Note Title 1/29/2009

Nelson, ch. 1 ppt slides 19–35, esp.

Case	Carry	Sign Bit	Condition	Overflow ?
B + C	0	0	$B + C \leq 2^{n-1} - 1$	No
	0	1	$B + C > 2^{n-1} - 1$	Yes
B - C	1	0	$B \leq C$	No
	0	1	$B > C$	No
-B - C	1	1	$-(B + C) \geq -2^{n-1}$	No
	1	0	$-(B + C) < -2^{n-1}$	Yes

- When numbers are represented using two's complement number system:
  - Addition: Add two numbers.
  - Subtraction: Add two's complement of the subtrahend to the minuend.
  - Carry bit is discarded, and overflow is detected as shown above.

For one-digit decimal numbers,

notation of [Roth]

$A - B = A + (10 - B) - 10$, i.e.,

$A - B = A + (\text{10's complement of } B) - 10 = A + B^* - 10 = (A + B^*)$ ignore the carry bit

$$\begin{array}{r} 12 \\ -\ 7 \\ \hline 5 \end{array} \qquad \begin{array}{r} 12 \\ +\ 93 \\ \hline 105 \end{array}$$

93 = 10's compl. of 7

ignore the carry bit, get 5 ✓

$$\begin{array}{r} -3 \\ -6 \\ \hline -9 \end{array} \qquad \begin{array}{r} 97 \\ +\ 94 \\ \hline 191 \end{array}$$

ignore the carry bit, get 91, which is the 100's complement of −9.

This works for any base, and in particular for base 2. Your text only describes base 2.

"The design of logic circuits to do arithmetic with sign and magnitude binary numbers is awkward; therefore other representations are used." [Roth, p. 15]
The other representations are 2's complement and 1's complement.

For the 2's compl. number system, a positive # is represented by $\emptyset$ followed by the magnitude of the number, just as for sign & magnitude; but a negative number, $-N$, is represented by its 2's complement, N^*.
If the word length is n bits, i.e, you have n bits to represent numbers, $N^* = 2^n - N$

$n = 4$ S&M $\quad N^* = 2^n - N = 16 - N = 10000 - N \qquad \bar{N} = (2^n - 1) - N = 15 - N = 1111 - N$

$\bar{N}$ (complement bit-by-bit) = 1's complement of N

+N	-N	S&M	N^*
0	-0	1000	
1	-1	1001	1111
2	-2	1010	1110
3	-3	1011	1101
4	-4	1100	1100
5	-5	1101	1011
6	-6	1110	1010
7	-7	1111	1001
8	-8		1000

$$\begin{array}{r} 10000 \\ -\quad 1 \\ \hline 1111 \end{array} \qquad \begin{array}{r} 10000 \\ -\quad 10 \\ \hline 1110 \end{array} \qquad \begin{array}{r} 10000 \\ -\ 1000 \\ \hline 1000 \end{array}$$

N	$\bar{N}$
0000	1111
0001	1110
0010	1101
0011	1100
0100	1011
0101	1010
0110	1001
0111	1000

two representations of zero

for sign magnitude, 2's compl., and 1's complement representations

Is there a faster way to compute N^* than do the subtraction $2^n - N$? Yes!

There are two ways: (1) complement N bit by bit, then add 1

(2) Take N, start from the right and complement all bits to the left of the first 1.

$$\begin{array}{cccccccccc} 1 & 1 & 1 & 1 \cdots 1 & & & 1 & 1 & 1 & \\ a_n & a_{n-1} & a_{n-2} & a_{n-3} & \cdots & a_i & \cdots & a_2 & a_1 & a_0 \\ \hline \bar{a}_n & \bar{a}_{n-1} & \bar{a}_{n-2} & \bar{a}_{n-3} & \cdots & \bar{a}_i & \cdots & \bar{a}_2 & \bar{a}_1 & \bar{a}_0 \end{array}$$

Addition of 2's compl. #s (similar to 10's compl)

Errors may occur only if addends have the same sign. Errors occur iff sign of result is different from sign of addends.

overflow