

EE 2720

Handout # 2.

Signed Fixed Point (FXP) SystemsRepresentation of Negative Numbers

There are three different systems for representing signed Fixed Point numbers. These are: The

Signed-Magnitude system, the Radix-Complement system and the Diminished Radix-Complement system

- Signed-Magnitude system for integer binary numbers

In the signed-magnitude system an  $n$ -bit integer binary signed number is of the form  $X = x_{n-1}x_{n-2}\dots x_1x_0$ .

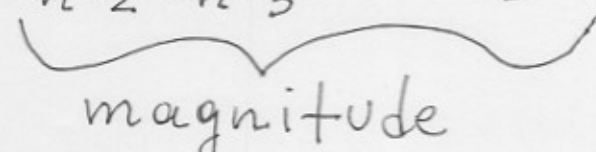
Here the left most bit  $x_{n-1}$  is called

(2)

the sign bit (SB) and dictates if the number  $X$  is positive or negative. If  $x_{n-1} = 0$  that means that  $X$  is positive while if  $x_{n-1} = 1$  that means that  $X$  is negative. The remaining bits  $x_{n-2} \dots x_1 x_0$  represent the magnitude of the number  $X$ .


In such a system any positive number  $X (X > 0)$  is represented as

$$X = \underset{\substack{\uparrow \\ \text{sign} \\ \text{bit}}}{0} x_{n-2} x_{n-3} \dots x_1 x_0$$


  
 magnitude

The additive inverse of  $X$  is then represented as

$$-X = \underset{\substack{\uparrow \\ \text{sign} \\ \text{bit}}}{1} x_{n-2} x_{n-3} \dots x_1 x_0$$


  
 magnitude

In general (for the signed-magnitude system) if

$$X = \overset{\substack{\uparrow \\ \text{sign} \\ \text{bit}}}{x_{n-1}} \underbrace{x_{n-2} \dots x_1 x_0}_{\text{magnitude}}$$

then

$$-X = \overset{\substack{\uparrow \\ \text{sign} \\ \text{bit}}}{x'_{n-1}} \underbrace{x_{n-2} \dots x_1 x_0}_{\text{magnitude}}$$

where  $x'_{n-1} = 1$  if  $x_{n-1} = 0$   
 and  $x'_{n-1} = 0$  if  $x_{n-1} = 1$ .

The Dynamic Range (DR) of an n-bit integer binary signed-magnitude system is

$$DR = \left[ -\left(2^{n-1} - 1\right) \quad + \left(2^{n-1} - 1\right) \right]$$

Some examples follow

(4)

$$+85_{10} = 01010101_2$$

$$-85_{10} = 11010101_2$$

$$+127_{10} = 01111111_2$$

$$-127_{10} = 11111111_2$$

In a signed-magnitude system the number zero has two representations

$$000 \dots 000_2 = +0_{10}$$

$$100 \dots 000_2 = -0_{10}$$

- Addition/Subtraction in the signed-magnitude system.
- Examine sign bits of the two numbers.
  - If sign bits are the same, add magnitudes and give the result the same sign.
  - If sign bits are different, compare the magnitudes, subtract the smaller from the larger, and give the result the sign of the larger.

⑤

\* More about addition/subtraction in the signed-magnitude system will be provided later in another hand-out.

• Generalization for any radix  $r$

Consider an  $n$ -digit integer signed-magnitude number  $X$  in radix  $r$ . The number  $X$  is represented as

$$X = (\overset{\substack{\uparrow \\ \text{sign} \\ \text{digit} \\ \text{or SD}}}{x_{n-1}} \underbrace{x_{n-2} \dots x_1 x_0}_{\text{magnitude}})_r$$

In the above the leftmost digit  $x_{n-1}$  is the sign digit (SD) and dictates if  $X$  is positive or negative. If  $x_{n-1} = 0$  that means that  $X$  is positive while if  $x_{n-1} = r-1$  that means that  $X$  is negative. The

6  
remaining digits  $x_{n-2} \dots x_1 x_0$   
represent the magnitude of  $X$ .

- No further details about radix  $r$  signed-magnitude systems will be provided.

- Radix-Complement system

We first present the general case for any radix  $r$ .

Consider an  $n$ -digit integer number in radix  $r$ .

$$X = (x_{n-1} x_{n-2} \dots x_1 x_0)_r$$

The additive inverse of  $X$  which is  $-X$  is represented by the so called

$r$ 's complement of  $X$ .

