Number Systems

BME 360 Lecture Notes *Ying Sun*

Unsigned Binary and 2's Complement

The most important concept is that in a computer system the unsigned binary representation and the 2's complement representation coexist. For a given binary number you can interpret it as either an unsigned binary number (positive integer) or a 2's complement number (positive or negative integer). The most commonly used numbers in the PIC C program are:

```
unsigned char     // 8-bit unsigned binary number, range: 0 – 255
signed char       // 8-bit 2's complement number, range: -128 – 127
unsigned int      // 16-bit unsigned binary number, range: 0 – 65535
int               // 16-bit 2's complement number, range: -32768 –  32767
```

We use the 8-bit number as an example below:

binary	hexadecimal	unsigned binary representation (decimal)	2's complement representation (decimal)
0000 0000	0 0	0	0
0000 0001	0 1	1	1
0000 0010	0 2	2	2
------ ------	--- ---	---	---
0111 1111	7 F	127	127
1000 0000	8 0	128	-128
1000 0001	8 1	129	-127
------ ------	--- ---	---	---
1111 1101	F D	253	-3
1111 1110	F E	254	-2
1111 1111	F F	255	-1

2's complement Representation

There are only two symbols (0 and 1) in a computer system. There is no symbol for the negative sign (–). There are at least three ways to represent a negative number: 2's complement, 1's complement, and sign-magnitude. The 2's complement is arguably the best representation and is used by all existing computer systems. With the 2's complement representation, the most significant bit it the sign bit. If the most significant bit is 0, the number is positive and its value is the same as that of the unsigned binary. If the most significant bit is 1, the number is negative and the magnitude is obtained by *taking the 2's complement* of the number. *Taking the 2's complement* is the operation of *negating* the number. In other words, taking the 2's complement of a negative number makes the number positive, which is the magnitude of the original number.

Example 1

With unsigned binary representation, hexadecimal number $A3 is 163 in decimal.

$$(A3)_{16} = 10\times16^{1}+3\times16^{0}=160+3=163$$

With 2's complement representation, hexadecimal number $A3 is -93 in decimal.

$(A3)_{16} = (1010\ \ 0011)_2$. It is a negative number because the most significant bit is 1.

Take the 1's complement: $1010\ \ 0011 \rightarrow 0101\ \ 1100$.

Add 1 to get the 2's complement: $0101\ \ 1100 + 1 = 0101\ \ 1101 = (5D)_{16} = 5\times16+13 = 93$.

Status Register (page 52 of the PIC18F4525 Datasheet)

The PIC processor use the Status Register to keep track of an arithmetic/logic operation.

REGISTER 4-2: STATUS REGISTER

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	N	OV	Z	DC	C
bit 7							bit 0

bit 7-5 **Unimplemented:** Read as '0'

bit 4 **N:** Negative bit
This bit is used for signed arithmetic (2's complement). It indicates whether the result was negative (ALU MSB = 1).
1 = Result was negative
0 = Result was positive

bit 3 **OV:** Overflow bit
This bit is used for signed arithmetic (2's complement). It indicates an overflow of the 7-bit magnitude, which causes the sign bit (bit7) to change state.
1 = Overflow occurred for signed arithmetic (in this arithmetic operation)
0 = No overflow occurred

bit 2 **Z:** Zero bit
1 = The result of an arithmetic or logic operation is zero
0 = The result of an arithmetic or logic operation is not zero

bit 1 **DC:** Digit carry/$\overline{\text{borrow}}$ bit
For `ADDWF`, `ADDLW`, `SUBLW`, and `SUBWF` instructions
1 = A carry-out from the 4th low order bit of the result occurred
0 = No carry-out from the 4th low order bit of the result

Note: For $\overline{\text{borrow}}$, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either the bit 4 or bit 3 of the source register.

bit 0 **C:** Carry/$\overline{\text{borrow}}$ bit
For `ADDWF`, `ADDLW`, `SUBLW`, and `SUBWF` instructions
1 = A carry-out from the Most Significant bit of the result occurred
0 = No carry-out from the Most Significant bit of the result occurred

Note: For $\overline{\text{borrow}}$, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either the high or low order bit of the source register.

After an addition or subtraction operation, the N Ov Z C bits are set according to:

N = 1, if the result is negative; 0 otherwise.

Ov = 1, if there is a 2's complement overflow, 0 otherwise. (See below.)

Z = 1, if the result is zero, 0 otherwise.

C = 1, if there is a carry after ADD or if there is no borrow after SUBTRACT, 0 otherwise.

The Ov bit is set to 1 for one of the following four conditions:

1. a positive number + a positive number = a negative number
2. a negative number + a negative number = a positive number
3. a positive number – a negative number = a negative number
4. a negative number – a positive number = a positive number

2's complement overflow never occurs (Ov = 0) for the following two conditions:

1. a positive number + a negative number (= either a positive number or a negative number)
2. a negative number - a negative number (= either a positive number or a negative number)

Example 2

Hex operation	Unsigned binary	2's compliment	N	Ov	Z	C
\$7F + \$01 = \$80	127 + 1 = 128	127 + 1 = -128	1	1	0	0
\$FF + \$01 = \$00	255 + 1 = 0	-1 + 1 = 0	0	0	1	1
\$54 + \$1A = \$6E	84 + 26 = 110	84 + 26 = 110	0	0	0	0
\$E5 + \$DC = \$C1	229 + 220 = 193	-27 + (-36) = -63	1	0	0	1
\$83 - \$82 = \$01	131 – 130 = 1	-125 – (-126) = 1	0	0	0	1
\$82 - \$83 = \$FF	130 – (131) = 255	-126 – (-125) = -1	1	0	0	0
\$54 – \$BF = \$A5	84 – 191 = 165	84 – (-65) = -91	1	1	0	0

C = carry/$\overline{\text{borrow}}$. Why?

The processor does subtraction by adding the 2's complement.

\$83 - \$82 = \$01 ⇒ \$83 + \$7E = \$01 Thus, C = 1.

\$82 - \$83 = \$FF ⇒ \$82 + \$7D = \$FF Thus, C = 0.