2’s complement. Discussions and examples.

Mathematically, the two's complement REPRESENTATION of a NEGATIVE number is the value 2^n-V, where V is the absolute value of the represented number

Motto:

With the 2’s complement, “We interpret representations and represent interpretations ”

Obs. Notations in this document that refer to base 2 configurations will NOT contain the suffix b, as this requirement exists only at the level of assembly language syntax. This material does not contain assembly language source code. Instead, the reference to base 16 will contain the suffix h to distinguish it from values in base 10 (ex: 93h vs. 93).

2’s complement. Discussions and examples.

Mathematically, the two's complement representation of a NEGATIVE number is the value 2^n-V, where V is the absolute value of the represented number

1001 0011 (= 93h = 147), so in the UNSIGNED interpretation 1001 0011= 147

Being a binary number starting with 1, in the SIGNED interpretation, this number is negative. Which is its value ?
Answer: Its value is: – (2’s complement of the initial binary configuration) = – (2^n-V)

So, we have to determine the 2’s complement of the configuration 1001 0011
How can we obtain the 2’s complement of a number (represented in memory so we are talking about base 2) ?

Variant 1 (Official): Subtracting the binary contents of the location from 100 ...00 (where the number of zero’s are exactly the same as the number of bits of the location to be complemented).

1 0000 0000 –		2^n -	
1001 0011		V	
01101101	=	6Dh = 96+13 = 109 (so the 2’complement on 8 bits of 147 is 109)	

So, the value of 1001 0011 in the SIGNED interpretation is -109

Variant 2 (derived from the 2’s complement definition – faster from a practical point of view): reversing the values of all bits of the initial binary number (value 0 becomes 1 and value 1 becomes 0), after which we add 1 to the obtained value.

According to this rule, we start from 1001 0011 and reverse the values of all bits, obtaining 0110 1100 after which we add 1 to the obtained value: 0110 1100 + 1 = 0110 1101 = 109

So, the value of 1001 0011 in the SIGNED interpretation is -109

Variant 3 (MUCH MORE faster practically for obtaining the binary configuration of the 2's complement): We left unchanged the bits starting from the right until to the first bit 1 inclusive and we reverse the values of all the other bits (all the bits from the left of this bit with value 1).

Applying this rule, we start from 1001 0011 and left unchanged all the bits starting from the right until to the first bit 1 inclusive (in our case this means only the first bit 1 from the right – which is the only one that is left unchanged) and all other bits will be reversed, so we obtain…0110 1101 = 6DH = 109

So, the value of 1001 0011 in the SIGNED interpretation is -109

Variant 4 (the MOST faster practical alternative, if we are interested ONLY in the absolute value in base 10 of the 2's complement):

Rule derived from the definition of the 2's complement: The sum of the absolute values of the two complementary values is the cardinal of the set of values representable on that size.

On 8 bits we can represent 2^8 values = 256 values ([0..255] or [-128..+127])

- On 16 bits we can represent 2^16 values = 65536 values ([0..65535] or [-32768,+32767])
- On 32 bits we can represent 2^32 valori = 4.294.967.296 values (…)

So, on 8 bits, the 2's complement of 1001 0011 (= 93h = 147) is 256 – 147 = 109, so the corresponding value in SIGNED interpretation for 1001 0011 is -109.

[0..255] – admissible representation interval for "UNSIGNED integer represented on 1 byte" (UNSIGNED BYTE)
[-128..+127] – admissible representation interval for "SIGNED integer represented on 1 byte" (SIGNED BYTE)

[0..65535] – admissible repr. interval for "UNSIGNED integer represented on 2 bytes = 1 word" (UNSIGNED WORD)
[-32768..+32767] – admissible repr. interval for "SIGNED integer represented on 2 bytes = 1 word" (SIGNED WORD)

Why do we need to study the 2’s complement ? is it useful for us as programmers ? In which way ?...

1001 0011 (= 93h = 147), so in the UNSIGNED interpretation 1001 0011= 147

Which is the signed interpretation of 1001 0011 ? a). 01101101 b).-109 c). 6Dh d).+147
Which is the signed interpretation of 93h ? a). 01101101 b).-109 c). 6Dh d).+147
Which is the signed interpretation of 147 in base 10 ? a). 01101101 b).-109 c). 6Dh d).+147 (NONE – because this is a STUPID QUESTION – we cannot have DIFFERENT interpretations in base 10 of numbers ALREADY expressed in base 10 – **147 IS ALREADY AN INTERPRETATION !!**)

```
1 0000 0000 –
  1001 0011
----------------
   01101101      =   6Dh = 96+13 = 109 (so the 2’complement on 8 bits of 147 is 109)
```

So, the value of 1001 0011 in the SIGNED interpretation is … -109

147 and -109 are two complementary values, in the sense that 1001 0011 = either 147, or -109 depending on the interpretation.

