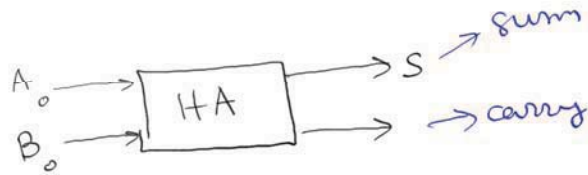




4.7 Binary Adders

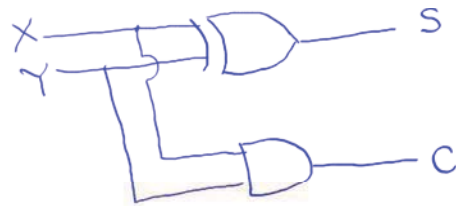
Half Adder



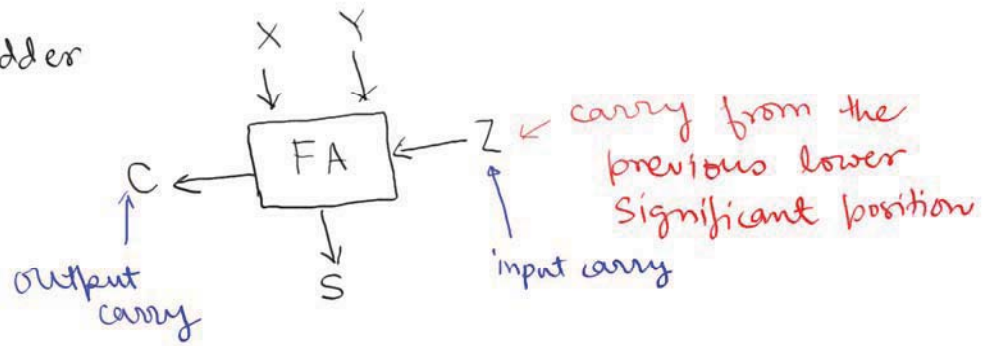
X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

↑ carry

$$S = \bar{X}Y + X\bar{Y} = X \oplus Y$$
$$C = XY$$



Full Adder



X	Y	Z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

See K maps for C & S in the book

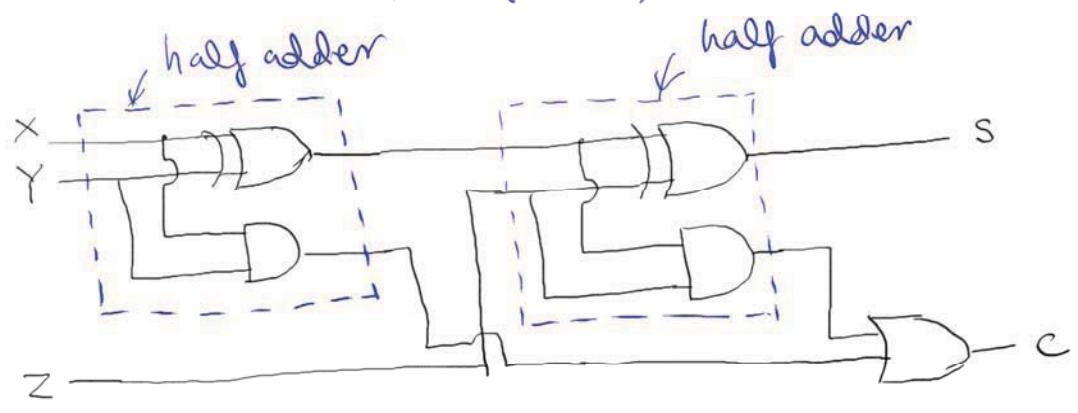
$$S = \bar{X}\bar{Y}Z + \bar{X}Y\bar{Z} + X\bar{Y}\bar{Z} + XYZ$$

$$C = XY + XZ + YZ$$



$$S = X \oplus Y \oplus Z$$

$$C = XY + Z(X \oplus Y)$$



K-Maps for full adder

		Y			
	YZ	00	01	11	10
X	0		1		1
	1	1		1	
		0	1	3	2
		4	5	7	6
		Z			

$$S = \sum(1, 2, 4, 7) = \bar{x}\bar{y}z + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xyz$$

$$= x \oplus y \oplus z$$

		Y			
	YZ	00	01	11	10
X	0			1	
	1		1	1	1
		0	1	3	2
		4	5	7	6
		Z			

Dashed blue circles highlight groups:

