Overflow flag

1.Signed

2.Unsigned

Signed number that can be positive or negative.

Unsigned number that can only be Positive

For 8 Bit register

Signed number: -128 to -1 AND 0 to 127

Unsigned number: 0 to 255

DEC	hex
53	35
+25	+19
78	4E

53=0 0 1 1 0 1 0 1

25=0 0 0 1 1 0 0 1

0 1 1 0 0 0 1

0 0 1 1 0 1 0 1

0 0 0 1 1 0 0 1

0 1 0 0 1 1 1 0 (78)

CF=0 (no carry bit 6 to 7 and 7 to c)

OF=0 (78 is between -128 to +127)

DEC	hex
53	35
+91	+5B
144	90

53=0 0 1 1 0 1 0 1

91= 0 1 0 1 1 0 1 1

1 1 1 1 1 1 1

0 0 1 1 0 1 0 1

0 1 0 1 1 0 1 1

1 0 0 1 0 0 0 0 (144) [NEGATIVE]

CF=0 (no carry bit 6 to 7 and 7 to c) {carry from bit 6 to 7 but no carry from 7 to c }

OF=1 (we add two positive number but result negative, so OF =1 because 144 is outside of -128 to +127)

53= 0 0 1 1 0 1 0 1

45= 1 1 0 1 0 0 1 1

-

DEC	hex
53	35
-45	D3 (ignore carry)
8	108

1 1 1 0 1 1 1

0 0 1 1 0 1 0 1

1 1 0 1 0 0 1 1

10 0 0 0 1 0 0 0

CF =1 (carry bit 6 to 7 and 7 to c)

OF=0 (we add two positive number got positive, so OF =0 because result 8 is between -128 to +127) {carry from bit 6 to 7 and carry from bit 7 to C OVERFLOW always 0 }

.

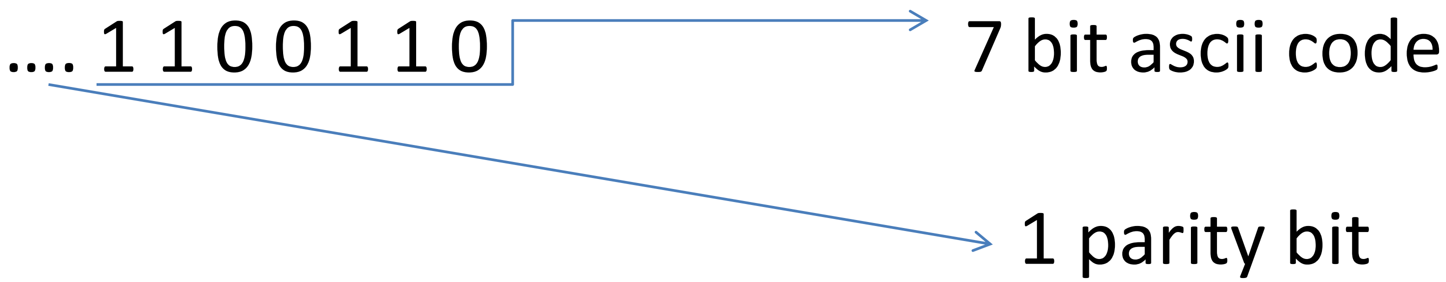
DEC	hex
-98	9E
-45	+D3 (ignore carry)
-143	171

```
  0 0 1 1 1 1 0
  1 0 0 1 1 1 1 0
  1 1 0 1 0 0 1 1
 ----------------
1 0 1 1 1 0 0 0 1
```

CF=1 (no carry bit 6 to 7 but there is a carry from bit 7 to C)

OF=1 (we add two negative number got positive !!! That must not be correct. so OF set to 1, because result 171 is outside the range 128 to +127) {no carry from bit 6 to 7 and carry from bit 7 to C OVERFLOW always 1 }

Parity Flag (PF)



.... 1 1 0 0 1 1 0

7 bit ascii code

1 parity bit

Even parity: even number of 1 the parity bit is 1

Odd parity: odd number of 1 the parity bit is 0

26 (11010) parity =0 {number of 1 odd }

102 (1100110) parity = 1 {number of 1 even }

Auxiliary Flag (AX)

BCD →Binary coded number

Auxiliary Flag (AX)

No carry

3 1

+3 4

6 5

31= 0 0 1 1 0 0 0 1

34= 0 0 1 1 0 1 0 0

0 0 1 1 0 0 0 1

0 0 1 1 0 1 0 0

0 1 1 0 0 1 0 1 (65)

Not defined

3 4

+3 7

7 1

34= 0 0 1 1 0 1 0 0

37=0 0 1 1 0 1 1 1

1 1 0 1 0 0 0

0 0 1 1 0 1 0 0

0 0 1 1 0 1 1 1

0 1 1 0 1 0 1 1 (6B)

1 to 9 . B is not recognize

Difference 06

Auxiliary Flag (AX)

2 8

+2 9

5 7

0 1 0 1 0 0 0 0

0 0 1 0 1 0 0 0

0 0 1 0 1 0 0 1

0 1 0 1 0 0 0 1 (51)

Difference 06

28 = 0 0 1 0 1 0 0 0

29 = 0 0 1 0 1 0 0 1

Auxiliary Flag (AX)

When AC=1 we should add 06 to the result

We will need to add 6.

How compiler do this ?

Auxiliary Flag (AX)

DAA → Decimal Adjust Accumulator

If it AC= 1 ;then compiler add 06 to the answer

when AC=0 ;then compiler do not add 06 to the answer

when nibble is > 09 it again add 06 to the answer

DAA only works on ADD,ADDC

Not works in SUB,SUBB,MUL,DIV

AX = 1 when there is a carry out from bit number 3 to bit number 4 on addition or borrow into bit number 3 during subtraction

Zero flag

The zero flag is used to show us whether the:

. Value of the indicated operation is zero

.A subtraction resulted in a zero,

.A decrement ended up being zero,

.Or there was zero difference between the two values compared

Sign Flag (SF)

SF = 1 when the MSB of a signed number is 1.

10000001

here SF = 1

2’s compliment

Equation		binary	signed
2	=	0 0 0 0 0 0 1 0	2
1’s compliment		1 1 1 1 1 1 0 1	
ADD 1		0 0 0 0 0 0 0 1	
2’s compliment		1 1 1 1 1 1 1 0	-2

•

A 16 bit number system can be used to create 65536 combinations (from 0 to 65535):

the first 32768 combinations (0 to 32767) represent positive numbers and next 32768 combinations (32768 to 65536) represent negative numbers.