# The Carry Flag

Young W. Lim

2024-11-09 Sat

Young W. Lim The Carry Flag 2024-11-09 Sat 1/59

#### Outline

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

#### Based on

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

I, the copyright holder of this work, hereby publish it under the following licenses: GNU head Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled GNU Free Documentation License.

CC BY SA This file is licensed under the Creative Commons Attribution ShareAlike 3.0 Unported License. In short: you are free to share and make derivative works of the file under the conditions that you appropriately attribute it, and that you distribute it only under a license compatible with this one.

Young W. Lim The Carry Flag 2024-11-09 Sat 3 / 59

## 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

4 / 59

### 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

5 / 59

Young W. Lim The Carry Flag 2024-11-09 Sat

## 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 <a href="OxFFFFB7D0">OxFFFFB7D0</a>
- 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

6 / 59

# Subtracting 0x0000618D from 0x0000195D (1)

a subtraction example

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

• 2's complement

```
\begin{array}{ccc} \texttt{0x0000618D} & \leftrightarrow & \texttt{0xFFFF9E73} \\ \texttt{0xFFFFB7D0} & \leftrightarrow & \texttt{0x00004830} \end{array}
```

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

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

# Subtracting 0x0000618D from 0x0000195D (2)

• 2's complement

• interpreting 0xFFFF9E73

as an unsigned number	+0xFFFF9E73	$+4294942323_{10}$
as a signed number	-0x0000618D	-24973 <sub>10</sub>

• interpreting 0xFFFFB7D0 = 0x0000195D - 0x0000618D

as an unsigned number	+0xFFFFB7D0	+429494881610
as a <mark>signed</mark> number	-0x00004830	-18480 <sub>10</sub>

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

#### Signed and unsigned integer arithmetic examples

 $\bullet$  0x0000195D - 0x0000618D = 0xFFFFB7D0

```
unsigned subtraction
                      0 \times 0000195D - 0 \times 0000618D
                                                           = 0xFFFFB7D0
                                                           = 16^8 - 18480 = 4294948816
                       6493 - 24973
                       subtraction by hand
                                                           borrow (CF=1)
                                                           16^8 = 2^{32} = 4294967296
signed subtraction
                       0x0000195D + (-0x0000618D)
                                                           = -0 \times 00004830
                       0x0000195D + 0xFFFF9E73
                                                           = 0xFFFFB7D0
                       6493 - 24973
                                                           = -18480
                       the transformed addition using CF = 1 \leftarrow Cn = 0
                       the 2's complement of subtrahend
                                                           CF = !Cn for subtraction
```

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

#### 2's complement of the subtrahend 0x0000618D

• the 2's complement of  $0x0000618D \rightarrow 0xFFFF9E73$  (=-24973<sub>10</sub>)

ullet the 2's complement of <code>OxFFFF9E73</code> ightarrow <code>Ox0000618D</code> (= +24973 $_{10}$ )

```
0x0000618D 0x0000_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 F 9 E 7 3
```

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

## Interpreting 0xFFFF9E73

- interpreting 0x0xFFFF9E73
  - as an unsigned Number :  $+0xFFFF9E73 (= +4294942323_{10})$

F	F	F	F	9	Е	7	3
15	15	15	15	9	14	7	3
16 <sup>7</sup>	16 <sup>6</sup>	16 <sup>5</sup>	16 <sup>4</sup>	16 <sup>3</sup>	16 <sup>2</sup>	16 <sup>1</sup>	16 <sup>0</sup>

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

0	0	0	0	6	1	8	D
0	0	0	0	6	1	8	13
16 <sup>7</sup>	16 <sup>6</sup>	16 <sup>5</sup>	16 <sup>4</sup>	$16^{3}$	16 <sup>2</sup>	$16^{1}$	16 <sup>0</sup>

11 / 59

### Unsigned subtraction 0x0000195D - 0x0000618D (1)

• 0x0000195D - 0x0000618D : unsigned subtraction

```
subtraction by hand
```

Young W. Lim The Carry Flag 2024-11-09 Sat 12 / 59

## Unsigned subtraction 0x0000195D - 0x0000618D (2)

- 0x0000195D 0x0000618D : unsigned subtraction subtraction by hand
  - unsigned integer overflow is indicated by the carry flag (CF)
  - a borrow is indicated by the carry flag (CF=1)
    - whenever an unsigned integer overflow happened
    - A B, when A < B, for non-negative integers A, B

