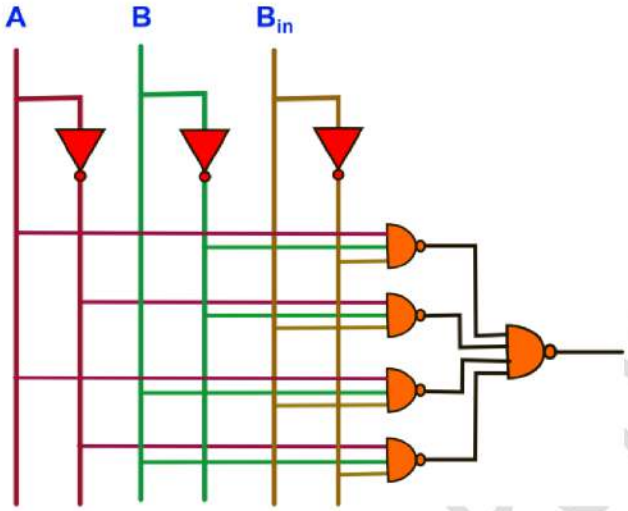
K – Map for Difference (D):



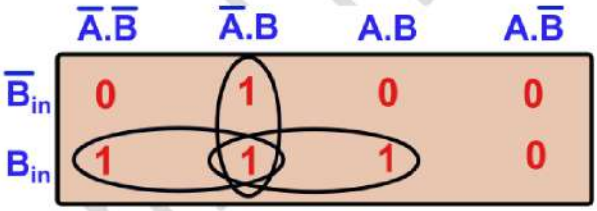
Hence the equation for the Difference from the K-map is

$$D = A.\overline{B}.\overline{B}_{in} + \overline{A}.\overline{B}.B_{in} + A.B.B_{in} + \overline{A}.B.\overline{B}_{in} \ .$$

Therefore for realizing the Difference circuit of the Full-subtractor we require the following



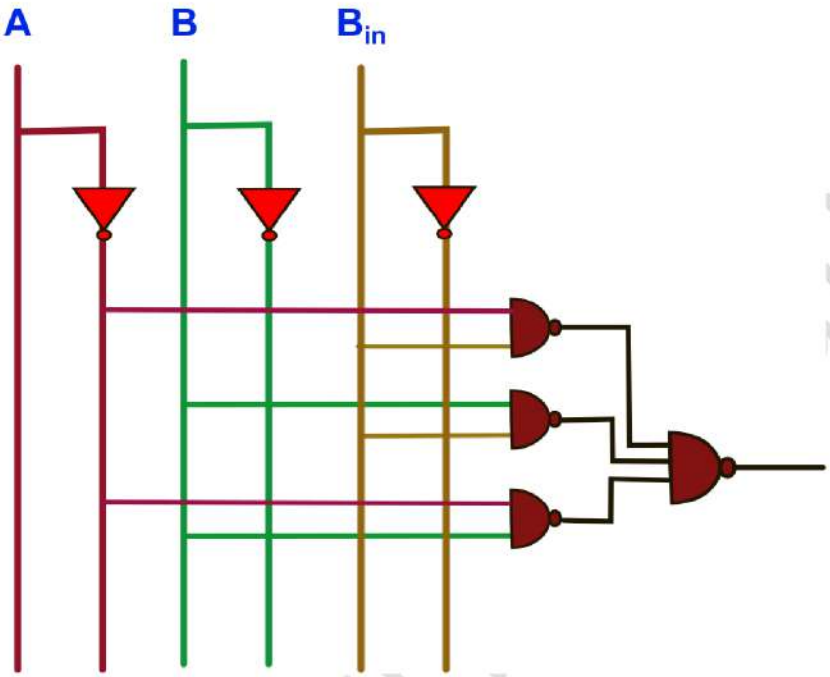
K – Map for Borrow out (B):



From the above K-map we see that we get 3-pairs, and since a pair eliminates a variable hence equation for Borrow out is as follows

$$B_o = \overline{A}.B_{in} + B.B_{in} + \overline{A}.B$$

Therefore for realizing the Borrow out circuit of the Full-subtractor we require the following



For video demonstration of the above practical part follow the link or scan the QR-code

https://youtu.be/seHYDjQRNds

c) **Design and implement BCD adder.**

A BCD (Binary-Coded Decimal) adder is a digital circuit that adds two BCD numbers. BCD is a way of representing each decimal digit (0-9) using its equivalent 4-bit binary value. In BCD addition, the result must also be in valid BCD form. This requires special handling since a simple binary addition of two BCD digits can produce a result that is not a valid BCD digit.

