51, -43 as signed 7-bit nos in binary format.

a) Signed magnitude format (b) 1's compliment format

c) 2's compliment format.

	a	b	c
5	0,000101	0,000101	0,000101
-12	1,001100	1,110011	1,110100
14	0,001110	0,001110	0,001110
-10	1,001010	1,110101	1,110110
26	0,011010	0,011010	0,011010
-19	1,010011	1,101100	1,101101
51	0,110011	0,110011	0,110011
-43	1,101011	1,010100	1,010101
-2	1,000010	1,111101	1,111110

ADDITION & SUBTRACTION OF SIGNED NUMBERS:

x_i	y_i	c_i	s_i	c_{i+1}
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

$$s_i = x_i \oplus y_i \oplus c_i$$

$$c_{i+1} = x_i y_i + x_i c_i + y_i c_i$$

The table above shows truth table for sum & carry-out functions for adding equally weighted bits x_i & y_i in 2 nos x & y.

Ex: $X = 7 \Rightarrow 0111$, $+Y = +6 \Rightarrow 0110$

c_i 1 1

c_{i+1} 0 1 0 1

The logic expression for S_i can be implemented a 3 input Ex-or gate & C_{i+1} is implemented a 2-level AND-OR logic circuit.

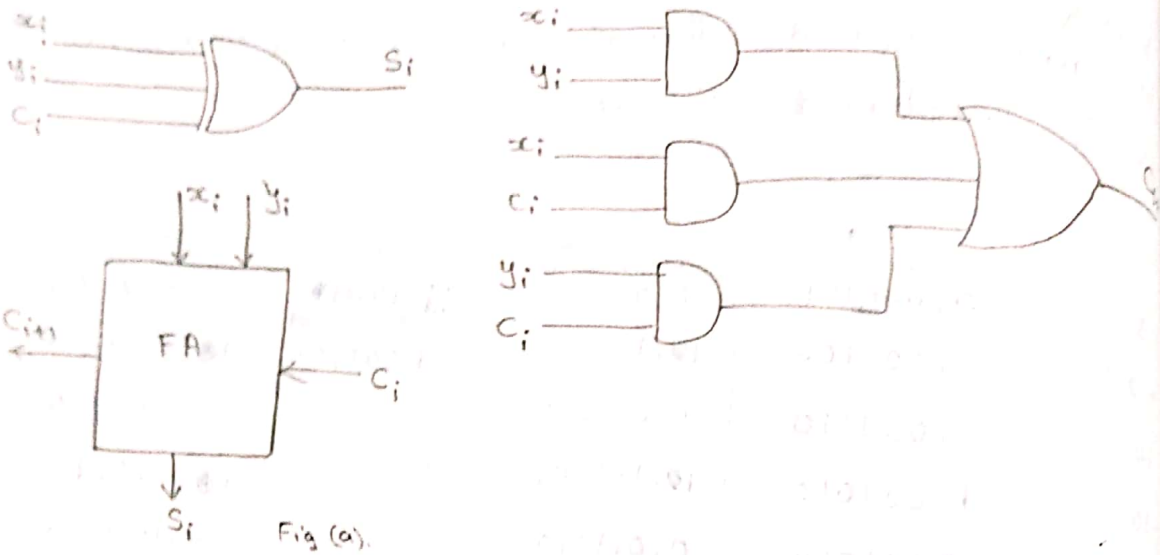
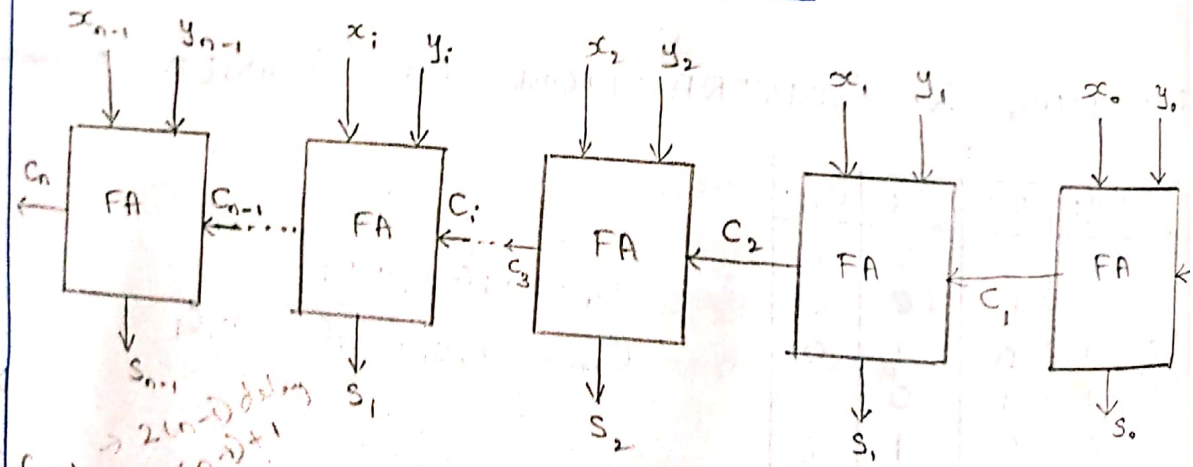
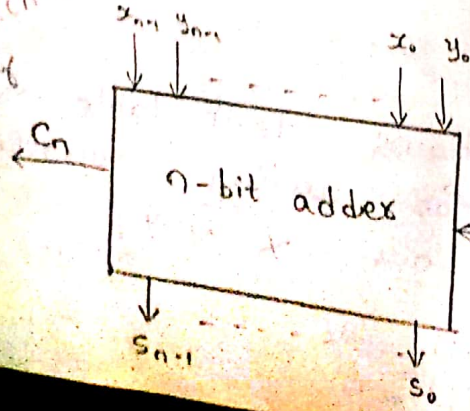