Young W. Lim The Carry Flag 2024-11-09 Sat 13 / 59

## Signed subtraction 0x0000195D + (- 0x0000618D) (1)

• 0x0000195D + (-0x0000618D) : signed subtraction the *transformed addition* using the 2's complement of subtrahend

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

Young W. Lim The Carry Flag 2024-11-09 Sat 14 / 59

## Signed subtraction 0x0000195D + (- 0x0000618D) (2)

- 0x0000195D + (-0x0000618D) : signed subtraction the *transformed addition* uses the 2's complement of a subtrahend
  - signed integer overflow 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-fe

#### 2's complement of the subtraction result 0xFFFFB7D0

- $\bullet$  0xFFFFB7D0 = 0x0000195D 0x0000618D
- the 2's complement of 0xFFFFB7D0: 0x00004830 (= +18480<sub>10</sub>) F F F F B 7 D 0 0xFFFFB7D0  $0x1111_1111_1111_1111_1111_1101_0111_1101_0000$  0x0000482F  $0x0000_0000_0000_0000_0100_1000_0010_1111$  (1's complement) 0x00004830  $0x0000_0000_0000_0000_0100_1000_0011_0000$  (2's complement)  $0x0000_0000_0000_0000_0000_0100_0001_0000$  (2's complement)
- the 2's complement of 0x00004830: 0xFFFFB7D0 (= -18480<sub>10</sub>) 0 0 0 0 4 8 3 0 0x00004830 0x0000\_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)

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

- (ロ) (個) (E) (E) (E) (9(C)

16 / 59

#### Interpreting the subtraction result 0xFFFFB7D0

- interpreting 0xFFFFB7D0 = 0x0000195D 0x0000618D
  - as an unsigned Number :  $+0xFFFFB7D0 (= +4294948816_{10})$

F	F	F	F	В	7	D	0
15				11			0
16 <sup>7</sup>	16 <sup>6</sup>	16 <sup>5</sup>	16 <sup>4</sup>	16 <sup>3</sup>	16 <sup>2</sup>	16 <sup>1</sup>	16 <sup>0</sup>

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

0	0	0	0	4	8	3	0
0	0	0	0	4	8	3	0
16 <sup>7</sup>	16 <sup>6</sup>	$16^{5}$	16 <sup>4</sup>	$16^{3}$	16 <sup>2</sup>	$16^{1}$	16 <sup>0</sup>

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

17 / 59

Young W. Lim The Carry Flag 2024-11-09 Sat

# Summary of signed and unsigned subtractions (1)

- 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 subtrahend
  - the same result : 0xFFFFB7D0 (the same bit pattern)
    - interpreting as a unsigned integer 4294948816<sub>10</sub>
       0xFFFFB7D0 with a borrow (CF=1) 2<sup>32</sup> 18480
    - interpreting as a signed integer -18480<sub>10</sub>
       -0x00004830 (meaningless CF=1)

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

18 / 59

# Summary of signed and unsigned subtractions (2)

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

Young W. Lim The Carry Flag 2024-11-09 Sat 19 / 59

# Carry flag (CF) and unsigned integer overflows

- 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),

Young W. Lim The Carry Flag 2024-11-09 Sat 20 / 59

## Overflow flag (OF) and signed integer overflows

- 0x0000195D + (-0x0000618D) : signed subtraction
  - there is no signed integer overflow the overflow flag won't be set (0F=0)
  - signed overflow occurrs in the transformed addition,
    - two *positive* numbers are added and the result is a *negative*,  $(P + P \rightarrow N)$ , or
    - two *negative* numbers are added and the result is a *positive*,  $(N + N \rightarrow P)$

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

- 4 ロ ト 4 個 ト 4 重 ト 4 重 ト 9 Q ()

21 / 59

### 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

22 / 59

Young W. Lim The Carry Flag 2024-11-09 Sat

# 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

### Carrys in unsigned arithmetics

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

## Carrys in signed arithmetics

- <u>carry out</u> in a <u>signed</u> <u>addition</u> / <u>subtraction</u> 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  $C_n$

Young W. Lim The Carry Flag 2024-11-09 Sat 25 / 59

# Signed and unsigned arithmetics (1)

	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 <sub>n</sub> relation	Carry $CF = C_n$ for addition instructions	Borrow CF = $!C_n$ for subtraction instructions

