The Carry Flag

Young W. Lim

2024-11-09 Sat

Outline

Based on

- The CARRY flag and OVERFLOW flag in binary arithmetic
  Ian! D. Allen - idallen@idallen.ca - www.idallen.com
  `https://teaching.idallen.com/dat2343/10f/notes/`
  040_overflow.ttx

Compling 32-bit program on 64-bit gcc

- `gcc -v`
- `gcc -m32 t.c`
- `sudo apt-get install gcc-multilib`
- `sudo apt-get install g++-multilib`
- `gcc-multilib`
- `g++-multilib`
- `gcc -m32`
- `objdump -m i386`

TOC: Carry flag

- Examples of signed and unsigned integer arithmetic
- Carry flag in unsigned and signed computations
- Rules for the carry flag
- Method for computing the carry flag
- More examples of the carry flag

TOC: Examples of signed and unsigned integer arithmetic

- Subtracting `0x0000618D` from `0x0000195D`
- Signed and unsigned integer arithmetic examples
- 2's complement of the subtrahend `0x0000618D`
- Interpreting `0xFFFF9E73` (2's complement of `0x0000618D`)
- Unsigned subtraction `0x0000195D` - `0x0000618D`
- Signed subtraction `0x0000195D` + (- `0x0000618D`)
- 2's complement of the subtraction result `0xFFFFB7D0`
- Interpreting the subtraction result `0xFFFFB7D0`
- Summary of signed and unsigned subtractions
- Carry flag (`CF`) and unsigned integer overflows
- Overflow flag (`OF`) and signed integer overflows

- a subtraction example

Minuend	- Subtrahend	= Difference
0x0000195D	– 0x0000618D	= 0xFFFFB7D0

- 2's complement

0x0000618D	↔	0xFFFF9E73
0xFFFFB7D0	↔	0x00004830

- the <u>results</u> of unsigned and signed subtractions have the <u>same</u> <u>bit pattern</u> 0xFFFFB7D0

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-f

- 2's complement

0x0000618D	↔	0xFFFF9E73 *
0xFFFFB7D0 *	↔	0x00004830

- interpreting 0xFFFF9E73

as an unsigned number	+0xFFFF9E73	$+4294942323_{10}$
as a signed number	–0x0000618D	-24973_{10}

- interpreting 0xFFFFB7D0 = 0x0000195D - 0x0000618D

as an unsigned number	+0xFFFFB7D0	$+4294948816_{10}$
as a signed number	–0x00004830	-18480_{10}

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-fo

- 0x0000195D – 0x0000618D = 0xFFFFB7D0

unsigned subtraction	0x0000195D - 0x0000618D	= 0xFFFFB7D0
	$6493 - 24973$	$= 16^8 - 18480 = 4294948816$
	subtraction by hand	borrow (CF=1)
		$16^8 = 2^{32} = 4294967296$

signed subtraction	0x0000195D + (-0x0000618D)	= -0x00004830
	0x0000195D + 0xFFFF9E73	= 0xFFFFB7D0
	$6493 - 24973$	$= -18480$
	the *transformed addition* using	CF = 1 ← Cn = 0
	the 2's complement of subtrahend	CF = !Cn for subtraction

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-f

2's complement of the subtrahend 0x0000618D

- the 2's complement of 0x0000618D → 0xFFFF9E73 (=-24973$_{10}$)

```
                   F    F    F    F    9    E    7    3
0xFFFF9E73   0x1111_1111_1111_1111_1001_1110_0111_0011
0x0000618C   0x0000_0000_0000_0000_0110_0001_1000_1100 (1's complement)
0x0000618D   0x0000_0000_0000_0000_0110_0001_1000_1101 (2's complement)
                   0    0    0    0    6    1    8    D
```

- the 2's complement of 0xFFFF9E73 → 0x0000618D (= +24973$_{10}$)

```
                   0    0    0    0    6    1    8    D
0x0000618D   0x0000_0000_0000_0000_0110_0001_1000_1101
0xFFFF8E72   0x1111_1111_1111_1111_1001_1110_0111_0010 (1's complement)
0xFFFF8E73   0x1111_1111_1111_1111_1001_1110_0111_0011 (2's complement)
                   F    F    F    F    9    E    7    3
```

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-fo

- interpreting 0x0xFFFF9E73

  - as an unsigned Number : +0xFFFF9E73 ($= +4294942323_{10}$)

