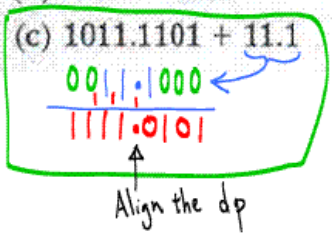
6-1. Add the following in binary. Check your results by doing the addition in decimal.

(a) 1010 + 1011

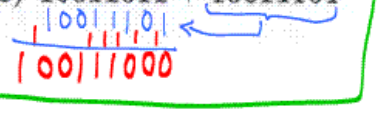
(b) 1111 + 0011

(c) 1011.1101 + 11.1



(d) 0.1011 + 0.1111

(e) 10011011 + 10011101



(Ans. on p. 562)

8-bit → 2^8 = 256 nos. { 0 to +127 ; -1 to -128

6-2. Represent each of the following signed decimal numbers in the 2's-complement system. Use a total of eight bits, including the sign bit.

(a) +32 = 0010 0000 (e) +127 = 0111 1111$_2$ (i) −1 = 1111 1111$_2$ (m) +84

(b) −14 (f) −127 = 1000 0001$_2$ (j) −128 = 1000 0000$_2$ (n) +3 = 0000 0011$_2$ (8-bit)

(c) +63 = 0011 1111 (g) +89 (k) +169 (o) −3 = 1111 1100 +1 = 1111 1101$_2$

(d) −104 (h) −55 (l) 0 = 0000 0000$_2$ (p) −190

Start with +14 = 0000 1110$_2$ (8-bit)

Negation { Invert : 1111 0001 ; Add 1 : 1

to get −14 : 1111 0010

8-bit is not sufficient to represent this signed number!

(h) −55

Start from +55 : 0011 0111 (8-bit)

Invert : 1100 1000

+1 : 1

−55 → 1100 1001

Unsigned Number	8-bit Binary	Signed Number
0	0000 0000	+0
1	0000 0001	+1
2	0000 0010	+2
3	0000 0011	+3
:	:	:
125	0111 1101	+125
126	0111 1110	+126
127	0111 1111	+127
128	1000 0000	−128
129	1000 0001	−127
130	1000 0010	−126
:	:	:
252	1111 1100	−4
253	1111 1101	−3
254	1111 1110	−2
255	1111 1111	−1

2^8 = 256 nos.

The upper half is used for representing **positive** numbers.

(This half is same as unsigned numbers.)

$+(2^7 - 1)$

$-(2^7)$

The lower half is used for representing **negative** numbers.

(-1 at the bottom)

(Signed numbers which are positive have same values as their unsigned number equivalent.)

6-3. Each of the following numbers represents a signed decimal number in the 2's-complement system. Determine the decimal value in each case. (*Hint:* Use negation to convert negative numbers to positive.)

(a) 01101 — 5-bit
(b) 11101
(c) 01111011
(d) 10011001
(e) 01111111 — 8-bit
(f) 10000000 → Pattern for: biggest -ve $(-2^7 = -128)$
(g) 11111111 → Pattern for: -1
(h) 10000001
(i) 01100011
(j) 11011001 — 8-bit

Which of them are negative?

Pattern for: biggest +ve $(+2^7 - 1 = +127)$

(d) Let 10011001 = $-x$

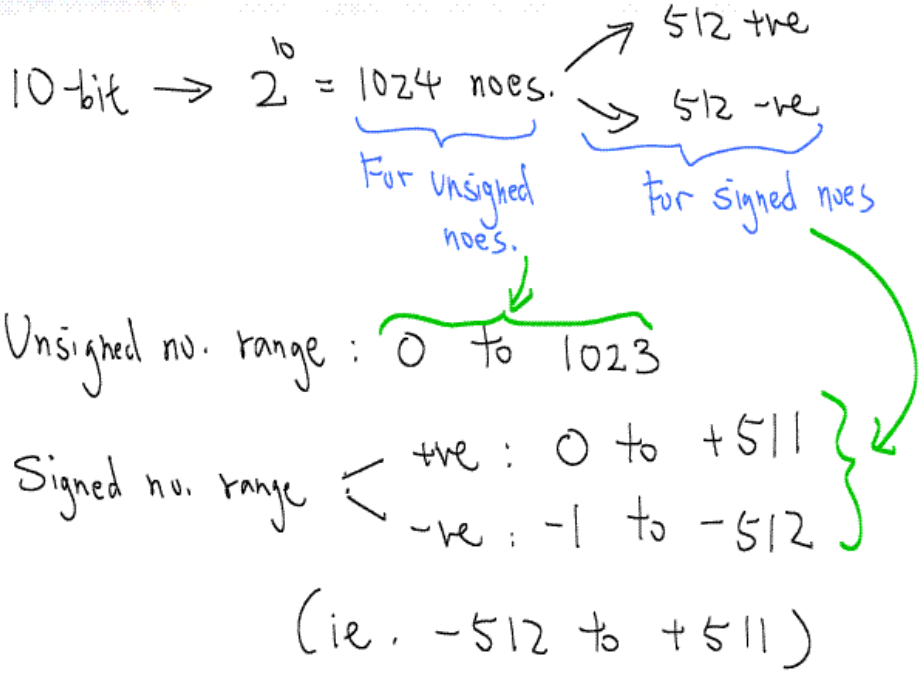
Negation steps:
Invert: 01100110
Add 1: 1
01100111 = $+x$

Weight: 64 32 16 8 4 2 1
1 1 0 0 1 1 1
103

Hence $x = 103 \rightarrow -x = -103$

ie min. to max.