Rules for BCD addition:

i) If addition of two BCD numbers result in a sum less than 9 (1001) and no carry is generated, then the given result is the final result
For e.g

	3	0	0	1	1	
+	4	0	1	0	0	
	7	0	1	1	1	Valid BCD

ii) If addition of two BCD numbers result in a sum less than 9 (1001) and a carry is generated, then add 6(0110) to the sum for the final result
For e.g

		1				1	
	9	CY	1	0	0	1	
+	9		1	0	0	1	
	18	1	0	0	1	0	Valid BCD but incorrect result

Add 6 (0 1 1 0) to the sum we get

0	0	1	0
0	1	1	0
1	0	0	0

We substitute the corrected sum in the above and get the correct result

		1				1	
	9	CY	1	0	0	1	
+	9		1	0	0	1	
	18	1	1	0	0	0	Valid BCD and correct result

iii) If addition of two BCD numbers result in a sum greater than 9 (invalid BCD), then add 6(0110) to the result.

For e.g

			1	1	1		
	7	CY	0	1	1	1	
+	5		0	1	0	1	
	12	0	1	1	0	0	Invalid BCD and incorrect result

Add 6 (0 1 1 0) to the sum we get

		1			
CY	1	1	0	0	
	0	1	1	0	
1	0	0	1	0	

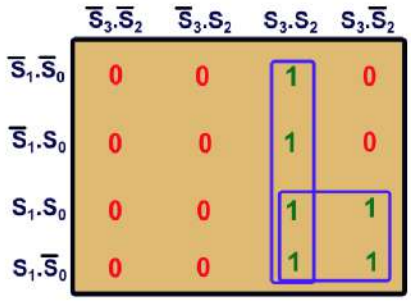
We substitute the corrected sum along with the carry and get the correct result

	7	CY	0	1	1	1	
+	5		0	1	0	1	
	12	1	0	0	1	0	Valid BCD correct result

Truth Table: For implementing the correction part, assuming the given Sums

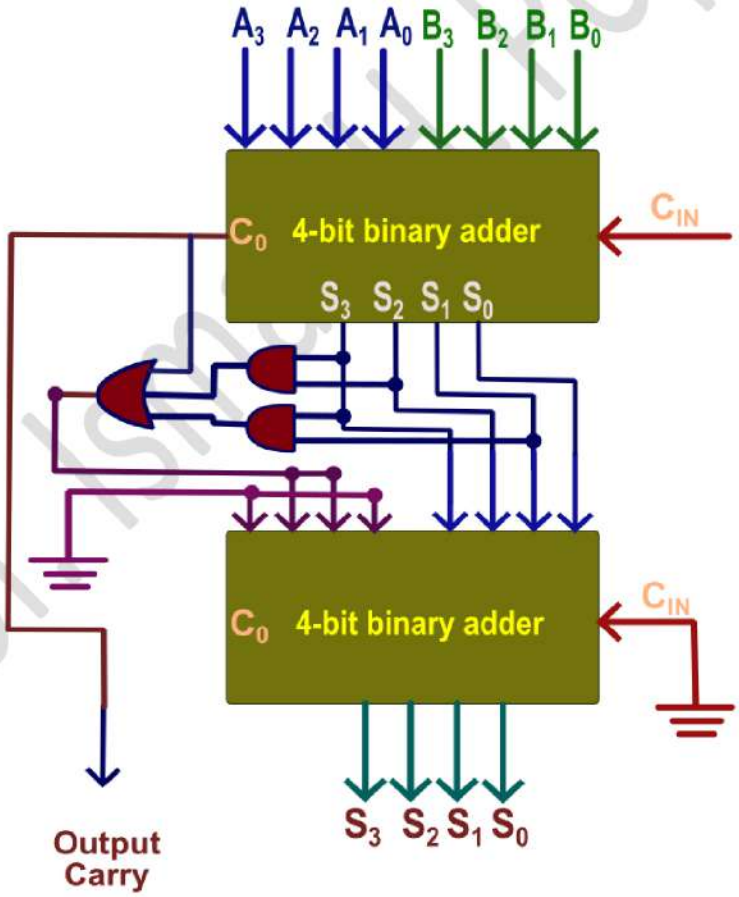
Inputs				Output
S_3	S_2	S_1	S_0	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