F	F	F	F	9	E	7	3
15	15	15	15	9	14	7	3
16^7	16^6	16^5	16^4	16^3	16^2	16^1	16^0

  - as a signed number : -0x0000618D ($= -24973_{10}$)

0	0	0	0	6	1	8	D
0	0	0	0	6	1	8	13
16^7	16^6	16^5	16^4	16^3	16^2	16^1	16^0

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-f

- 0x0000195D – 0x0000618D : unsigned subtraction

  subtraction by hand

```
                      0    0    0    0    1    9    5    D
   0x0000195D      0x0000_0000_0000_0000_0001_1001_0101_1101
 - 0x0000618D      0x0000_0000_0000_0000_0110_0001_1000_1101
                      0    0    0    0    6    1    8    D
   -----------------------------------------------------------
   0xFFFFB7D0    1 0x1111_1111_1111_1111_1011_0111_1101_0000 (hand subtraction)
                 1    F    F    F    F    B    7    D    0
                 .
                 V borrow (CF=1) : unsigned integer overflow
```

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-fo

- 0x0000195D – 0x0000618D : unsigned subtraction

  subtraction by hand

  - unsigned integer <u>overflow</u> is indicated by the carry flag (CF)

  - a borrow is indicated by the carry flag (CF=1)
    - whenever an unsigned integer <u>overflow</u> happened
    - $A - B$, when $A < B$, for non-negative integers A, B

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-f

- 0x0000195D + (-0x0000618D) : signed subtraction

  the *transformed addition* using the 2's complement of <u>subtrahend</u>

```
                        0     0     0     0     1     9     5     D
   0x0000195D      0x0000_0000_0000_0000_0001_1001_0101_1101 (+0x0000195D)
 + 0xFFFF9E73      0x1111_1111_1111_1111_1001_1110_0111_0011 (-0x0000618D)
                        F     F     F     F     9     E     7     3

   ----------------------------------------------------------------
   0xFFFFB7D0  0  0x1111_1111_1111_1111_1011_0111_1101_0000 (hand addition)
               0       F     F     F     F     B     7     D     0
 -0x00004830  .  0x0000_0000_0000_0000_0100_1000_0011_0000 (2's complement)
               .       0     0     0     0     4     8     3     0
               V  no carry in the transformed addition (Cn=0) --> (CF=1)
```

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-fo

- 0x0000195D + (-0x0000618D) : signed subtraction

  the *transformed addition* uses the 2's complement of a <u>subtrahend</u>

  - signed integer <u>overflow</u> is indicated by the overflow flag (OF)

  - when a subtraction is performed by a transformed addition,
    the carry flag (CF) is set
    by inverting the carry out (C_n) of a transformed addition

  ---

  ADD instruction :   CF $\leftarrow c_n$ SUB instruction :   CF $\leftarrow \overline{c_n}$

  ---

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-fo

2's complement of the subtraction result 0xFFFFB7D0

- 0xFFFFB7D0 = 0x0000195D - 0x0000618D

- the 2's complement of 0xFFFFB7D0 : 0x00004830 $(= +18480_{10})$

```
                     F    F    F    F    B    7    D    0
0xFFFFB7D0   0x1111_1111_1111_1111_1011_0111_1101_0000
0x0000482F   0x0000_0000_0000_0000_0100_1000_0010_1111 (1's complement)
0x00004830   0x0000_0000_0000_0000_0100_1000_0011_0000 (2's complement)
                     0    0    0    0    4    8    3    0
```

- the 2's complement of 0x00004830 : 0xFFFFB7D0 $(= -18480_{10})$

```
                     0    0    0    0    4    8    3    0
0x00004830   0x0000_0000_0000_0000_0100_1000_0011_0000
0xFFFFB7CF   0x1111_1111_1111_1111_1011_0111_1100_1111 (1's complement)
0xFFFFB7D0   0x1111_1111_1111_1111_1011_0111_1101_0000 (2's complement)
                     F    F    F    F    B    7    D    0
```

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-fc

- interpreting 0xFFFFB7D0 = 0x0000195D - 0x0000618D

  - as an unsigned Number : +0xFFFFB7D0 (= $+4294948816_{10}$)

F	F	F	F	B	7	D	0
15	15	15	15	11	7	13	0
16^7	16^6	16^5	16^4	16^3	16^2	16^1	16^0

  - as a signed number : -0x00004830 (= -18480_{10})