Young W. Lim The Carry Flag 2024-11-09 Sat 26 / 59

# Signed and unsigned arithmetics (2)

```
CF (Carry Flag)

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

Cn (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

Young W. Lim The Carry Flag 2024-11-09 Sat 27 / 59

# Signed and unsigned arithmetics (3)

- signed and unsigned additions (or subtractions)
  - share the same *n-bit* result
  - a borrow (CF) in an unsigned subtraction is the inverted value of a carry out (C<sub>n</sub>) in a signed subtraction

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

Young W. Lim The Carry Flag 2024-11-09 Sat 28 / 59

# Signed and unsigned arithmetics (4)

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

unsign	ed add,	/sub	I	signed add	ition	signed sub	tractio	n 
0011 -1110	(3) -(14)	SUB		0011 +0010	(+3) +(+2)	0011 -1110	(+3) -(-2)	SUB
			-					
10101	(5)	(-16)	-	00101	(+5)	00101	(+5)	
CF=1				Cn=0		Cn=0		
1101	(13)		- 1	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

# Using the carry flag as a borrow (1)

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

30 / 59

Young W. Lim The Carry Flag 2024-11-09 Sat

# Using the carry flag as a borrow (2)

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

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

4□▷ 4ਾਊਾ▷ 4 ½ ▷ 4½ ▷ ½ ▷ ½ ▷ √

31 / 59

## Using the carry flag as a borrow (3)

- 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

 $\verb|https://stackoverflow.com/questions/47333458/assembly-x86-64-setting-carry-flag-free formula for the control of the contro$ 

## Using the carry flag as a borrow (4)

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

## TOC Rules for the carry flag

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

Young W. Lim The Carry Flag 2024-11-09 Sat 34 / 59

# 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

# The 1st rule for setting the carry flag

- ① CF = 1 : carry in unsigned addition
  - hand addition rule
  - the <u>carry flag</u> is set
    if the <u>addition</u> of two <u>unsigned</u> numbers causes
    a <u>carry</u> out of the most significant bits added.
  - unsigned integer overflow in unsigned addition

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

Young W. Lim The Carry Flag 2024-11-09 Sat 36 / 59

# The 2nd rule for setting the carry flag

- ② CF = 1 : borrow in unsigned subtraction
  - hand subtraction rule
  - the <u>carry flag</u> is <u>also</u> set
    if the <u>subtraction</u> of two <u>unsigned</u> numbers requires
    a <u>borrow</u> into the most significant bits subtracted.
  - unsigned integer overflow in unsigned subtraction

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

Young W. Lim The Carry Flag 2024-11-09 Sat 37 / 59

# Cases for setting the carry flag (1)

- **1** CF = 1 : carry in unsigned addition
  - all three interpretations have the same CF=1, the same S=0000

signed addition	signed subtraction
0111 (+7)   +1001 +(-7)	0111 (+7) -0111 -(+7)
   10000 (0)   Cn=1 -> CF=1	10000 ( 0) Cn=1 -> CF=1
CF meaningless   S = 0000	CF meaningless S = 0000
* think Cn of the c	corresponding addition
	0111 (+7)   +1001 +(-7)     10000 (0)   Cn=1 -> CF=1     CF meaningless   S = 0000

 $\verb|http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt|$ 

# Cases for setting the carry flag (2)

- ② CF = 1 : borrow in unsigned subtraction
  - all three interpretations have the same CF=1, the same S=0000

unsigned subtraction	I	signed addition	signed subtraction
0111 (7) -1001 -(9)		0111 (+7) +0111 +(+7)	0111 (+7) -1001 -(-7)
	!		
11110 (14) -(16)		01110 (14)	01110 (14)
CF=1	1	Cn=0 -> CF=1	Cn=0 -> CF=1
CF means -16	- 1	CF meaningless	CF meaningless
S = 1110	 	S = 1110	S = 1110
* think hand subtraction	   	* think Cn of the corr	esponding addition