- A circle around cells (0,1) and (1,1) is labeled ZY .
- A circle around cells (1,1), (1,2), (1,3), and (1,4) is labeled XY .
- A circle around cells (1,1), (1,2), (1,3), and (1,4) is labeled XZ .

$$C = xy + xz + yz$$

$$= xy + x\bar{y}z + \bar{x}yz$$

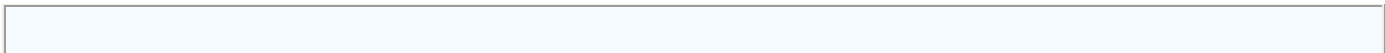
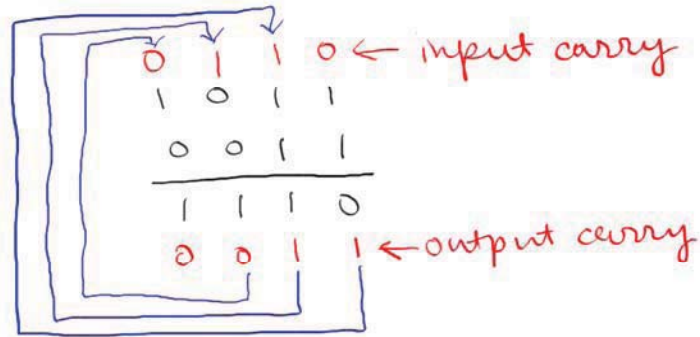
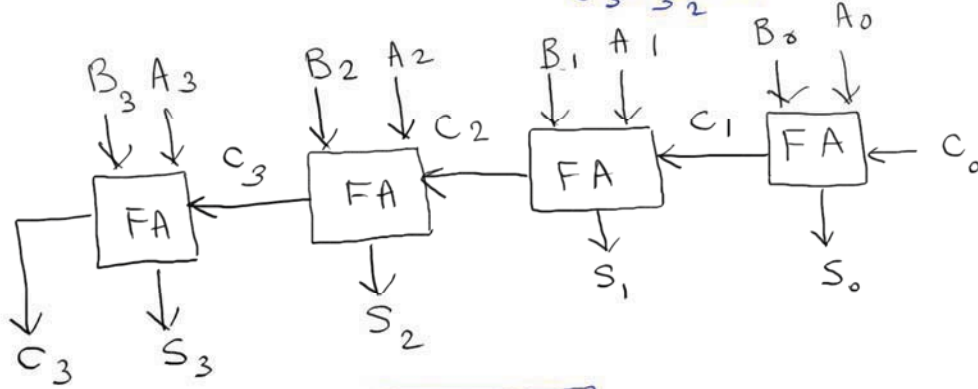
$$= xy + z(x\bar{y} + \bar{x}y)$$