0	0	0	0	4	8	3	0
0	0	0	0	4	8	3	0
16^7	16^6	16^5	16^4	16^3	16^2	16^1	16^0

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-f

- subtracting 0x0000618D from 0x0000195D

  - 0x0000195D - 0x0000618D : unsigned integer subtraction
    hand subtraction

  - 0x0000195D + (-0x0000618D) : signed integer subtraction
    the *transformed addition* using the 2's complement of the <u>subtrahend</u>

  - the same result : 0xFFFFB7D0 (the same bit pattern)

    - interpreting as a unsigned integer 4294948816_{10}
      0xFFFFB7D0 with a borrow (CF=1) $2^{32} - 18480$

    - interpreting as a signed integer -18480_{10}
      -0x00004830 (meaningless CF=1)

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-fo

unsigned subtraction	0x0000195D − 0x0000618D	0xFFFFB7D0
		4294948816_{10}
	subtraction by hand	with a borrow (CF=1)
signed subtraction	0x0000195D + (−0x0000618D)	−0x00004830
		-18480_{10}
	the *transformed addition* using the 2's complement of <u>subtrahend</u>	

- 0x0000195D - 0x0000618D : unsigned subtraction

  - there is an unsigned integer overflow
    so the carry flag will be set (CF=1) to indicate a borrow

  - $A - B$, when $A < B$, for non-negative integers A, B
    (unsigned integers can't be negative),

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-f

Overflow flag (OF) and signed integer overflows

- 0x0000195D + (-0x0000618D) : signed subtraction

  - there is no signed integer overflow
    the overflow flag won't be set (OF=0)

  - signed overflow occurrs in the transformed addition,

    - two *positive* numbers are added and
      the result is a *negative*, (P + P → N), or

    - two *negative* numbers are added and
      the result is a *positive*, (N + N → P)

`https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-fc`

TOC Carry flag in unsigned and signed computations

- 2's complement numbers : 4-bit
- Carrys in signed arithmetics
- Carrys in unsigned arithmetics
- Signed and unsigned arithmetics
- Using the carry flag as a borrow

2's complement numbers : 4-bit

	signed	unsigned
0111	(+7)	7
0110	(+6)	6
0101	(+5)	5
0100	(+4)	4
0011	(+3)	3
0010	(+2)	2
0001	(+1)	1
0000	(0)	0
1000	(-8)	8
1001	(-7)	9
1010	(-6)	10
1011	(-5)	11
1100	(-4)	12
1101	(-3)	13
1110	(-2)	14
1111	(-1)	15

- a <u>carry</u> in an unsigned <u>addition</u> can be obtained
  by performing a *hand* <u>addition</u>
  - CF=1 represents a <u>carry</u> in an unsigned <u>addition</u>

- a <u>borrow</u> in an unsigned <u>subtraction</u> can be obtained
  by performing a *hand* <u>subtraction</u>
  - CF=1 represents a <u>borrow</u> in an unsigned <u>subtraction</u>

Carrys in signed arithmetics

- carry out in a signed addition / subtraction are
    carry out of an *n-bit* 2's complement adder

  - a signed subtraction $A - B$ is performed by
    the transformed addtion $A + \overline{B} + 1$

  - carry out is denoted by C_n

	Addition	Subtraction
Unsigned	Hand Addition with a Carry (CF)	Hand Subtraction with a Borrow (CF)
Signed	2's complement addition with a carry out C_n $A + B$	2's complement transfomred addition with a carry out C_n $A + \overline{B} + 1 \leftarrow A - B$
CF and C_n relation	Carry CF = C_n for addition instructions	Borrow CF = $!C_n$ for subtraction instructions

CF   (Carry Flag)

      - a carry in an unsigned addition
      - a borrow in an unsigned subtraction

C_n (the carry out of an n-bit adder)

      - a carry out in a signed addition
      - a carry out in a signed subtraction

http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt

Signed and unsigned arithmetics (3)

- signed and unsigned additions (or subtractions)

  - share the <u>same</u> *n-bit* result

  - a <u>borrow</u> (CF) in an unsigned <u>subtraction</u> is
    the inverted value of a <u>carry out</u> (C_n) in a signed <u>subtraction</u>