We need to draw the K-map to realize the correction logic



$$Y = S_2S_3 + S_1S_3$$

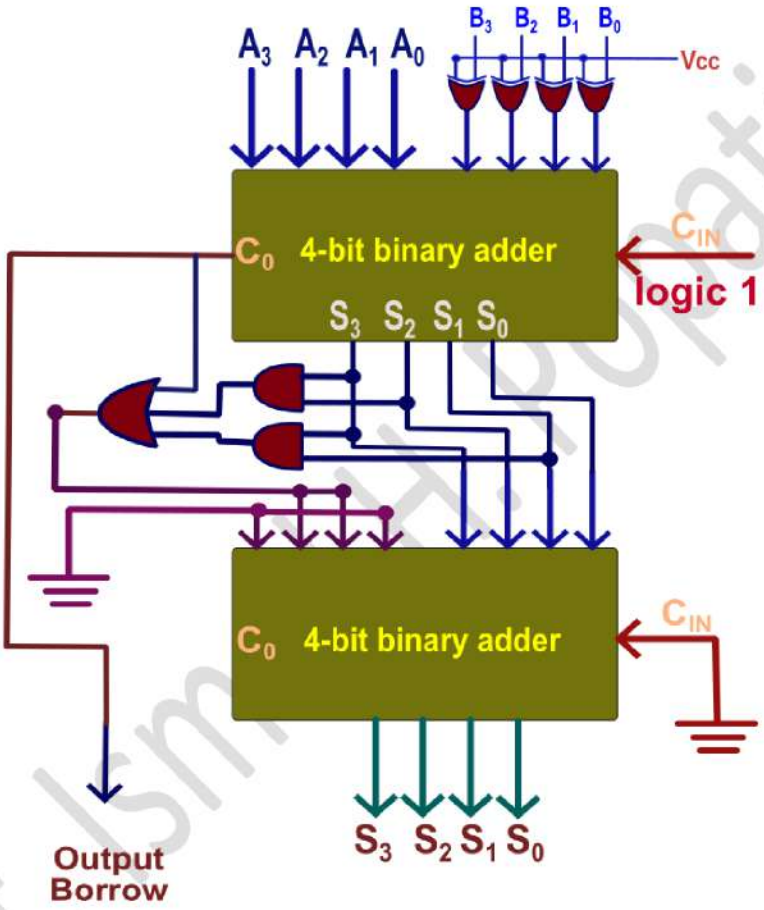
Circuit Diagram:



d) **Design and implement BCD subtractor.**

A BCD subtractor can be designed from the BCD adder, by just adding the logic to find the 2'complement of the minuend.

For e.g if we want to perform the subtraction A – B then we find the 2's complement of B (B') and add it with A (A + B' = S), the resulting sum (S) will give the given difference



For video demonstration of the above practical part follow the link or scan the QR-code

https://youtu.be/UNjCJwtVKYM

e) **Design and implement XS – 3 adder.**

An XS-3 (Excess-3) adder is a digital circuit that adds two XS-3 numbers. XS-3 is a way of representing decimal digits (0-9) by adding 3 (0011) to their binary equivalent, resulting in a 4-bit code. In XS-3 addition, the result must also be in valid XS-3 form. This requires special handling since a simple binary addition of two XS-3 digits can produce a result that is not a valid XS-3 digit or an incorrect value.

Rules for XS-3 addition:

i) If addition of two XS-3 numbers result in a sum less than 9 (1001), then subtract 3(0011) from the given result to get the final result

For e.g

	1	0	1	0	0	XS-3 of 1
+	3	0	1	1	0	XS-3 of 3
	4	1	0	1	0	Incorrect Result

The expected result was 0111 (XS-3) of 4, so to get the proper result we need to subtract 3(0011) from the above result

	4	1	0	1	0	Incorrect Result
-	3	0	0	1	1	Subtract 3
		0	1	1	1	Correct Result

Hence the result is

	1	0	1	0	0	XS-3 of 1
+	3	0	1	1	0	XS-3 of 3
	4	0	1	1	1	Correct Result

ii) If addition of two XS-3 numbers result in a sum greater than 9 (1001), then add 3(0011) to the all the digits, for the final result