The  $r$ 's complement of  $X$  is defined to be

$$\boxed{r\text{'s complement of } X = r^n - X}$$

Example: Find the  $10^3$  complement <sup>⑦</sup> of the number  $1849_{10}$ .

Answer: Here  $r=10$  and  $n=4$ ; ( $n$  is the number of digits of the number). So

$$10^3 \text{ complement of } 1849 = 10^4 - 1849 = 10,000 - 1849 = 8151$$

Example: Find the  $10^3$  complement of the number  $2067_{10}$ .

Answer: Here  $r=10$  and  $n=4$ . So

$$10^3 \text{ complement of } 2067 = 10^4 - 2067 = 10,000 - 2067 = 7933$$

Example: Find the  $10^3$  complement of the number  $8151_{10}$ .

Answer: Here  $r=10$  and  $n=4$ . So

$$10^3 \text{ complement of } 8151 = 10^4 - 8151 = 10,000 - 8151 = 1849$$



⑧

Note: If the number  $X$  is 0 the  $r$ 's complement of  $X$  is  $r^n - 0 = r^n = \underbrace{1000\dots00}_{n \text{ zeroes}}$ . The total length of

$r^n$  is  $n+1$  digits and in this case we ignore the leftmost 1 to obtain the result  $\underbrace{000\dots000}_{n \text{ zeroes}}$ . Therefore,

there is ~~one~~ only one ~~copy~~ representation of the number 0 in the radix-complement system. This representation is  $000\dots000$ . In other words  $+0 = -0 = 000\dots000$ .

- Another way of finding the  $r$ 's complement of a number  $X$  is the following:

Consider an  $n$ -digit integer number  $X$  in radix  $r$ .

$$X = (x_{n-1}x_{n-2}\dots x_1x_0)_r$$

The  $r$ 's complement of  $X$  is then

$r$ 's complement of  $X = (x_{n-1}' x_{n-2}' \dots x_1' x_0')_r + 1$

where in the above

$$x_i' = r-1 - x_i, \quad i = 0, 1, \dots, n-1.$$

Example: Find the  $10$ 's complement of the number  $1849_{10}$ .

Answer: Here  $r=10$ , so  $r-1=9$ . The digit 1 becomes  $9-1=8$ , the digit 8 becomes  $9-8=1$ , the digit 4 becomes  $9-4=5$  and the digit 9 becomes  $9-9=0$ . Therefore

$10$ 's complement of  $1849 = 8150 + 1 = 8151$ .

Example: Find the  $10$ 's complement of the number  $2067_{10}$ .

Answer: Here  $r=10$ , so  $r-1=9$ . The digit 2 becomes  $9-2=7$ , the digit 0 becomes  $9-0=9$ , the digit 6 becomes  $9-6=3$  and the digit 7 becomes  $9-7=2$ . Therefore

(10)

$10^3$  complement of 2067 =  $7932 + 1 = 7933$ .

### • The Two's-Complement System

If the radix is  $r=2$  in which case we have binary numbers, the radix-complement system is called two's complement ( $2^2$  complement) system

Consider an  $n$ -bit integer binary number  $X$

$$X = (x_{n-1}x_{n-2}\dots x_1x_0)_2$$

Here the left most bit  $x_{n-1}$  is the sign bit and dictates if the number  $X$  is positive or negative. If  $x_{n-1} = 0$  that means that  $X$  is positive while if  $x_{n-1} = 1$  that means that  $X$  is negative.

In the two's complement system any positive number  $X$  ( $X > 0$ ) is represented as

$$X = \overset{\substack{\uparrow \\ \text{sign} \\ \text{bit}}}{0} x_{n-2} x_{n-3} \dots x_1 x_0$$

The additive inverse of  $X$  is then represented as

$$\begin{aligned} -X &= \text{two's complement of } X = \\ &= (\overset{\substack{\uparrow \\ \text{sign bit}}}{1} x'_{n-2} x'_{n-3} \dots x'_1 x'_0) + 1 \end{aligned} \quad (2)$$

where in the above

$$\begin{aligned} x'_i &= 1 \quad \text{if } x_i = 0 \\ \text{and } x'_i &= 0 \quad \text{if } x_i = 1 \quad ; \quad i = 0, 1, \dots, n-2 \end{aligned} \quad (3)$$

The above equations (2) and (3) can be derived from equation (1) on page 9 of this handout if we consider that here the radix is  $r=2$  and each bit  $x_i$  can take values of either 0 or 1. Observe that  $r-1=2-1=1$  and  $1-1=0$  while  $1-0=1$ .