`http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt`

- a <u>borrow</u> (CF) in an unsigned subtraction is
  the inverted value of a <u>carry out</u> (C_n) in a signed subtraction

```
----------------------------------------------------------------
   unsigned add/sub    | signed addition      signed subtraction
----------------------------------------------------------------
     0011   ( 3)        |      0011    (+3)          0011    (+3)
    -1110  -(14)  SUB   |    +0010   +(+2)         -1110   -(-2)  SUB
    -----  -----        |     -----  -----          -----  -----
    10101  ( 5) (-16)   |     00101    (+5)         00101    (+5)
  CF=1                  |    Cn=0                  Cn=0
                        |
     1101   (13)        |      1101    (-3)          1101    (-3)
    -0010  -( 2)  SUB   |    +1110   +(-2)         -0010   -(+2)  SUB
    -----  -----        |     -----  -----          -----  -----
    01011  (11) (+11)   |     11011    (-5)         11011    (-5)
  CF=0                  |    Cn=1                  Cn=1
----------------------------------------------------------------
```

http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt

- a <u>borrow</u> ($CF = 1$) occurs
  in an unsigned <u>subtraction</u> $A - B$
  when $A < B$ as <u>unsigned numbers</u>

- computer hardware can detect
  a <u>borrow</u> ($CF = 1$) in unsigned <u>subtraction</u>
  by looking at whether there is <u>no</u> <u>carry out</u> ($C_n = 0$)
  in the <u>transformed addition</u> $A + \overline{B} + 1$

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-f

- If there is <u>no</u> carry out ($C_n = 0$)
  in the <u>transformed addition</u> $A + \overline{B} + 1$ of $A - B$
  then there is a borrow ($\texttt{CF} = 1$)

- If there is a carry out ($C_n = 1$)
  in the <u>transformed addition</u> $A + \overline{B} + 1$ of $A - B$
  then there is <u>no</u> borrow ($\texttt{CF} = 0$)

- <u>borrows</u> $\texttt{CF} = !C_n$ for <u>subtraction</u> instrucitons
- <u>carrys</u> $\texttt{CF} = C_n$ for <u>addition</u> instrucitons

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-f

- the same *addition* and *subtraction* instructions
  are used for both unsigned and signed integer arithmetic.

  - no special *addition* and *subtraction* instructions
    for unsigned and signed integer arithmetic

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-fo

- the only difference is
  - <u>which flags</u> you *test* afterwards
    - CF for unsigned <u>additions</u> and <u>subtractions</u>
    - OF for signed <u>additions</u> and <u>subtractions</u>
  - how you *interpret* the result
    - Carry CF unsigned <u>additions</u>
    - Borrow CF unsigned <u>subtractions</u>
    - Overflow OF signed <u>additions</u> and <u>subtractions</u>

https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-fo

- The <u>1st</u> rule for <u>setting</u> the carry flag
- The <u>2nd</u> rule for <u>setting</u> the carry flag
- Cases for <u>clearing</u> the carry flag
- Computing CF in unsigned additions and subtractions

2's complement numbers : 4-bit

	signed	unsigned
0111	(+7)	7
0110	(+6)	6
0101	(+5)	5
0100	(+4)	4
0011	(+3)	3
0010	(+2)	2
0001	(+1)	1
0000	(0)	0
1000	(-8)	8
1001	(-7)	9
1010	(-6)	10
1011	(-5)	11
1100	(-4)	12
1101	(-3)	13
1110	(-2)	14
1111	(-1)	15

1. CF = 1 : carry in unsigned addition

   - *hand addition rule*

   - the carry flag is set
     if the addition of two unsigned numbers causes
     a carry out of the most significant bits added.

   - unsigned integer overflow in unsigned addition

   `http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt`

The 2nd rule for setting the carry flag

2. **CF = 1** : <u>borrow</u> in unsigned <u>subtraction</u>

   - *hand <u>subtraction</u> rule*

   - the <u>carry flag</u> is <u>also</u> set
     if the <u>subtraction</u> of two unsigned numbers requires
     a <u>borrow</u> into the most significant bits subtracted.