For e.g

	6	CY	1	0	0	1	
+	5		1	0	0	0	
	11	1	0	0	0	1	Incorrect result

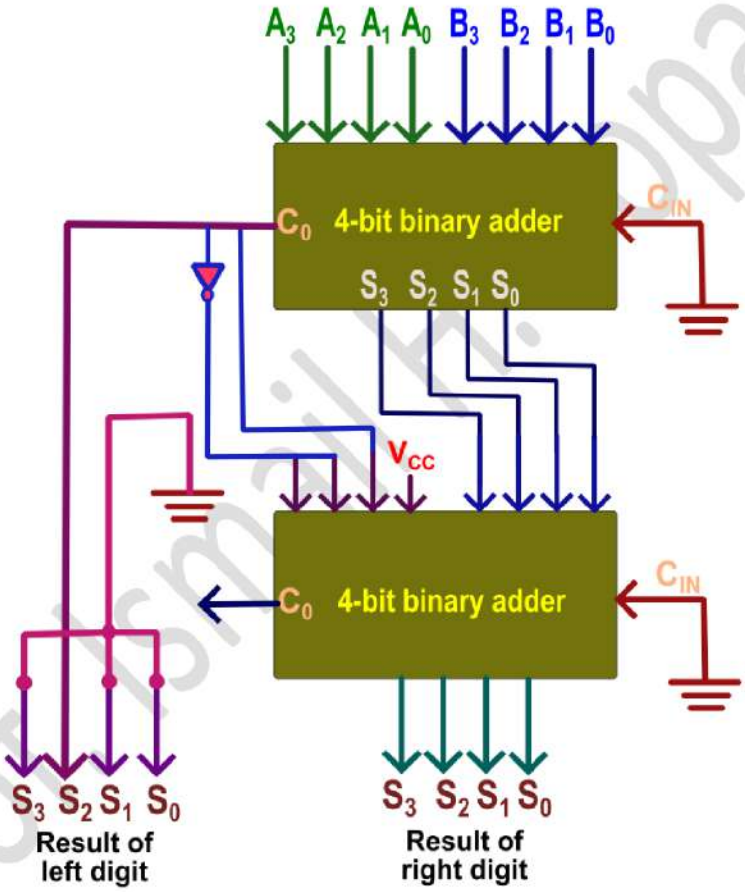
The expected result was 0100 0100 (XS-3) of 11, so to get the proper result we need to add 3(0011) to each digit of the above result

Adding 3 (0 1 1 0) to the result we get

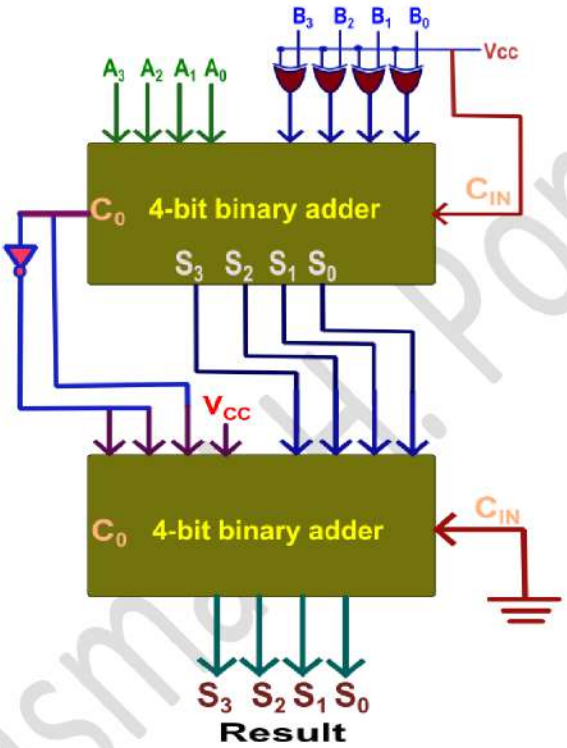
	1	1			1	1		
0	0	0	1	0	0	0	1	Carry and Sum
0	0	1	1	0	0	1	1	Adding 3
0	1	0	0	0	1	0	0	Correct Result

Circuit diagram:

For implementing the correction part, we need to subtract 3(0011) when sum is less than 9, as in adder circuits addition is done using 2's complement which means subtracting 3 is same as adding 2's complement of 3 which is 1101
While when sum is greater than 9, we need to add 3(0011), the following is the logic diagram for XS-3 adder



f) **Design and implement XS – 3 subtractor.**
XS-3 Subtractor can be designed from XS-3 adder, with some added logic. For subtraction we use the 2's complement method and hence for performing A – B, we find 2's complement of B (B') and add it with A, the resulting sum may need some corrections which are applied as done in XS-3 Adder



For video demonstration of the above practical part follow the link or scan the QR-code

https://youtu.be/kSKNxnNpk24

Result: The following Circuits were designed, implemented and verified using the simulator

a) Design and implement half adder and Full adder.
b) Design and implement binary subtractor.
c) Design and implement BCD adder.
d) Design and implement BCD subtractor.
e) Design and implement XS – 3 adder.
f) Design and implement XS – 3 subtractor.