So the complement of 147 is -109. Is it also true the other way around ? Is -147 the complement of the 109 ?...

Let’s check…109 = 01101101 , the 2’s complement of 01101101 is 10010011 = 147, so... Which is the conclusion then ?...

Mathematically, the two's complement representation of a NEGATIVE number is the value 2^n-V, where V is the absolute value of the represented number. So the whole discussion about the 2's complement makes practical sense ONLY WHEN we refer to the BINARY REPRESENTATION of a NEGATIVE number from base 10!!! Or to the SIGNED INTERPRETATION of a binary number starting with 1 !! That is, ONLY when we discuss the INTERPRETATION of numbers that in base 2 start with 1 !!!!!! When we have a binary number starting with 0, its INTERPRETATION WILL BE THE SAME in both SIGNED and UNSIGNED, i.e. 01101101 = 109 IN BOTH INTERPRETATIONS !! As a result, the fact that 147 is itself "109's complement of 2 as an absolute value" does not help us at all...

So: all the discussion about complementary values makes sense only if our attention is focused on the NEGATIVE NUMBERS topic!! Ex: we start from a representation in base 2 that begins with 1 and we ask ourselves <u>what will be the associated negative number in the SIGNED interpretation</u> !! Or: we start from an absolute value (109 or 147) and ask ourselves <u>which is the representation in base 2 for -109 or for -147 !!</u> So everything must be around NEGATIVE numbers!!

An approach that starts from a representation in base 2 that begins with 0 and asks ONLY what will be the "complementary value of that representation" (without this complementary value being then used for something concrete) does not make sense out of the context of interpreting a SIGNED number <u>as negative</u>.

The fact that there is a range of values for which the interpretation of a binary configuration is the same in both interpretations happens (taking the byte size as an example) because [0..255] ∩ [-128..+127] = [0..+ 127] and thus any number in the range [0..+127] will represent both the SIGNED and the UNSIGNED version of the given binary configuration! This REPRESENTATION in base 2 will be a sequence of bits starting with 0, such a sequence being characterized by the fact that in base 10 its value will be the same (a positive one!) in both interpretations. These numbers are part of the intersection of the unsigned interpretation [0..255] with the signed one [-128 ..+127].

So the intersection of the admissible representation intervals on a dimension N consists ONLY of the values that in binary begin with bit 0! As a result, binary values starting with bit 1 are NOT common to these "complementary" ranges, meaning that the signed and unsigned interpretations of any binary configuration starting with 1 WILL ALWAYS BE DIFFERENT and they will NEVER be parts of the same admissible representation interval !!! <u>The absolute values of the two interpretations represent two complementary values</u>. Ex: -128, 128; 147, -109; -1, 255; -3, 253; -127, 129 will NEVER be part of the same admissible representation interval. (However, it does NOT apply also to 127, -129 !!! – explanation … later ☺)

So... which is the binary representation for -147? Or how much is 10010011 in the signed interpretation?
147 = 10010011, so... how do we get -147?

More exactly, let's see which are the types of questions that we can ask starting from a given binary configuration and thus determine what are the situations in which the need to use the 2's complement arises:

a) If we have a REPRESENTATION of type $\overline{0xxx}\ldots$ having the value $+\overline{abc}$ in the UNSIGNED interpr., which will be the value of this REPRESENTATION in the SIGNED interpretation ? (b2 – b10)

R: The same ! A number that begins with 0 in base 2 has the same value in base 10 both in signed and unsigned interpretation, being a positive number (109 is +109 in both interpretations).

b). If we have a REPRESENTATION of type $\overline{0xxx}\ldots$ having the value $+\overline{abc}$, which will be the binary REPRESENTATION of $-\overline{abc}$? (Ex: if we consider 109, how is -109 represented in base 2 ?) (b2 – b2)

R: It is only in such a question that "the 2's complement starts to play a role", and the answer is: its REPRESENTATION will be "the 2's complement of the initial binary configuration". For the value 109 = 01101101, the 2's complement of 01101101 is 10010011, so -109 = 10010011
As a result we can conclude that the complementary value of an integer that begins with 0, will begin with 1 (exception making only the value 0) and will fit as a complementary value in the SIGNED interpretation on the same representation size as the initial value !! (-109 is also a byte , similar to 109).

c). If we have a REPRESENTATION of type $\overline{1xxx}\ldots$ having the value $+\overline{abc}$ in the UNSIGNED interpretation, which will be the value in the SIGNED interpretation ? (b2 – b10)

R: The value is: – (the 2's complement of the initial binary configuration). For our example we have:
10010011 = 147 (unsigned) = - (the 2's complement of 10010011) = -(01101101) = -109.