In general (for the two's complement system) if  $X$  is

$$X = x_{n-1} x_{n-2} \dots x_1 x_0$$

↑  
sign bit

then the additive inverse of  $X$  is  $-X$  provided below

$$-X = \text{two's complement of } X = (x'_{n-1} x'_{n-2} \dots x'_1 x'_0) + 1$$

where in the above

$$x'_i = 0 \text{ if } x_i = 1$$
$$\text{and } x'_i = 1 \text{ if } x_i = 0 ; \quad i = 0, 1, \dots, n-1$$

Again equations (4), (5) can be derived from equation (1) of page 9 as already explained.

The Dynamic Range (DR) of an  $n$ -bit integer binary two's complement system is

$$DR = \left[ -2^{n-1} + (2^{n-1} - 1) \right].$$

In the two's complement system the number zero has a unique representation. Observe that

$$\begin{array}{r} +0 = 000 \dots 000 \\ -0 = 111 \dots 111 \\ \hline + \quad \quad \quad + \\ 1 \quad 000 \dots 000 \\ \hline \text{ignore} \quad \quad \quad -0 \\ \text{this 1} \end{array}$$

$$\text{So } +0 = -0 = 000 \dots 000.$$

Lemma: Let  $X$  be an  $n$ -bit integer signed binary number where the two's complement system is used for representing signed numbers. The number  $X$  is  $X = x_{n-1}x_{n-2} \dots x_1x_0$ . Here  $x_{n-1}$  is the sign bit of  $X$ . The value of  $X$  is then given by

$$X_{\text{value}} = -2^{n-1} x_{n-1} + \sum_{i=0}^{n-2} x_i \times 2^i \quad (6)$$

(14)

Equation (6) of the lemma simply says that the decimal equivalent of an  $n$ -bit integer binary two's complement number can be computed the same way as for an unsigned number, except that the sign bit  $x_{n-1}$  has a weight of  $-2^{n-1}$  instead of  $+2^{n-1}$ .

Example: Find the value of the following binary number:  $10011101_2$

Answer: According to equation (6) we have  $10011101 = -2^7 \times 1 + 0 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = -128 + 0 + 0 + 16 + 8 + 4 + 0 + 1 = -128 + 29 = -99_{10}$

Some examples of finding the two's complement of 8-bit numbers are presented below.

Example: Find the two's complement of the number  $X$  where  $X = 00010001_2 = 17_{10}$ .

Answer: On next page  $\rightarrow$

According to equations (4) and (5) on page 12, what we have to do is take the number X and change all its zeroes into ones and all its ones into zeroes and then add 1.

This changing the zero-bits into ones and the one-bits into zeroes is called complementing the bits

We now have

$$\begin{array}{r}
 17_{10} = 00010001_2 \\
 \quad \quad \quad \downarrow \text{complement bits} \\
 \quad \quad \quad 11101110 \\
 \quad \quad \quad \quad \quad +1 \\
 \hline
 \quad \quad \quad 11101111_2 = -17_{10}
 \end{array}$$

Example: Find the two's complement of the number  $10011101_2 = -99_{10}$

Answer:

$$\begin{array}{r}
 10011101 \\
 \quad \quad \quad \downarrow \text{complement bits} \\
 \quad \quad \quad 01100010 \\
 \quad \quad \quad \quad \quad +1 \\
 \hline
 \quad \quad \quad 01100011_2 = 99_{10}
 \end{array}$$



Example: Find the two's complement of the number  $01110111_2 = 119_{10}$ .

Answer:

$$\begin{array}{r}
 01110111 \\
 \downarrow \text{complement bits} \\
 10001000 \\
 + 1 \\
 \hline
 10001001_2 = -119_{10}
 \end{array}$$

Two observations follow:

- Observation 1: Consider the following signed numbers where the two's complement system is used to represent signed numbers

$$\begin{aligned}
 &0011_2 = +3 ; 00011_2 = +3 ; 000011_2 = +3 ; \\
 &000 \dots 000011_2 = +3 \\
 &1101_2 = -3 ; 11101_2 = -3 ; 111101_2 = -3 ; \\
 &111 \dots 111101_2 = -3
 \end{aligned}$$

The above shows that copying the sign bit as many times as we want to the left of a number preserves (does not alter or change) its value. This is called sign extension.

• Observation 2: Consider the following signed numbers where the two's complement system is used for representing signed numbers.