$$= xy + z(x \oplus y)$$

Binary Ripple Carry Adder

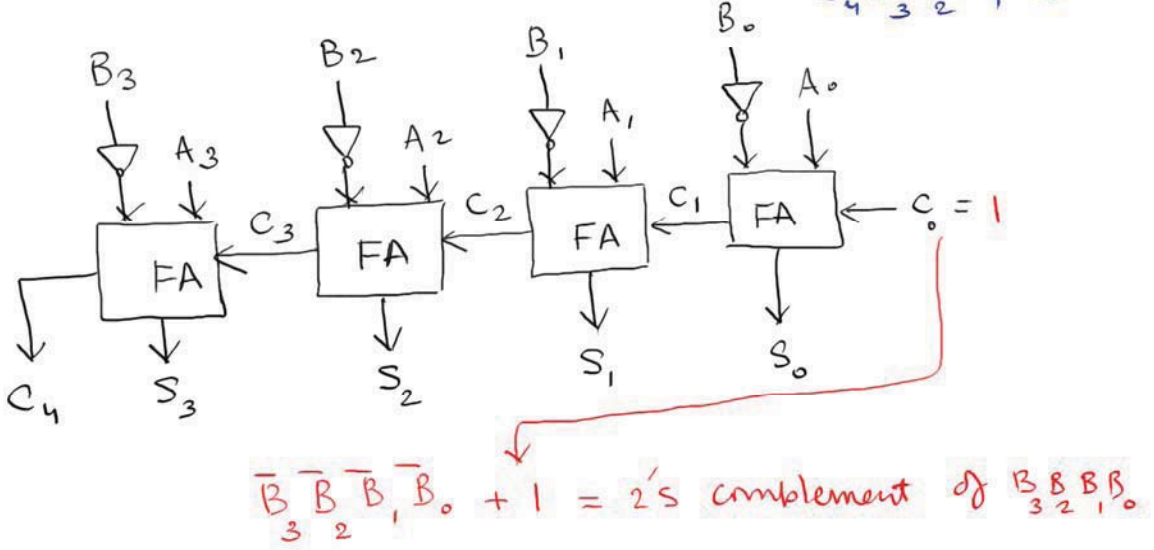
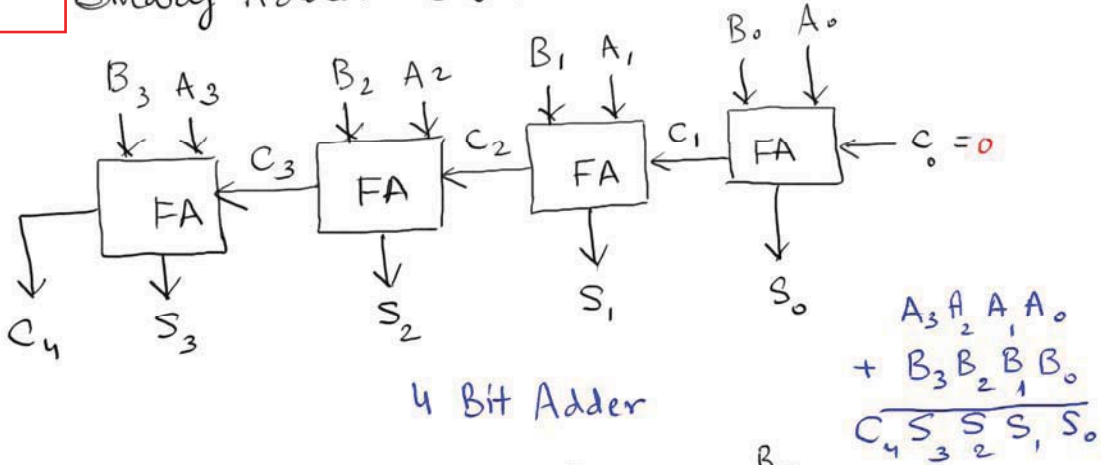
Four bit adder

$$\begin{array}{r}
 A_3 A_2 A_1 A_0 \\
 + B_3 B_2 B_1 B_0 \\
 \hline
 C_3 S_3 S_2 S_1 S_0
 \end{array}$$