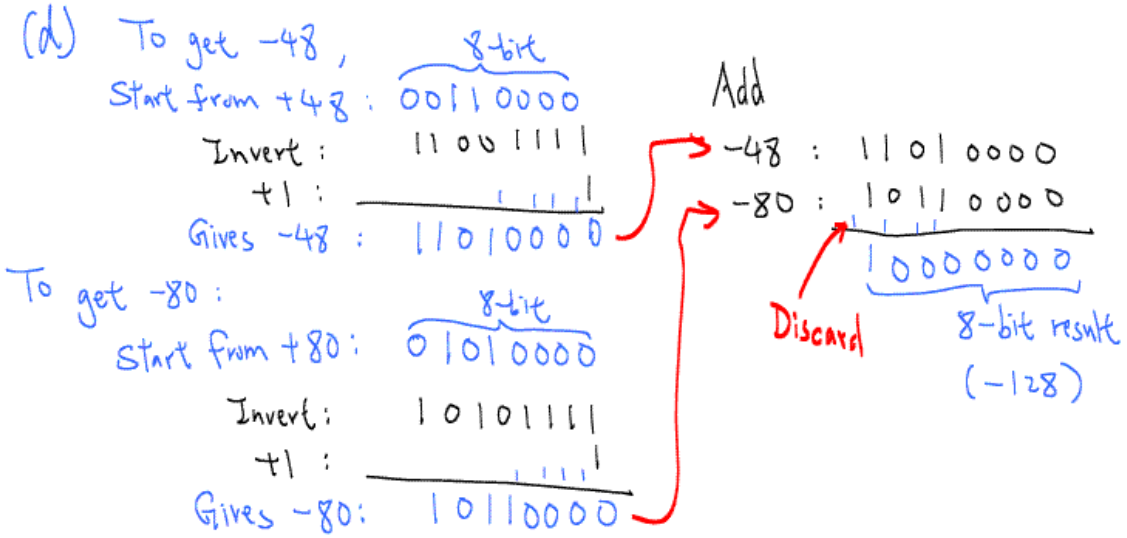
6-7. What is the range of unsigned decimal values that can be represented in 10 bits? What is the range of signed decimal values using the same number of bits?

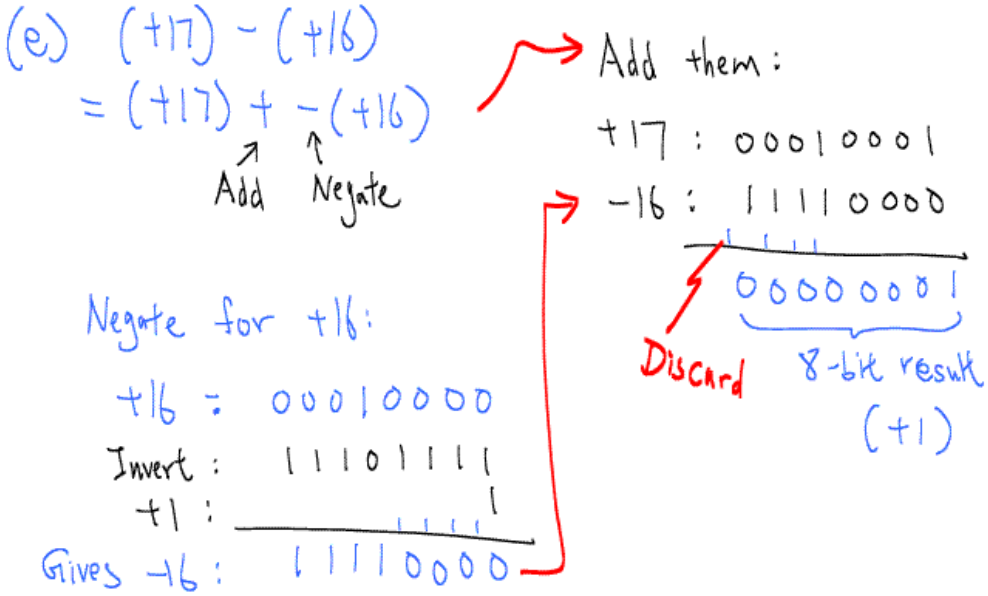


*(Subtracting a number is same as adding its negated version, i.e. : **A - B = A + -B**)*

6-9. Perform the following operations in the 2's-complement system. Use eight bits (including the sign bit) for each number. Check your results by converting the binary result back to decimal.