$0011_2 = +3$  ;  $00110_2 = +6 = +2^1 \times 3$  ;

$001100_2 = +12 = +2^2 \times 3$  ;  $0011000_2 = +24 = +2^3 \times 3$ .

$1101_2 = -3$  ;  $11010_2 = -6 = -2^1 \times 3$  ;

$110100_2 = -12 = -2^2 \times 3$ .

The above shows that multiplying a number in the two's complement system by  $2^i$  can be mechanized by appending  $i$  zeroes at the right of the number. In other words, no actual multiplication needs to be performed.

• Addition/Subtraction in the two's complement system

• Addition in the two's complement system

Consider two n-bit integer signed binary numbers X and Y where the two's complement system is used for representing signed numbers. The numbers X and Y are shown below

$$X = x_{n-1} x_{n-2} \dots x_1 x_0$$

↑  
sign bit

$$Y = y_{n-1} y_{n-2} \dots y_1 y_0$$

↑  
sign bit.

In order to compute  $X+Y$  using the two's complement system, we add X and Y by ordinary binary addition (by now you know how to do this) and ignore the overall carry out of the addition. This is shown on the next page.

$$\begin{array}{r}
 X = x_{n-1} x_{n-2} \dots x_1 x_0 \\
 + Y = y_{n-1} y_{n-2} \dots y_1 y_0 \\
 \hline
 \rightarrow C \quad z_{n-1} \quad z_{n-2} \dots z_1 \quad z_0 \\
 \underbrace{\hspace{10em}} \\
 X+Y
 \end{array}$$

$C$  is overall carry out of addition and must be ignored

Example: Using the two's complement system perform  $X+Y$  where  $X = 1110_2 = -2_{10}$  and  $Y = 0110_2 = +6_{10}$ .

Answer:

$$\begin{array}{r}
 1110 \\
 + 0110 \\
 \hline
 10100
 \end{array}$$

1 is overall carry out of addition and must be ignored.  $X+Y$ . So  $X+Y = 0100_2 = +4_{10}$

Example: Using the two's complement system perform  $X+Y$  where  ~~$X = 0011_2 = +3_{10}$~~   $X = 0011_2 = +3_{10}$ ;  $Y = 0100_2 = +4_{10}$ .

Answer:

$$\begin{array}{r}
 0011 \\
 + 0100 \\
 \hline
 00111
 \end{array}$$

$\swarrow$  0 is overall carry out of addition  
 $\searrow$   $X+Y = 0111_2 = +7_{10}$

Note: The above described procedure for two's complement addition produces the correct result as long as this result is within the Dynamic Range (DR) of the system.

- Subtraction in the two's complement system

Consider two numbers  $X$  and  $Y$ . To perform  $X-Y$  all we do is

$$X - Y = X + \text{two's complement of } Y \text{ (ignore overall carry out).}$$

Example: Using the two's complement system perform  $X - Y$  where  $X = 1110_2 = -2_{10}$  and  $Y = 1010_2 = -6_{10}$ .

Answer: The two's complement of  $Y$  is two's complement of  $(1010) =$

$$\begin{array}{r} 0101 \\ + \quad 1 \\ \hline 0110 \end{array}$$

We now have

$$\begin{array}{r} 1110 \\ + 0110 \\ \hline 10100 \end{array}$$

1 is overall carry out and must be ignored

$\rightarrow X - Y = 0100_2 = +4_{10}$

Another way of doing the above thing is  
1 ← carry in (cin) of 1 instead of 0

$$\begin{array}{r} 1110 \\ + 0101 \text{ complementing bits of } Y \\ \hline 10100 \end{array}$$

1 is overall carry out and must be ignored

$\rightarrow X - Y = 0100_2 = +4_{10}$

(22)

Example: Using the two's complement system perform  $X - Y$  where  $X = 0011_2 = +3_{10}$  and  $Y = 1100_2 = -4_{10}$ .

Answer: The two's complement of  $Y$  is two's complement of  $(1100) = \begin{array}{r} 0011 \\ + \quad 1 \\ \hline 0100 \end{array}$

We now have

$$\begin{array}{r} 0011 \\ + 0100 \\ \hline 00111 \end{array}$$

↑  
0 is overall carry out

→  $X - Y = 0111_2 = +7_{10}$ .

The second way is shown below

1 ← initial cin of 1.

$$\begin{array}{r} 0011 \\ + 0011 \text{ complementing bits of } Y \\ \hline 00111 \end{array}$$

↑  
0 is overall carry out

→  $X - Y = 0111_2 = +7_{10}$ .