d). If we have a REPRESENTATION of type $\overline{1xxx}\ldots$ having the value $+\overline{abc}$, which will be the binary REPRESENTATION for the value $-\overline{abc}$? (Ex: if we consider 10010011=+147, which is the binary representation for -147 ?) (b2 – b2)

R: The answer can only be one similar to b): its REPRESENTATION will be "the 2's complement of the initial binary configuration". But: if a number begins with 1 in its binary representation and has the value $+\overline{abc}$ in the UNSIGNED interpretation, then its negative variant $-\overline{abc}$ will also have to begin with 1 as its associated binary REPRESENTATION (because otherwise it would not still be a negative number in the SIGNED interpretation). But, complementing a binary value of the form $\overline{1xxx}$.... will provide naturally a binary value STARTING WITH 0 <u>on a representation size identical to the initial one</u> !!! - excepting ONLY the values of the form $\overline{100}$.... (-128, +128,-32768,+32768 etc).

As a result, we conclude that if we start from a representation of the form $\overline{1xxx}$.... of value $+\overline{abc}$ WE CANNOT obtain the value $-\overline{abc}$ ON THE SAME REPRESENTATION SIZE !!!!!

<u>Proof</u> : 147 = 10010011 (147 ∈ [0..255], but -147 ∉ [-128..+127] , so -147 is NOT representable on one byte, even if +147 is !!!!!!)

<u>So not only that the complementation cannot be done correctly on the same representation size as the initial value as a METHODOLOGY, but also the analysis of the admissible representation intervals confirms this SEMANTICALLY!!!!</u>

So… obtaining -147 starting from 147 = 10010011 must be done in the following way:

i). The binary representation of 147 begins with 1, but we must notice that -147 ∉ [-128..+127], but -147 ∈ [-32768..+32767] which concludes that <u>-147 is NOT representable as a byte</u> BUT ONLY AS A WORD !!

ii). On a WORD size, 147 = 00000000 10010011 (so a binary number beginning with 0) and according to b), we have that -147 = "the 2's complement of the initial binary configuration"

The 2's complement of the configuration 00000000 10010011 is 11111111 01101101, so
-147 = 11111111 01101101 = FF6Dh

Let’s check: 11111111 01101101 = FF6Dh = 65389 (UNSIGNED), the sum of the absolute values of the two complementary values being 65389 + 147 = 65536 = the cardinal of the set of values representable on 1 WORD, so the above two interpretations of the configuration 11111111 01101101 are correct and consistent !!

As a result, we may conclude that the involvement of the “2’s complement” is manifest in 3 cases :

Binary format	Interpretation	Value	In what way is involved “2’s complement”	Answer
0xxx	Unsigned	+abc	-	-
	Signed	+abc	How do we represent -abc ? (b)	2’s complement of 0xxx
1xxx	Unsigned	+def	-	-
	Signed		Which value will have 1xxx in the SIGNED interpretation ? (c)	-(2’s complement of 1xxx)
	Signed		How do we represent -def ? (d)	2’s complement of 1xxx’s UNSIGNED extension on 2 * sizeof (1xxx)

(with the exception of the representations of the form $\overline{100}\ldots$ (-128, $+128, -32768, +32768\ etc$).

Let’s notice that the “2’s complement” is NOT involved in any way when we approach only unsigned interpretations ! Example:

Binary nr	Interpretation	Value	“2’s complement” involvement	Answer
01101101	Unsigned	+109	-	-
	Signed	+109	How do we represent -109 ? (b)	10010011
10010011	Unsigned	+147	-	-
	Signed		Which value will have 10010011 in the SIGNED interpretation ? (c)	- (01101101) = -109
	Signed		How do we represent -147 ? (d)	The 2’s complement of 00000000 10010011 which is 11111111 01101101

Columns 3 and 4 from above can be summarized as follows:

Number X in binary representation begins with	-X begins with	-X is represented on	Examples:
0	1	Same sizeof as X	109 = 01101101 ; -109 = 10010011
1	1	2 * sizeof(X)	147 = 10010011; -147 = 11111111 01101101

(exceptions are the representations of the form $\overline{100}\ldots$ (-128, $+128, -32768, +32768\ etc$).

As conclusions:

* for a binary VAL value configuration starting with 0 in base 2, -VAL will be represented on the same representation size as the original representation (109 and -109 are REPRESENTED on the same representation size - 1 byte).

* for a binary VAL value configuration starting with 1 in base 2, -VAL will be represented on a representation size LARGER than the original - of DOUBLE size as an assembly language data type (147 representable on 1 byte, -147 representable on 1 WORD), but in fact only 1 bit more "on paper" from the point of view of the "theory" of the complementation rules (147 = 10010011 = 8 bits, -147 = 1 01101101 = 9 bits required MINIMUM for representation) . So -147 cannot be represented on one byte !!