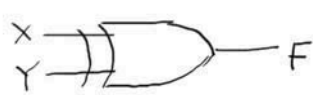




Binary Adder - Subtractors



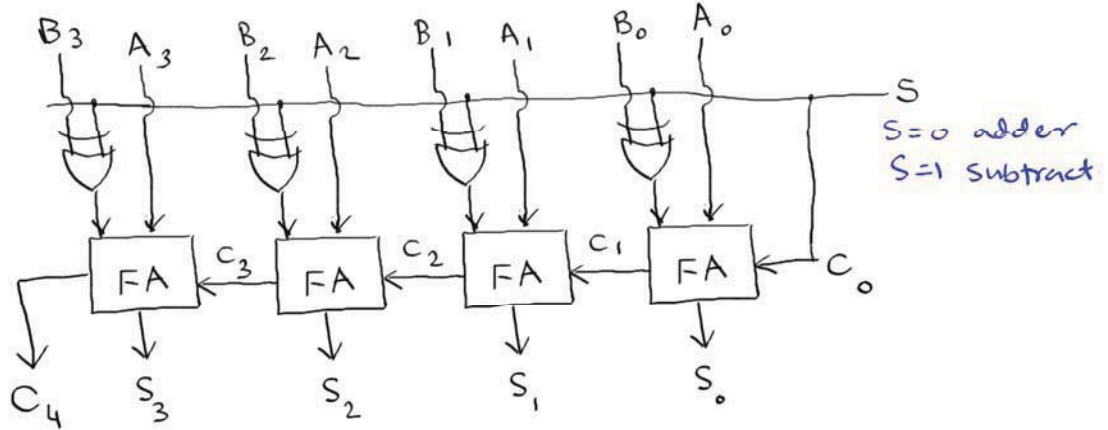
Combining addition & subtraction in one circuit



X	Y	F
0	0	0
0	1	1
1	0	1
1	1	0

\rightarrow

X	F
0	Y
1	\bar{Y}



overflow in 2's complement representation

Assume 8 bits

Sign bit
0 1 1 1 1 1 1 1 → +127 ← maximum positive number
127

1 1 1 1 1 1 1 1 → -1

1 0 0 0 0 0 0 1 → -127

1 0 0 0 0 0 0 0 → -128 ← maximum negative number

Note: complementing the max. negative number → overflow
2's complement of 10000000 should give +128

but +128 needs more than 8 bits!

9 bits
0 1 0 0 0 0 0 0 0 +128
+

keep this "special case" in mind when converting the maximum negative number

8 bits can store a number if $-128 \leq \text{number} \leq +127$

otherwise there will be overflow

overflow in 2's complement representation

Addition: overflow can happen only if the two numbers have the same sign

↻ carry into sign bit position

+ 70	01000110
+ 80	01010000
<hr/>	<hr/>
+150	10010110

< > 127 therefore overflow

↓ wrong result

↻ carry out of sign bit position

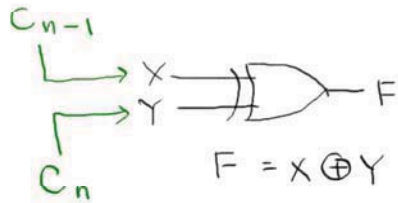
-70	1011010
-80	10110000
<hr/>	<hr/>
-150	1011010

< < -128 therefore overflow

↓ + wrong result

↓ discard

overflow if
 carry into sign bit position \neq carry out of sign bit pos.