Fig (a).

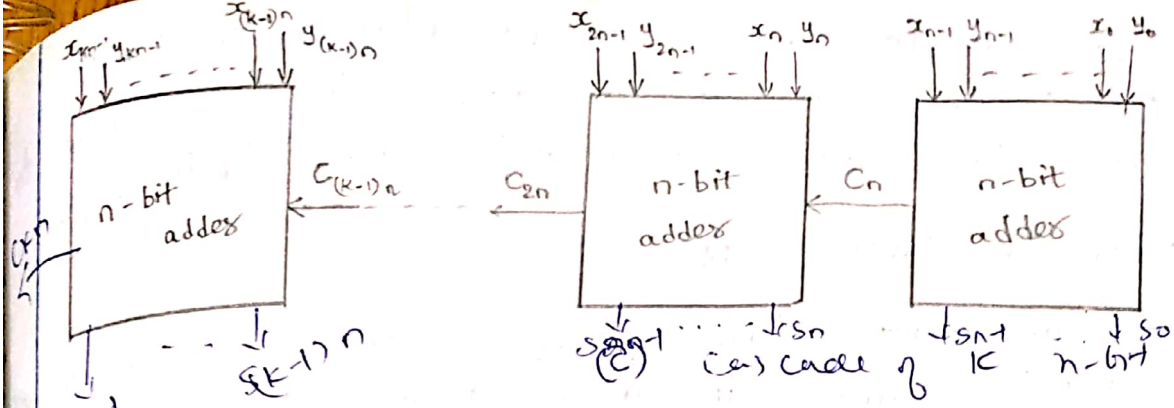
If it is required to add 2 'n'-bit nos, a cascaded connection of 'n' full-adder blocks is as shown below. Since the carry's must propa or ripple through this cascade, the configuratio called n-bit RIPPLE CARRY ADDER.



(a) n-bit ripple carry ad



(b)