Which is the MINIMUM number of BITS on which we can represent -147 ?

- On n bits we may represent 2^n values: - either the UNSIGNED values [0..2^n – 1]
  - or the SIGNED values [-2^(n-1), 2^(n-1)-1]

On 8 bits we can though represent 2^8 values (=256 values), either [0..2^8-1] = [0..255] in the UNSIGNED interpretation, either [-2^(8-1), 2^(8-1)-1] = [-2^7, 2^7-1] = [-128..+127] in the SIGNED interpretation

On 9 bits… [0..511] or [-256..+255] and because -147 € [-256..+255] it follows that the MINIMUM number of bits on which we may represent -147 is 9 and -147 representation is:

(…On 9 bits we may represent 512 numbers, 512-147 = 365 = 1 6Dh = 1 0110 1101 …)

So 1 0110 1101 = 16Dh = 256 + 6*16 + 13 = 256 + 96 + 13 = 365 in the UNSIGNED interpretation !

1 0110 1101 = -(2's complement of 1 0110 1101) = -(0 1001 0011) = -(093h)= -147

As a DATA TYPE in ASM, obviously that we have to enroll any value as being a byte, word or dword, so -147 € [-32768..+32767] and accordingly to the above discussion we have -147 = 11111111 01101101 = FF6Dh as a value represented on 1 word = 2 bytes.

Which is the MINIMUM number of BITS on which we can represent 3 ?
Answer: 2 bits; 3 = 11b

On 2 bits we may represent 2^2 values (=4 values), either [0..2^2-1] = [0..3] in UNSIGNED interpretation , either [-2^(2-1), 2^(2-1)-1] = [-2^1, 2^1-0] = [-2..+1] in SIGNED interpretation.

Which is the MINIMUM number of BITS on which we can represent -3 ?
Answer: 3 bits; because on 2 bits AREN'T possible due to the above explanation and on 3 bits we have that:

On 3 bits we may represent 2^3 values (=8 values), either [0..2^3-1] = [0..7] in UNSIGNED interpretation, either [-2^(3-1), 2^(3-1)-1] = [-2^2, 2^2-1] = [-4..+3] in the SIGNED interpretation

As a result,
(…On 3 bits we may represent 8 numbers, 8-3 = 5 = 101b…) - so 101b is the representation of -3 on 3 bits

101b = 5 in UNSIGNED interpretation !

101b = -(2's complement of 101) = -(011) = -3 in the SIGNED interpretation

---

Finally, we have to clarify one thing: <u>Who is for who the "2's complement"</u>? <u>And relative to what</u> ?

One binary configuration compared to another ? Or a signed decimal number versus an unsigned one ? Or two absolute values relative to each other ?... We will see that all these 3 expressions are used in practice...

<u>"2's complement" is reffering to REPRESENTATIONS or to INTERPRETATIONS… ?</u>

Because it is called the 2's complement it is reffering to BASE 2, so we are reffering first of all as a definition to REPRESENTATIONS:

- The 2's complement of 01101101 is 10010011 (this helps us to answer the question "which is the binary representation of -109 ?") (b)

- The 2's complement of 10010011 is 01101101 (this helps us to answer the question "which is the value in the SIGNED interpretation of 10010011 ?") (c)

On the other hand, we ALSO refer to the INTERPRETATIONS, more precisely to the UNSIGNED INTERPRETATIONS of the two complementary binary representations - 01101101 and 10010011 for example - saying that "109 and 147 are two complementary values" (semantically equivalent to "The absolute values of the two interpretations of a binary number starting with 1 represent two complementary values"). That is, 147 and 109 are 2 complementary values, in the sense that the REPRESENTATION of 1001 0011 = either +147 or -109 depending on the interpretation. BUT there is NO representation that has either the value -147, either 109 !!

That is why, in order to distinguish the clarity of what is meant to be emphasized, the expression is often forced in a sense that may be somehow incorrect relative to the definition but relevant as a conclusion in cases like the one above, in the following manner: 147 and 109 there are two complementary values, in the sense that "-109 is the complement of +147", but "-147 is NOT the complement of +109" !!! The forcing here is represented by the appearance of the SIGNS in the expression... but on the other hand, what is meant to be expressed becomes clear.

So... it does NOT make sense to discuss complementary values in base 10 starting from a REPRESENTATION that begins with 0 in base 2 !! ... for example in the sense that starting from 01101101 (109) its complementary value is 10010011 (147). For these concepts to make sense, goals like "representing a negative number" or "interpreting a binary configuration as a signed number"must be involved in the discussion FROM START.

So the involvement of "2's complement" is UNIDIRECTIONAL!!! making sense starting ONLY from the SIGNED interpretation and the representation of negative numbers! This is actually what the above "forcing" expresses finally...