   - unsigned <u>integer overflow</u> in unsigned <u>subtraction</u>

`http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt`

1. **CF = 1** : <u>carry</u> in unsigned <u>addition</u>
   - all three interpretations have the same CF=1, the same S=0000

```
--------------------------------------------------------------------
unsigned addition    |   signed addition        signed subtraction
--------------------------------------------------------------------
 0111   ( 7)         |    0111   (+7)            0111   (+7)
+1001  +( 9)         |  +1001   +(-7)           -0111   -(+7)
-----  -----         |  -----   -----           -----   -----
10000   ( 0) +(16)   |  10000   ( 0)            10000   ( 0)
CF=1                 |  Cn=1 -> CF=1            Cn=1 -> CF=1
                     |
CF means +16         |  CF meaningless          CF meaningless
 S = 0000            |   S = 0000                S = 0000
 -------------------------------------------------------------------
* think hand         |  * think Cn of the corresponding addition
  addition           |    CF <- Cn
 -------------------------------------------------------------------
```

http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt

Cases for setting the carry flag (2)

2. **CF** = 1 : <u>borrow</u> in <u>unsigned</u> <u>subtraction</u>
   - all three interpretations have the same CF=1, the same S=0000

```
--------------------------------------------------------------------
unsigned subtraction |    signed addition          signed subtraction
--------------------------------------------------------------------
 0111   ( 7)         |    0111   (+7)               0111   (+7)
-1001  -( 9)         |  +0111  +(+7)               -1001  -(-7)
-----  -----         |  -----  -----               -----  -----
11110   (14) -(16)   |  01110   (14)               01110   (14)
                     |
CF=1                 |  Cn=0 -> CF=1               Cn=0 -> CF=1
                     |
CF means -16         |  CF meaningless             CF meaningless
 S = 1110            |   S = 1110                   S = 1110
--------------------------------------------------------------------
* think hand         |  * think Cn of the corresponding addition
  subtraction        |    CF <- Cn
--------------------------------------------------------------------
```

http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt

- Computing CF in an unsigned addition
  - do the signed 2's complement addition
  - Cn is the carry out
  - CF ← Cn (CF means carry)

- Computing CF in an unsigned subraction
  - do the transformed signed 2's complement addition
  - Cn is the carry out
  - CF ← !Cn (CF means borrow)

http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt

TOC: Method for computing the carry flag

- 2's complement numbers : 4-bit
- Addened and augend in a n-bit addition
- Full adder (hardware) operation
- Internal and external carry bits
- 2's complement signed addition and subtraction
- Carry flag computation

n	bits	addened	A	$\{$	$a_{n-1}, a_{n-2}, \cdots, a_1, a_0\}$
n	bits	augend	B	$\{$	$b_{n-1}, b_{n-2}, \cdots, b_1, b_0\}$
(n+1)	bits	carry bits	C	$\{c_n,$	$c_{n-1}, c_{n-2}, \cdots, c_1, c_0\}$
n	bits	sum bits	S	$\{$	$s_{n-1}, s_{n-2}, \cdots, s_1, s_0\}$

external carry bits : c_n carry out, c_0 carry in

	a_{n-1}	a_{n-2}	$\cdots\cdots\cdots$	a_1	a_0
	b_{n-1}	b_{n-2}	$\cdots\cdots\cdots$	b_1	b_0
					c_0
c_n	s_{n-1}	s_{n-2}	$\cdots\cdots\cdots$	s_1	s_0

Full adder (hardware) operation

full adder operation in the ith bit position

$$\{c_{i+1}, s_i\} = a_i + b_i + c_i$$

$$
\begin{array}{cc}
 & a_i \\
 & b_i \\
 & c_i \\
\hline
c_{i+1} & s_i
\end{array}
$$

Internal and external carry bits

external carrys	c_n output, c_0 input	
internal carrys	$\{c_{n-1}, c_{n-2}, \cdots\cdots, c_2, c_1\}$	output / input
sum bits	$\{s_{n-1}, s_{n-2}, \cdots\cdots, s_1, s_0\}$	output

	a_{n-1}	a_{n-2}	$\cdots\cdots\cdots$	a_1	a_0
	b_{n-1}	b_{n-2}	$\cdots\cdots\cdots$	b_1	b_0
c_n	c_{n-1}	c_{n-2}	$\cdots\cdots\cdots$	c_1	c_0
	s_{n-1}	s_{n-2}	$\cdots\cdots\cdots$	s_1	s_0

	a_{n-1}	a_{n-2}	$\cdots\cdots\cdots$	a_1	a_0
	b_{n-1}	b_{n-2}	$\cdots\cdots\cdots$	b_1	b_0
					c_0
c_n	s_{n-1}	s_{n-2}	$\cdots\cdots\cdots$	s_1	s_0