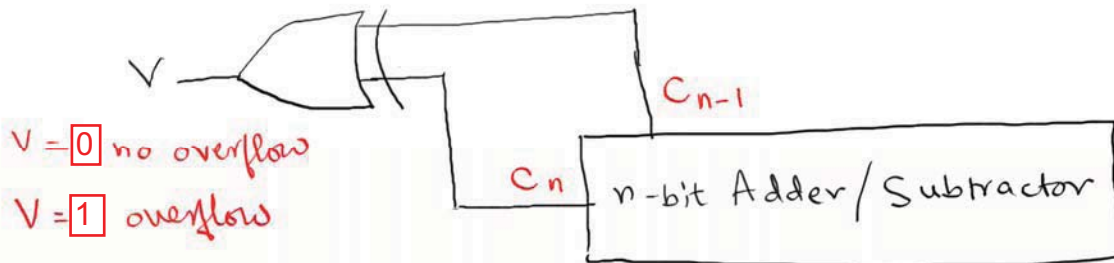
X	Y	F
0	0	0
0	1	1
1	0	1
1	1	0

\rightarrow overflow

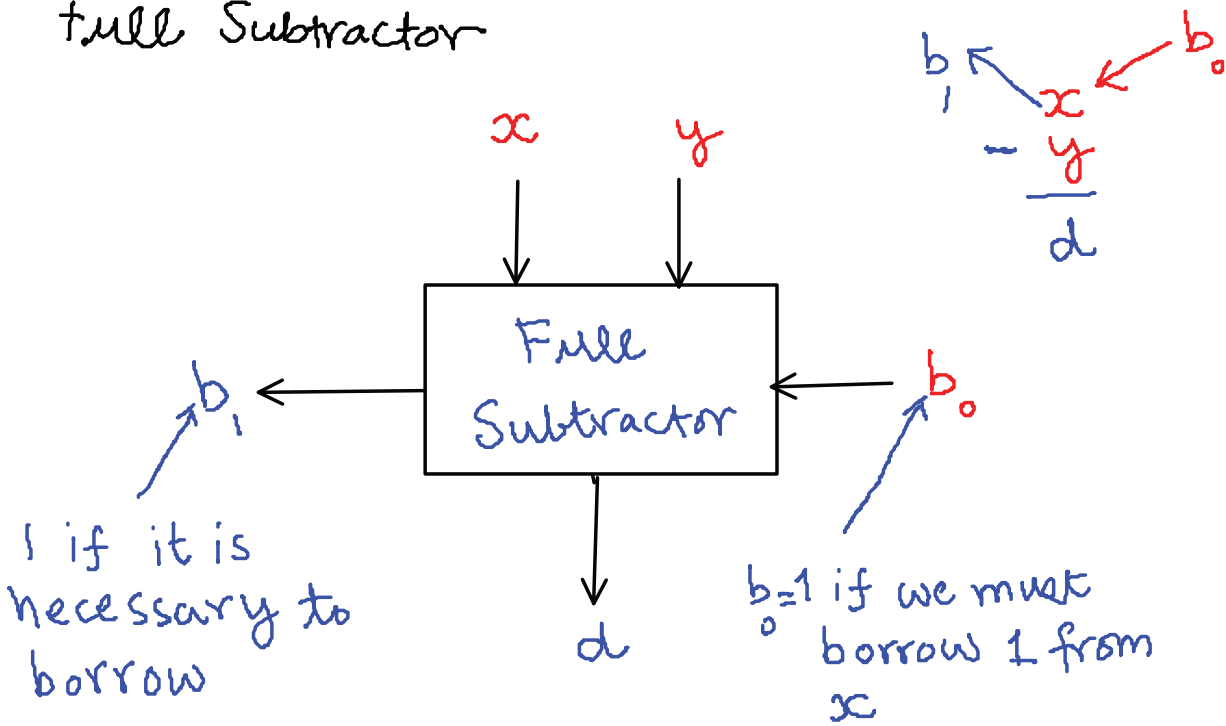
10000000 \rightarrow -128

2's complement = 1's complement + 1
 \rightarrow carry into signed bit position
 01111111 \rightarrow 1's complement
 +1

 10000000
 overflow



Full Subtractor



x	y	b_0	b_1	d
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	0

Handwritten examples of subtraction with borrow propagation:

$$\begin{array}{r} \leftarrow 0 \\ - 0 \\ \hline 0 \end{array}$$

$$\begin{array}{r} \leftarrow 1 \\ - 0 \\ \hline 1 \end{array}$$

$$\begin{array}{r} \leftarrow 1 \\ - 1 \\ \hline 0 \end{array}$$

$$\begin{array}{r} \leftarrow 1 \\ - 1 \\ \hline 0 \end{array}$$

Subtractor