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

Young W. Lim The Carry Flag 2024-11-09 Sat 39 / 59

#### Computing CF in unsigned additions and subtractions

- Computing CF in an unsigned addition
  - do the signed 2's complement addition
  - Cn is the carry out
  - CF  $\leftarrow$  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

Young W. Lim The Carry Flag 2024-11-09 Sat 40 / 59

## 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

Young W. Lim The Carry Flag 2024-11-09 Sat 41/59

#### Addend and augend in a *n*-bit addition

n	bits	addened	Α	$\{a_{n-1}, a_{n-2}, \cdots, a_1, a_0\}$
n	bits	augend	В	$\{b_{n-1},b_{n-2},\cdots,b_1,b_0\}$
(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

(ロ) (個) (意) (意) (意) (の)

42 / 59

#### Full adder (hardware) operation

full adder operation in the  $i^{th}$  bit position

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

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

43 / 59

Young W. Lim The Carry Flag 2024-11-09 Sat

#### Internal and external carry bits

$$\begin{array}{lll} \text{external carrys} & \textit{c}_n \text{ output, } \textit{c}_0 \text{ input} \\ \text{internal carrys} & \{c_{n-1}, c_{n-2}, \cdot \cdot \cdot \cdot \cdot \cdot, c_2, c_1\} & \text{output / input} \\ \text{sum bits} & \{s_{n-1}, s_{n-2}, \cdot \cdot \cdot \cdot \cdot \cdot, s_1, s_0\} & \text{output} \end{array}$$

#### 2's complement signed addition and subtraction

2's complement signed addition

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

	$a_{n-1}$	$a_{n-2}$	 $a_1$	$a_0$
	$b_{n-1}$	$b_{n-2}$	 $b_1$	$b_0$
	$c_{n-1}$	$c_{n-2}$	 $c_1$	0
Cn	$s_{n-1}$	$s_{n-2}$	 $s_1$	<i>s</i> <sub>0</sub>

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

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

2024-11-09 Sat 45 / 59

# Carry flag computation (1)

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 $A - B = A + \overline{B} + 1$
${c_n, s_{n-1}}$ = $a_{n-1} + b_{n-1} + c_{n-1}$	${c_n, s_{n-1}}$ = $a_{n-1} + \overline{b_{n-1}} + c_{n-1}$

 $\verb|https://www.csie.ntu.edu.tw/^cyy/courses/assembly/12fall/lectures/handouts/lec14_states/assembly/12fall/lectures/handouts/lec14_states/assembly/12fall/lectures/handouts/lec14_states/assembly/12fall/lectures/handouts/lec14_states/assembly/12fall/lectures/handouts/lec14_states/assembly/12fall/lectures/handouts/lec14_states/assembly/12fall/lectures/handouts/lec14_states/assembly/12fall/lectures/handouts/lec14_states/assembly/12fall/lectures/handouts/lec14_states/assembly/12fall/lectures/handouts/lec14_states/assembly/12fall/lectures/handouts/lec14_states/assembly/12fall/lectures/handouts/lec14_states/assembly/12fall/lectures/handouts/lec14_states/assembly/12fall/lectures/handouts/lec14_states/assembly/lec14_states/a$ 

## Carry flag computation (2)

- In unsigned arithmetic,
  - the carry flag is used to detect overflow
  - the carry flag is used to extend 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

47 / 59

Young W. Lim The Carry Flag 2024-11-09 Sat

## 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 <u>ADD</u>
  - the carry out of a transformed addition for SUB
- In signed arithmetic,
  - the carry flag is useless

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

48 / 59

Young W. Lim The Carry Flag 2024-11-09 Sat

## 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

Young W. Lim The Carry Flag 2024-11-09 Sat 49 / 59

#### Summary I

unsign	ed add/	/sub	signed	additio	 n	signed	subtrac	tion	   CF	0F
1101 +1110	(13) +(14)	ADD		(-3) +(-2)			(-3) -(+2)		   	
11011	(11)	(+16)	11011	(-5)		11011	(-5)		   1 	0
0011 -1110	(3) -(14)	SUB		(+3) +(+2)		0011 -1110	(+3) -(-2)	SUB	 	
10101	(5)	(-16)	   00101 	(+5)		00101	(+5)		   1 	0
0011 +0010	(3) +(2)	ADD	0011	(+3) +(+2)	ADD		(+3) -(-2)		i I	
00101	(5)	(+ 0)	   00101	(+5)		00101	(+5)		   0 	0