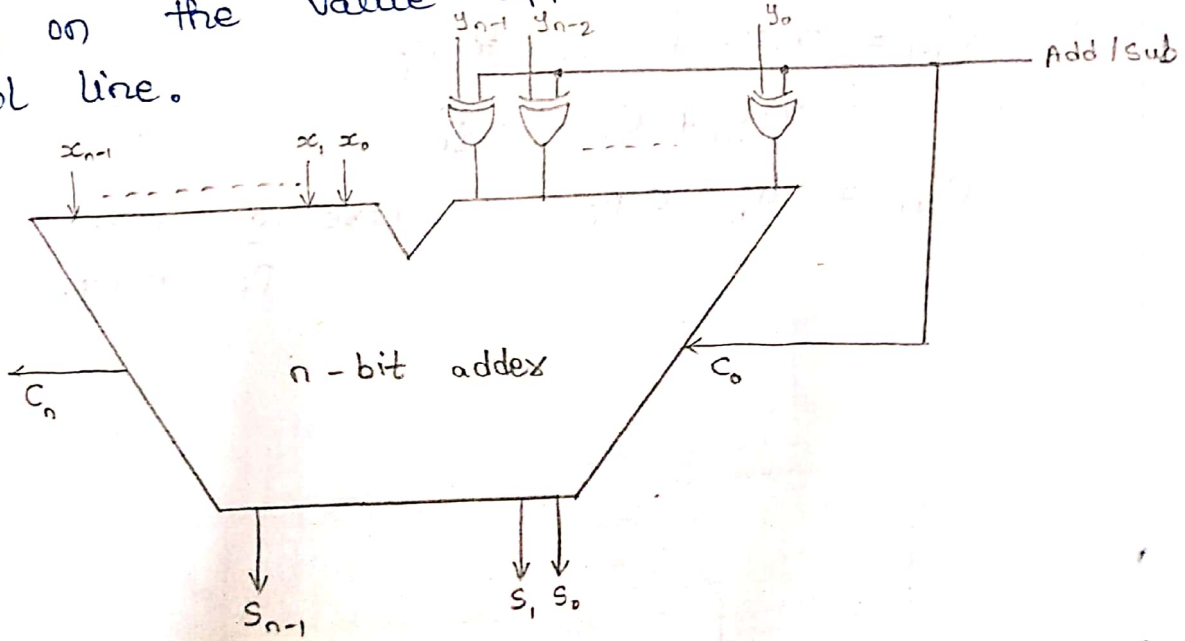


(a) Cascade of IC n-bit adders.

The n-bit adder shown in fig (b) can be used to add 2's compliment nos X & Y where x_{n-1}, y_{n-1} bits represent the sign.

Overflow can occur only when the sign of the result is different i.e., when the sign's of the 2 operands are same. It can be shown that overflow occurs when the carry bits C_n & C_{n-1} are different. Therefore, a simple circuit for detecting the overflow can be obtained by implementing the expression $C_n \oplus C_{n-1}$ with an Ex-or gate.

The logic circuit network shown below can be used to perform either addition or subtraction based on the value applied for 'Add/Sub' input control line.



The Add/Sub control input line is set to zero for addition applying the Y-vector unchanged to one of the adder inputs along with a carry-in signal C_0 of '0'. When Add/Sub line is set to '1', the Y-vector is 1's complemented by the Ex-or gates & C_0 is set to

1, to complete the 2's complementation of y.

An Ex-or gate can be added to the fig. detect the overflow condition i.e., $C_n \oplus C_{n-1}$

** DESIGN OF FAST ADDERS :

If a n-bit ripple carry adder is used in the ADD/SUB unit, it may have too much delay in developing its output, $S_0 \ldots S_{n-1}$ & C_n. All the Sum bits are available in '2n' gate delays including the delay through the XOR gate on the y-input.

The final Carry C_n is available after '2n' delays.

* DESIGN OF CARRY-LOOK AHEAD ADDER :

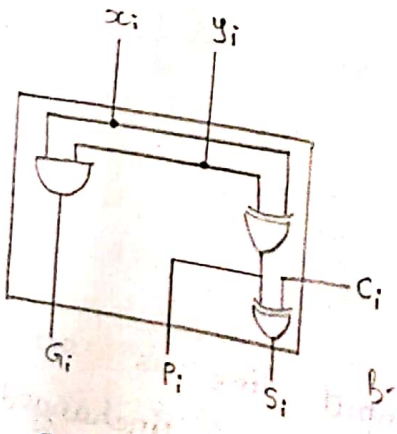
A fast adder must speed-up the generation of Carry signals. The logic expression for S_i (Sum) & C_{i+1} (carry-out) of the i^{th} stage is given by

$$S_i = x_i \oplus y_i \oplus C_i$$

$$C_{i+1} = x_i y_i + x_i C_i + y_i C_i$$

$$= x_i y_i + C_i (x_i + y_i)$$

$$C_{i+1} = g_i + C_i p_i \quad \text{where} \quad g_i = x_i y_i, \quad p_i = x_i + y_i$$



All G_i & P_i functions can be formed independently & in parallel in one logic-gate delay after X & Y vectors are applied to the inputs of n-bit adder.

The realization is shown above which is known as Basic cell - B' which can be used in each stage.

4 bit stage cell

Consider the design of 4-bit adder. The Carries can be implemented as follows:

$C_1 = g_0 + C_0 p_0$

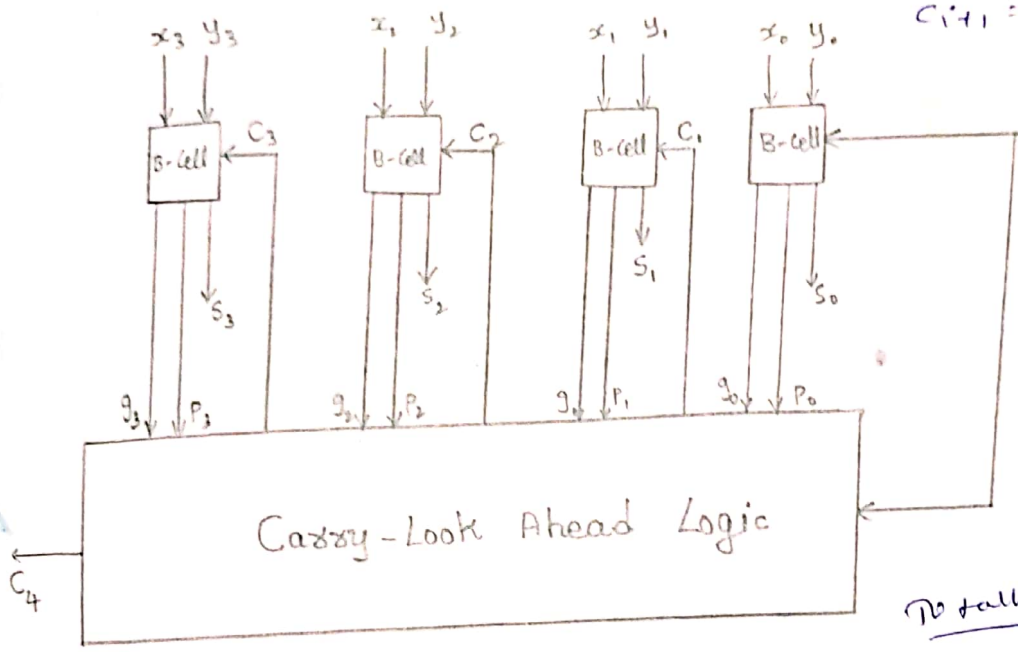
$C_2 = g_1 + p_1 g_0 + p_1 p_0 C_0 = g_1 + C_1 p_1$

$C_3 = g_2 + p_2 g_1 + p_2 p_1 g_0 + p_2 p_1 p_0 C_0 = g_2 + C_2 p_2$

$C_4 = g_3 + p_3 g_2 + p_3 p_2 g_1 + p_3 p_2 p_1 g_0 + p_3 p_2 p_1 p_0 C_0 = g_3 + C_3 p_3$

$C_{i+1} = g_i + C_i p_i$

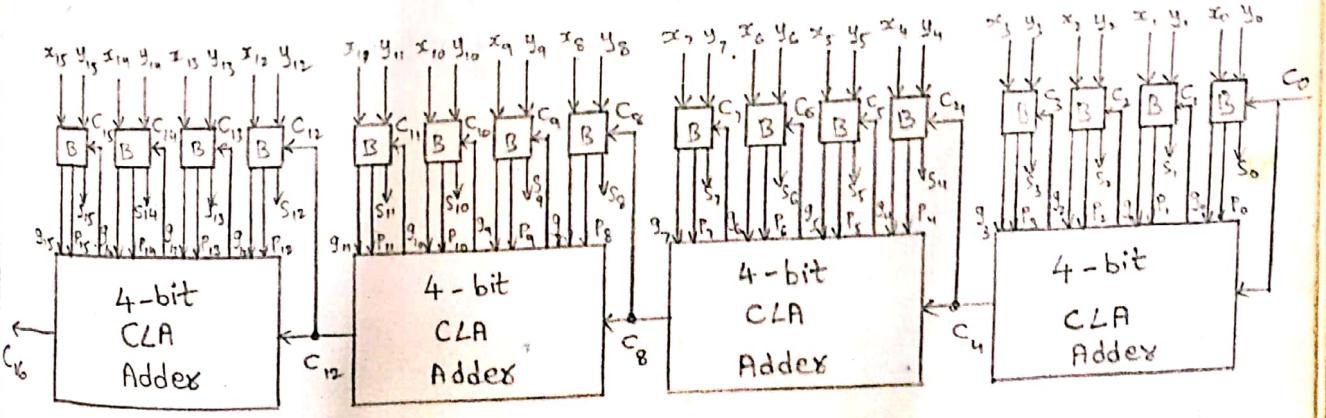
To generate any carry,
1 gate delay for all p_i & g_i
+ 2 gate delay for C_i
= 3 gate delay



for sum
$= x_i \oplus y_i \oplus C_i$
↓
2 gate
= 4 gate delay

Totally for 4-bit CLA
S_3 = 7 gate delays
= previous carry gate for + Sum
= 7 gate delay
C_4 = 8 gate delay
= previous carry gate + plus
= 3

ASSIGNMENT: 16-bit & 32-bit CLA Adder.



16-bit CLA Adder