(a) Add +9 to +6.
(b) Add +14 to −17.
(c) Add +19 to −24.
(d) Add −48 to −80.
(e) Subtract +16 from +17.
(f) Subtract +21 from −13.
(g) Subtract +47 from +47.
(h) Subtract −36 from −15.
(i) Add +17 to −17.
(j) Subtract −17 from −17.





(Too big for 8-bit signed no.!)

6-10. Repeat Problem 6-9 for the following cases, and show that overflow occurs in each case.

(a) Add +37 to +95. = +132 → Too big for 8-bit (max.: +127)

(b) Subtract +37 from −95.

8-bit

+37 : 0010 0101

+95 : 0101 1111

1000 0100

Negative

(Wrong – ie overflow error!)

(−95) − (+37)

= (−95) + −(+37)

= (−95) + (−37)

8-bit

−95 : 1010 0001

−37 : 1101 1011

Discard 0111 1100

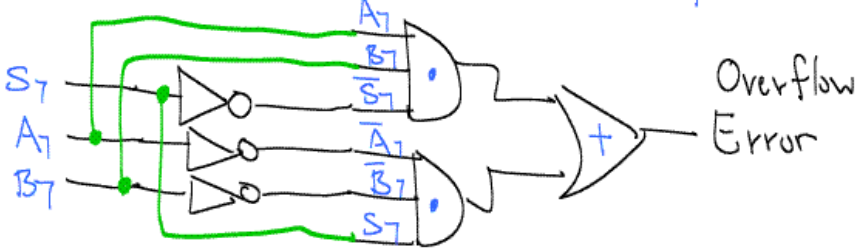
Positive (Overflow)

Additional Problem - Design a logic circuit to detect overflow error.

Sign bit of (MSB)

A7	B7	S7	Error
0	0	0	0
0	0	1	1 ← Add 2 +ve nves and get -ve!
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1 ← Add 2 -ve nves and get +ve!
1	1	1	0

$\therefore E = \overline{A_7} \cdot \overline{B_7} \cdot S_7 + A_7 \cdot B_7 \cdot \overline{S_7}$



6-13. Add the following decimal numbers after converting each to its BCD code. *(BCD: each decimal digit is represented by a 4-bit binary*

(a) 74 + 23
(b) 58 + 37
(c) 147 + 380
(d) 385 + 118
(e) 998 + 003
(f) 623 + 599

3-digit

*In BCD addition, if the sum of the digit (i.e. 4-bit) is > 9, then **add 6** to it for correction*

4-bit per decimal digit

(c) 147 in BCD : 0001 0100 0111
380 " " : 0011 1000 0000

0101 1100 0111
+0110 >9, need correction (ie +6)
0010

(Must show working in the test.)

5 2 7 in BCD (as expected)

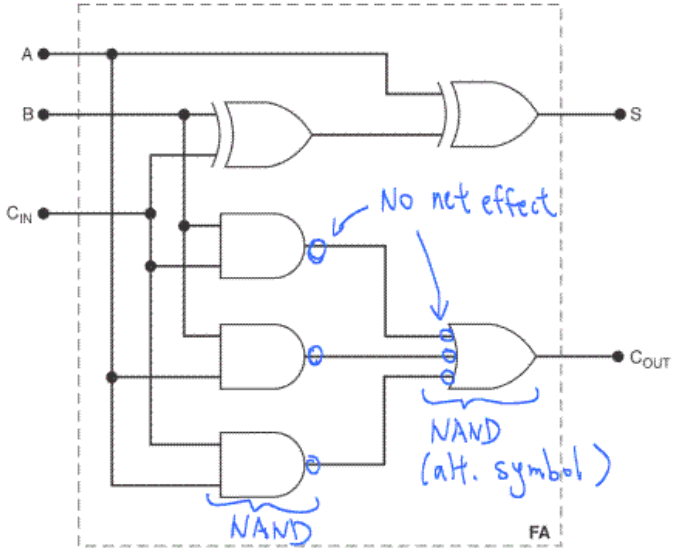
(f) 623 in BCD : 0110 0010 0011
599 " " : 0101 1001 1001

1100 1100 1100 >9, need correction (ie +6)
+0110 +0110 +0110
0001 0010 0010 0010

1 2 2 2 in BCD

6-18 Convert the FA circuit (Fig. 6-7, p.271) to all NAND gates.

Full Adder



For the results in K-map (p. 271):

$$S = \overline{A}\overline{B}C_{IN} + \overline{A}B\overline{C}_{IN} + AB C_{IN} + A\overline{B}\overline{C}_{IN}$$