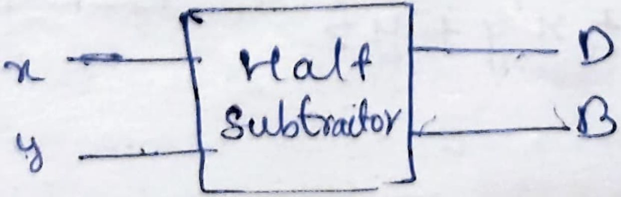
Half Subtractor

→ A half subtractor has two inputs and two outputs.

→ The two inputs x and y form the minuend and the subtrahend.

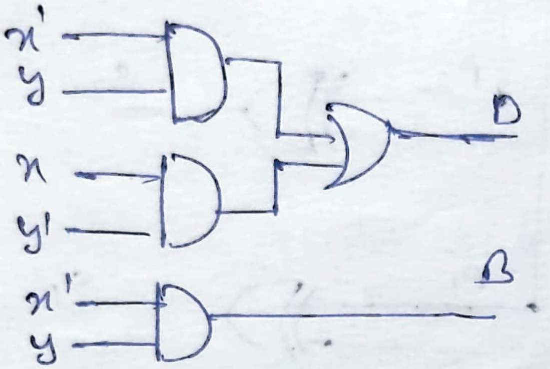
→ D is the difference output and B is the borrow output.

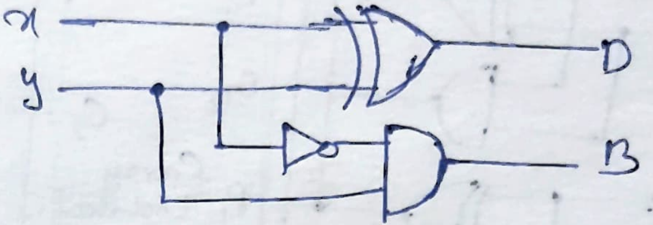


x	y	D	B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$D = x'y + xy'$

$B = x'y$





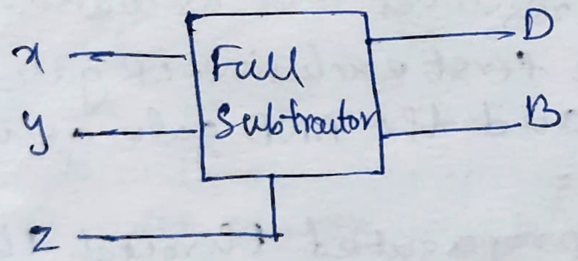
$D = x \oplus y$

$B = x'y$

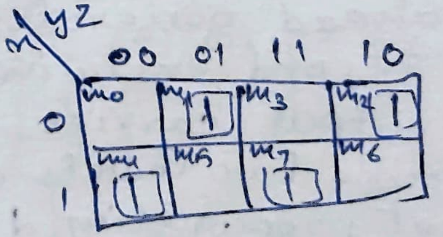
Full Subtractor

→ A full subtractor has three inputs and two outputs x, y, and z are the inputs to be subtracted in which z represents borrow from the next stage.

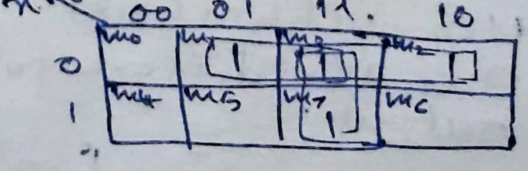
→ D and B are the outputs



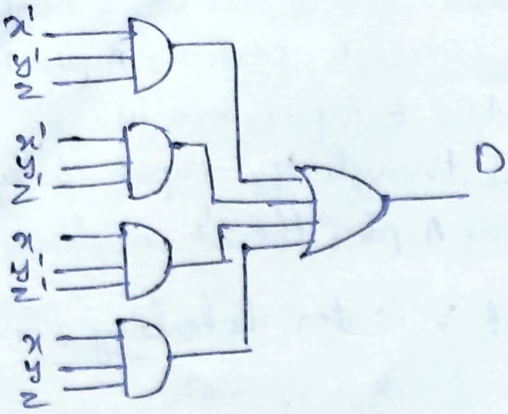
x	y	z	D	B
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