Note: This second way requires only one addition.

Note: The described procedure for two's complement subtraction produces the correct result as long as this result is within the Dynamic Range (DR) of the system.

• Overflow/Underflow

— For the two's complement system, overflow is defined to be the situation where the result of an addition/subtraction of two numbers is larger than the largest value of the Dynamic Range (DR) of the system.

— For the two's complement system, underflow is defined to be the situation where the result of an addition/subtraction of two numbers is smaller than the smallest value of the Dynamic Range (DR) of the system.

Remember that the Dynamic Range (DR) of an  $n$ -bit integer binary two's



complement system is

$$DR = [-2^{n-1} \quad + (2^{n-1} - 1)]$$

Therefore,

- if result  $> 2^{n-1} - 1$  this means overflow.  
and
- if result  $< -2^{n-1}$  this means underflow.

Note: An overflow or underflow might occur only when adding numbers of the same sign. When adding numbers with different signs, overflow or underflow can never occur.

- Overflow/Underflow Detection in the two's complement system

Overflows and underflows are easily detectable in the two's complement system as follows:

If two positive numbers (having sign bits of 0) added together result in a negative result (having sign bit of 1), this indicates an overflow.

And if two negative numbers (having sign bits of 1) added together produce a positive result (having sign bit of 0), this indicates an underflow.

Therefore, in the two's complement system, in order to detect overflows/underflows we compare the sign bits of the two numbers that are added with the sign bit of the result.

Example: Using the two's complement system perform the addition of the 4-bit numbers X and Y where  $X = 0110_2 = +6_{10}$  and  $Y = 0101_2 = +5_{10}$ .

Answer:

$$\begin{array}{r}
 0110 \leftarrow \text{positive number} \\
 + 0101 \leftarrow \text{positive number} \\
 \hline
 01011 \leftarrow \text{negative result; (wrong)}
 \end{array}$$

0 is overall carry out and must be ignored.

sign bit = 1. This means that we got a negative result; (observe that the obtained result is  $1011_2 = -5_{10} < 0$ ). Here an overflow occurred.

Remember that the Dynamic Range (DR)

of a 4-bit integer two's complement system is  $DR = [-2^{4-1} \quad + (2^{4-1} - 1)] = [-2^3 \quad + (2^3 - 1)] = [-8 \quad 7]$  and implies overflow.

Example: Using the two's complement system perform the addition of the 4-bit numbers X and Y where  $X = 1100_2 = -4_{10}$  and  $Y = 1011_2 = -5_{10}$ .

Answer:

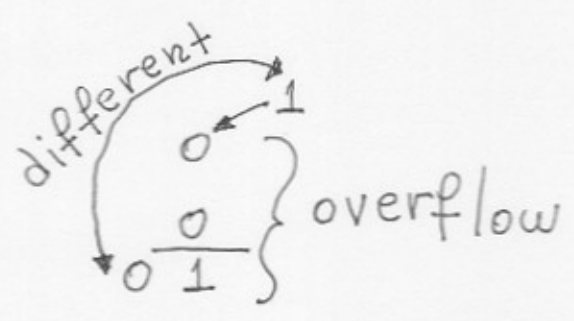
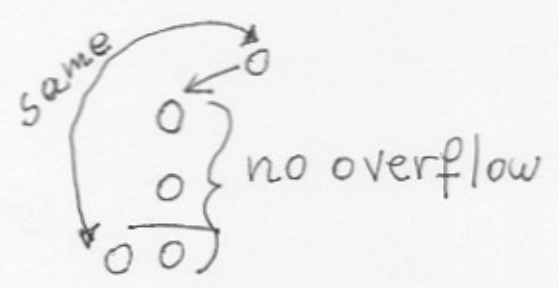
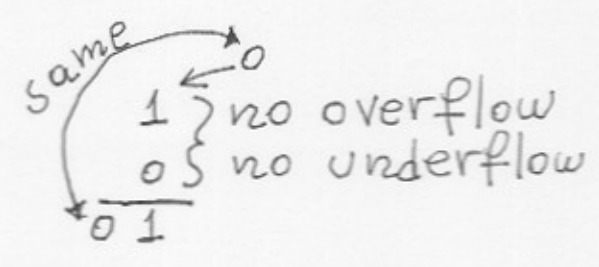
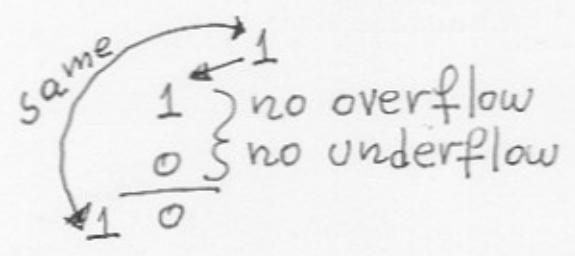
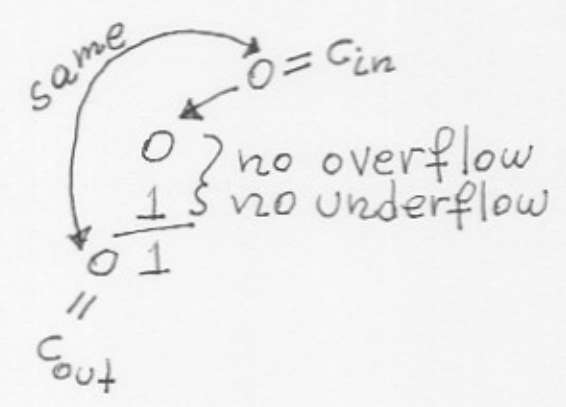
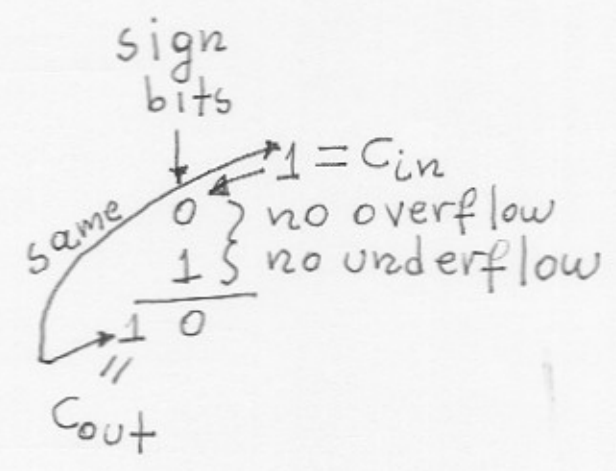
1100	←	negative number
+ 1011	←	negative number
10111	←	positive result; (wrong).

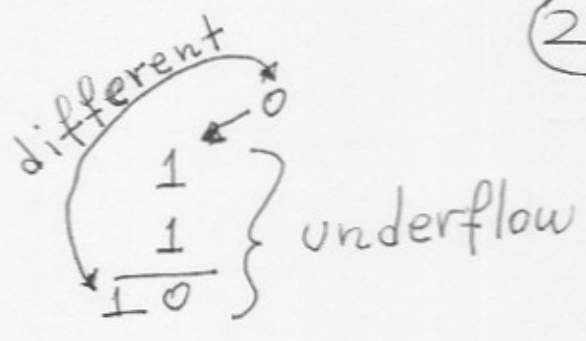
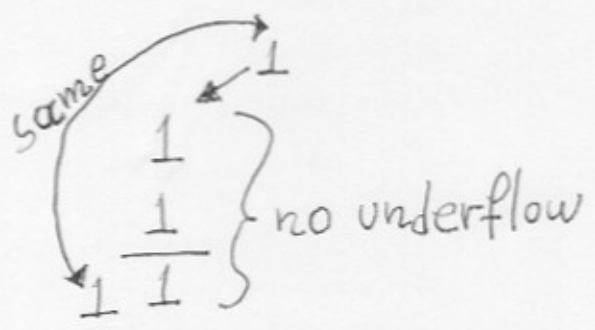
1 is overall carry out of addition and must be ignored

sign bit = 0. This means that we got a positive result; (observe that the obtained result is  $0111_2 = +7_{10} > 0$  which is wrong). Here an underflow occurred.

As we said before, the Dynamic Range of a 4-bit integer two's complement system is  $DR = [-8 \quad 7]$  and  $X + Y = (-4) + (-5) = -9 < -8$  which implies underflow.

- Another way to detect overflows/underflows in the two's complement system is to look at the carry into the sign bit location ( $C_{in}$ ) and the carry out of the sign bit location ( $C_{out}$ ). I now provide all possible scenarios:





From the above, the conclusion is:

- If  $C_{in} = C_{out}$  that means no overflow and no underflow
- If  $C_{in} \neq C_{out}$  that means either overflow or underflow.

More specifically:

- If  $C_{in} = 1$  and  $C_{out} = 0$  that means overflow.
- If  $C_{in} = 0$  and  $C_{out} = 1$  that means underflow.