1101 -0010	, ,	SUB	1101   +1110	(-3) +(-2)		1101 -0010	(-3) -(+2)	SUB	 	
01011	(11)	(+11)	1 11011	(-5)		11011	(-5)		I I 0	0

## Summary II

unsign	ed add,	/sub	signed	additio	n	signed	subtrac	tion	CF	OF
1011	(11)		1011	(-5)		1011	(-5)		 	
+1100	+(12)	ADD	+1100	+(-4)	ADD	-0100	-(+4)		l	
10111	(7)	(+16)	1 10111	(+7)		10111	(+7)		   1	1
10111	( 1)	(110)	10111	(11)		10111	(17)		1	_
0101	(5)		0101	(+5)		0101	(+5)		l	
-1100	-(12)	SUB	+0100	+(+4)		-1100	-(-4)	SUB	!	
11001	(9)	( 16)	01001	(-7)		01001	(-7)		   1	1
11001	( 9)	(-10)	1 01001	(-1)		01001	(-1)		1 1	1
0101	(5)		0101	(+5)		0101	(+5)		İ	
+0100	+(4)	ADD	+0100	+(+4)	ADD	-1100	-(-4)		1	
01001	( 0)	(, 0)		(7)		04004	(7)			,
01001	(9)	(+ 0)	01001	(-7)		01001	(-7)		0 	1
1011	(11)		1011	(-5)		1011	(-5)		i	
-0100	-(4)	SUB	+1100	+(-4)		-0100	-(+4)	SUB	I	
		,								
00111	(7)	( 0)	10111	(+7)		10111	(+7)		0	1

#### Cases for setting the carry flag (1) CF=1, OF=0

• unsigned integer overflow (CF=1 means +16)

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

<sup>\*</sup> CF=1, S=1011, OF=0 for all three interpretations

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

#### Cases for setting the carry flag (2) CF=1, OF=0

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

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

<sup>\*</sup> CF=1, S=0101, OF=0 for all three interpretations

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

# Cases for setting the carry flag (3) CF=1, OF=1

• unsinged integer overflow (CF=1 means +16)

* unsigned addition	* signed addition	signed subtraction
1011 (11) +1100 +(12) ADD	1011 (-5)   +1100 +(-4) ADD	1011 (-5) -0100 -(+4)
10111 (7) (+16)	   10111 (+7) 	10111 (+7)
CF=1	Cn=1 -> CF=1	Cn=1 -> CF=1
CF means +16 S = 0111	CF meaningless S = 0111	CF meaningless S = 0111
* think hand addition	* think Cn of the corresp   CF <- Cn (for unsigned	0

<sup>\*</sup> CF=1, S=0111, OF=1 for all three interpretations

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

Young W. Lim The Carry Flag 2024-11-09 Sat 54 / 59

## Cases for setting the carry flag (4) CF=1, OF=1

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

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

<sup>\*</sup> CF=1, S=1001, OF=1 for all three interpretations

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

Young W. Lim The Carry Flag 2024-11-09 Sat 55 / 59

## Cases for clearing the carry flag (1) CF=0, OF=0

• no unsigned integer overflow (CF=0)

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

 $<sup>\</sup>ast$  CF=0, S=0101, OF=0 for all three interpretations

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

## Cases for clearing the carry flag (2) CF=0, OF=0

• no unsigned integer overflow (CF=0)

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

<sup>\*</sup> CF=0, S=1011, OF=0 for all three interpretations

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

Young W. Lim The Carry Flag 2024-11-09 Sat 57 / 59

## Cases for clearing the carry flag (3) CF=0, OF=1

• no unsinged integer overflow (CF=0)

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

<sup>\*</sup> CF=0, S=1001, OF=1 for all three interpretations

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

Young W. Lim The Carry Flag 2024-11-09 Sat 58 / 59

## Cases for clearing the carry flag (4) CF=0, OF=1

• no unsinged integer overflow (CF=0)

1011 (11)		1011 (-5)	1011 (-5)
-0100 -( 4) SUB	-	+1100 +(-4)	-0100 -(+4) SUE
	1		
00111 (7) (0)	 	10111 (+7)	10111 (+7)
CF=0	İ	Cn=1 -> CF=0	Cn=1 -> CF=0
CF means 0	Ì	CF meaningless	CF meaningless
S = 0111	1	S = 0111	S = 0111
think hand	 	* think Cn of the tran	nsformed addition  gned subtraction)

<sup>\*</sup> CF=0, S=0111, OF=1 for all three interpretations

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