2's complement signed addition and subtraction

- 2's complement signed addition

$$\{c_n, S\} = A + B = A + B + \textcolor{red}{0}$$

	a_{n-1}	a_{n-2}	···········	a_1	a_0
	b_{n-1}	b_{n-2}	···········	b_1	b_0
	c_{n-1}	c_{n-2}	···········	c_1	$\textcolor{red}{0}$
c_n	s_{n-1}	s_{n-2}	···········	s_1	s_0

- 2's complement signed subtraction - via a transformed addition

$$\{c_n, S\} = A - B = A + \overline{B} + \textcolor{red}{1}$$

	a_{n-1}	a_{n-2}	···········	a_1	a_0
	$\overline{b_{n-1}}$	$\overline{b_{n-2}}$	···········	$\overline{b_1}$	$\overline{b_0}$
	c_{n-1}	c_{n-2}	···········	c_1	$\textcolor{red}{1}$
c_n	s_{n-1}	s_{n-2}	···········	s_1	s_0

ADD (addition)	SUB (subtraction)
CF $= c_n$	CF $= \overline{c_n}$
normal carry of a 2's complement addition $A + B = A + B + 0$	inverted carry of a transformed addition $A - B = A + \overline{B} + 1$
$\{c_n, s_{n-1}\}$ $= a_{n-1} + b_{n-1} + c_{n-1}$	$\{c_n, s_{n-1}\}$ $= a_{n-1} + \overline{b_{n-1}} + c_{n-1}$

- In unsigned arithmetic,
  - the carry flag is used to <u>detect</u> *overflow*
  - the carry flag is used to <u>extend</u> *n-bit* result into *(n+1)-bit* result
  - for addition, the carry flag is a carry out
  - for subtraction, the carry flag is a borrow in

- In signed arithmetic,
  - the carry flag is useless
  - the carry flag neither detects overflow nor extends n-bit result

http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt

Carry flag computation (3)

- In unsigned arithmetic,

Addition	CF = 1 means carry out	when Cn = 1
Subtraction	CF = 1 means borrow in	when Cn = 0

  - CF - Carry Flag in x86
  - Cn - the normal carry out
    - the <u>carry out</u> of a 2's complement addition for ADD
    - the <u>carry out</u> of a *transformed* addition for SUB

- In signed arithmetic,
  - the carry flag is useless

`http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt`

TOC: More examples of the carry flag

- Summary I
- Summary II
- Cases for <u>setting</u> the carry flag
- Cases for <u>clearing</u> the carry flag

Summary I

```
-----------------------------------------------------------------------
  unsigned add/sub  | signed addition         signed subtraction | CF OF
-----------------------------------------------------------------------
  1101   (13)       |  1101   (-3)             1101   (-3)        |
 +1110  +(14)  ADD  | +1110  +(-2)  ADD       -0010  -(+2)        |
 -----  -----       | -----  -----            -----  -----        |
 11011   (11) (+16) | 11011   (-5)            11011   (-5)        | 1  0
                    |                                             |
  0011   ( 3)       |  0011   (+3)             0011   (+3)        |
 -1110  -(14)  SUB  | +0010  +(+2)            -1110  -(-2)   SUB  |
 -----  -----       | -----  -----            -----  -----        |
 10101   ( 5) (-16) | 00101   (+5)            00101   (+5)        | 1  0
                    |                                             |
  0011   ( 3)       |  0011   (+3)             0011   (+3)        |
 +0010  +( 2)  ADD  | +0010  +(+2)  ADD       -1110  -(-2)        |
 -----  -----       | -----  -----            -----  -----        |
 00101   ( 5) (+ 0) | 00101   (+5)            00101   (+5)        | 0  0
                    |                                             |
  1101   (13)       |  1101   (-3)             1101   (-3)        |
 -0010  -( 2)  SUB  | +1110  +(-2)            -0010  -(+2)   SUB  |
 -----  -----       | -----  -----            -----  -----        |
 01011   (11) (+11) | 11011   (-5)            11011   (-5)        | 0  0
-----------------------------------------------------------------------
```