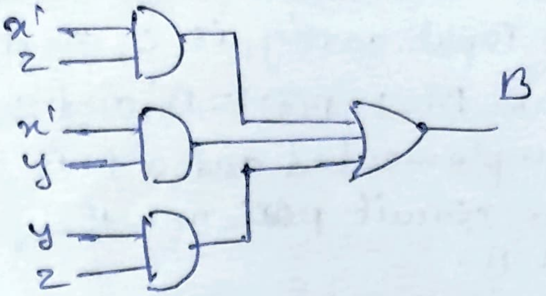


$D = x'y'z + x'yz'$
$+ xy'z' + xyz$



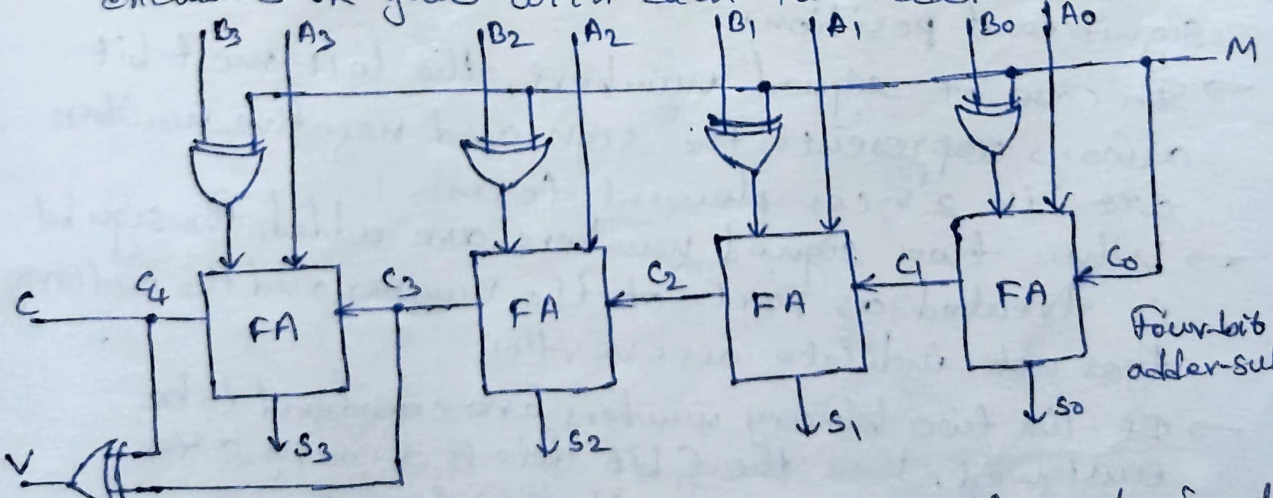
$B = x'z + x'y + yz$





Binary Subtractor

- → The circuit for subtracting A-B consists of an adder with inverters placed between each data input B and the corresponding input of the full adder.
- → The input carry C_0 must be equal to 1 when subtraction is performed. So the operation becomes A, plus the 1's complement of B, plus 1.
- → This is equal to A plus the 2's complement of B
- → for unsigned numbers, the circuit gives A-B if A≥B or the 2's complement of (B-A) if A<B.
- → For signed numbers, the result is A-B, provided that there is no overflow.
- → The addition and subtraction operations can be combined into one circuit with one common binary adder by including an exclusive-OR gate with each full adder.



Four-bit adder-subtractor

- → In four-bit adder-subtractor circuit, the mode input M controls the operation.
- → When M=0, the circuit is an adder, and when M=1, the circuit becomes a subtractor.
- → Each exclusive-OR gate receives input M and one of the inputs of B.

→ When $M=0$, $B \oplus 0 = B$; the full adders receive the value of B, the input carry is 0, and the circuit performs A plus B.

→ When $M=1$, $B \oplus 1 = B'$ and $C_0 = 1$; the B inputs are all complemented and a 1 is added through the input carry. The circuit performs the operation A plus the 2's complement of B.

→ The exclusive-OR with output V is for detecting an overflow.

Overflow

→ When two numbers with n digits each are added and the sum is number occupying $n+1$ digits, we say that an overflow occured.

→ Overflow is a problem in digital computers because the number of bits that hold the number is finite and a result that contains $n+1$ bits cannot be accomodated by an n-bit word.

→ So many computers detect the occurence of an overflow, and when it occurs, a corresponding flip-flop is set that can be checked by the user.

→ The detection of an overflow after the addition of two binary numbers depends on whether the numbers are considered to be signed or unsigned.

→ When two unsigned numbers are added, an overflow is detected from the end carry out of the most significant position.

→ In case of signed numbers, the left-most bit always represents the sign, and negative numbers are in 2's complement form.

→ When two signed numbers are added, the sign bit is treated as part of the number and the end carry does not indicate an overflow.

→ If the two binary numbers are considered to be unsigned, then the C bit detects a carry after addition or a borrow after subtraction.

→ If the numbers are considered to be signed, then the V bit detects an overflow.

→ If $V=0$ after an addition or subtraction, then no overflow occured and the n-bit result is correct.

→ If V=1, then the result of the operation contains n+1 bits, but only the rightmost n bits of the number fit in the space available, so an overflow has occured.

→ The (n+1)th bit is the actual sign and has been shifted out of position.

Carries:	0 1	Carries:	1 0
+70	0 1000110	−70	10111010
+80	0 1010000	−80	10110000
+150	1 0010110	−150	01101010