Summary II

```
--------------------------------------------------------------------------
 unsigned add/sub  | signed addition        signed subtraction | CF OF
--------------------------------------------------------------------------
  1011   (11)      |  1011    (-5)            1011   (-5)        |
 +1100  +(12)  ADD | +1100   +(-4)  ADD      -0100  -(+4)        |
 -----  -----      | -----   -----           -----  -----        |
 10111  ( 7) (+16) | 10111    (+7)           10111   (+7)        | 1  1
                   |                                              |
  0101   ( 5)      |  0101    (+5)            0101    (+5)        |
 -1100  -(12)  SUB | +0100   +(+4)           -1100  -(-4)  SUB   |
 -----  -----      | -----   -----           -----  -----        |
 11001  ( 9) (-16) | 01001    (-7)           01001   (-7)        | 1  1
                   |                                              |
  0101   ( 5)      |  0101    (+5)            0101    (+5)        |
 +0100  +( 4)  ADD | +0100   +(+4)  ADD      -1100  -(-4)        |
 -----  -----      | -----   -----           -----  -----        |
 01001  ( 9) (+ 0) | 01001    (-7)           01001   (-7)        | 0  1
                   |                                              |
  1011   (11)      |  1011    (-5)            1011   (-5)        |
 -0100  -( 4)  SUB | +1100   +(-4)           -0100  -(+4)  SUB   |
 -----  -----      | -----   -----           -----  -----        |
 00111  ( 7) (  0) | 10111    (+7)           10111   (+7)        | 0  1
--------------------------------------------------------------------------
```

- unsigned integer overflow (`CF=1` means $+16$)

```
----------------------------------------------------------------------
* unsigned addition   | * signed addition          signed subtraction
----------------------------------------------------------------------
   1101   (13)         |    1101   (-3)               1101   (-3)
  +1110  +(14)   ADD   |   +1110  +(-2)   ADD        -0010  -(+2)
  -----  -----         |   -----  -----              -----  -----
  11011   (11) (+16)   |   11011   (-5)              11011   (-5)
                       |
  CF=1                 |   Cn=1 -> CF=1              Cn=1 -> CF=1
                       |
  CF means 16          |   CF meaningless           CF meaningless
   S = 0000            |    S = 0000                 S = 0000
----------------------------------------------------------------------
* think hand           | * think Cn of the corresponding addition
  addition             |   CF <- Cn  (for unsigned addition)
----------------------------------------------------------------------

* CF=1, S=1011, OF=0 for all three interpretations
```

`http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt`

- unsigned integer overflow (`CF=1` means -16)

```
---------------------------------------------------------------------
* unsigned subtraction |   signed addition      * signed subtraction
---------------------------------------------------------------------
   0011   ( 3)         |    0011   (+3)            0011   (+3)
  -1110  -(14)   SUB   |  +0010  +(+2)           -1110  -(-2)   SUB
  -----  -----         |  -----  -----           -----  -----
  10101   ( 5) (-16)   |  00101   (+5)           00101   (+5)
                       |
  CF=1                 |  Cn=0 -> CF=1           Cn=0 -> CF=1
                       |
  CF means -16         |  CF meaningless         CF meaningless
   S = 0101            |   S = 0101               S = 0101
---------------------------------------------------------------------
* think hand           | * think Cn of the transformed addition
  subtraction          |   CF <- !Cn (for unsigned subtraction)
---------------------------------------------------------------------

* CF=1, S=0101, OF=0 for all three interpretations
```

`http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt`

- unsinged integer overflow (`CF=1` means $+16$)

```
------------------------------------------------------------------
* unsigned addition   | * signed addition       signed subtraction
------------------------------------------------------------------
   1011   (11)        |    1011   (-5)           1011   (-5)
  +1100  +(12)  ADD   |   +1100  +(-4)  ADD     -0100  -(+4)
  -----  -----        |   -----  -----          -----  -----
  10111  ( 7) (+16)   |   10111   (+7)          10111   (+7)
                      |
  CF=1                |   Cn=1 -> CF=1          Cn=1 -> CF=1
                      |
  CF means +16        |   CF meaningless        CF meaningless
   S = 0111           |    S = 0111              S = 0111
------------------------------------------------------------------
 * think hand         | * think Cn of the corresponding addition
   addition           |   CF <- Cn  (for unsigned addition)
------------------------------------------------------------------

* CF=1, S=0111, OF=1 for all three interpretations
```

`http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt`

- unsinged integer overflow (`CF=1` means -16)

```
----------------------------------------------------------------------
* unsigned subtraction |    signed addition      * signed subtraction
----------------------------------------------------------------------
  0101   ( 5)          |    0101   (+5)             0101   (+5)
 -1100  -(12)   SUB    |   +0100  +(+4)            -1100  -(-4)   SUB
 -----  -----          |   -----  -----            -----  -----
 11001   ( 9) (-16)    |   01001   (-7)            01001   (-7)
                       |
 CF=1                  |   Cn=0 -> CF=1            Cn=0 -> CF=1
                       |
 CF means -16          |   CF meaningless          CF meaningless
  S = 1001             |    S = 1001                S = 1001
----------------------------------------------------------------------
* think hand           | * think Cn of the transformed addition
  subtraction          |   CF <- !Cn (for unsigned subtraction)
----------------------------------------------------------------------

* CF=1, S=1001, OF=1 for all three interpretations
```

http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt

- no unsigned integer overflow (`CF=0`)

```
---------------------------------------------------------------------
* unsigned addition    | * signed addition          signed subtraction
---------------------------------------------------------------------
   0011   ( 3)         |    0011    (+3)              0011    (+3)
  +0010  +( 2)   ADD   |   +0010   +(+2)   ADD       -1110   -(-2)
  -----  -----         |   -----   -----             -----   -----
  00101   ( 5) (+ 0)   |   00101    (+5)             00101    (+5)
                       |
  CF=0                 |   Cn=0 -> CF=0              Cn=0 -> CF=0
                       |
  CF means 0           |   CF meaningless           CF meaningless
   S = 0101            |    S = 0101                 S = 0101
---------------------------------------------------------------------
* think hand           | * think Cn of the corresponding addition
  addition             |   CF <- Cn  (for unsigned addition)
---------------------------------------------------------------------

* CF=0, S=0101, OF=0 for all three interpretations
```

`http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt`

- no unsigned integer overflow (`CF=0`)

```
---------------------------------------------------------------------
* unsigned addition    | * signed addition          signed subtraction
---------------------------------------------------------------------
  1101   (13)          |   1101   (-3)              1101   (-3)
 -0010  -( 2)   SUB    |  +1110  +(-2)             -0010  -(+2)   SUB
 -----  -----          |  -----  -----             -----  -----
 11011   (11) (-16)    |  11011   (-5)             11011   (-5)
                       |
 CF=0                  |  Cn=0 -> CF=0             Cn=0 -> CF=0
                       |
 CF means 0            |  CF meaningless           CF meaningless
  S = 1011             |   S = 1011                 S = 1011
---------------------------------------------------------------------
* think hand           | * think Cn of the corresponding addition
  subtraction          |   CF <- Cn  (for unsigned addition)
---------------------------------------------------------------------

* CF=0, S=1011, OF=0 for all three interpretations
```

http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt

- no unsinged integer overflow (`CF=0`)

```
--------------------------------------------------------------------
* unsigned addition    | * signed addition        signed subtraction
--------------------------------------------------------------------
  0101   ( 5)          |    0101   (+5)            0101   (+5)
 +0100  +( 4)  ADD     |   +0100  +(+4)  ADD      -1100  -(-4)
 -----  -----          |   -----  -----           -----  -----
 01001   ( 9) (+ 0)    |   01001   (-7)           01001   (-7)
                       |
 CF=0                  |   Cn=0 -> CF=0           Cn=0 -> CF=0
                       |
 CF means +0           |   CF meaningless         CF meaningless
  S = 1001             |    S = 1001               S = 1001
--------------------------------------------------------------------
 * think hand          | * think Cn of the corresponding addition
   addition            |   CF <- Cn  (for unsigned addition)
--------------------------------------------------------------------

 * CF=0, S=1001, OF=1 for all three interpretations
```

http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt

- no unsinged integer overflow (CF=0)

```
-----------------------------------------------------------------------
* unsigned subtraction |    signed addition       * signed subtraction
-----------------------------------------------------------------------
   1011   (11)         |    1011   (-5)              1011   (-5)
  -0100  -( 4)   SUB   |   +1100  +(-4)             -0100  -(+4)   SUB
  -----  -----         |   -----  -----             -----  -----
  00111   ( 7) (  0)   |   10111   (+7)             10111   (+7)
                       |
  CF=0                 |   Cn=1 -> CF=0             Cn=1 -> CF=0
                       |
  CF means  0          |   CF meaningless          CF meaningless
   S = 0111            |    S = 0111                S = 0111
-----------------------------------------------------------------------
* think hand           | * think Cn of the transformed addition
  subtraction          |   CF <- !Cn (for unsigned subtraction)
-----------------------------------------------------------------------

* CF=0, S=0111, OF=1 for all three interpretations
```

http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt