CSE 260 – Lecture 2

BCD = Binary Coded Decimal

= Also known as 8421 code.

= BCD is not equivalent to Binary.

Decimal	BCD	Decimal	BCD
0	0000	5	0101
1	0001	6	0110
2	0010	7	0111
3	0011	8	1000
4	0100	9	1001

For Example:

$(234)_{10} = (11101010)_2$ ← This is Binary

2 3 4

0010 0011 0100

$(0010\ 0011\ 0100)_{BCD}$ ← This is BCD.

Excess-3 = This is another code scheme

= Here we add 3 to the Decimal Digit & then we convert it to Binary.

= Example: $(3)_{10} = (3+3) = 6 = (0110)_{Excess\ 3}$

$\therefore (3)_{10} = (0110)_{Excess\text{-}3}$

Decimal Digit	BCD 8421	Excess-3
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100
Unused bit combi-nations	1010	0000
	1011	0001
	1100	0010
	1101	1101
	1110	1110
	1111	1111

} This bit aren't used

As we add 3 in the conversion process in Excess-3,

Similarly, we add 5 in Excess 5.

Negative Number Representation

* 3 ways:
  1. Sign & Magnitude
  2. 1's complement
  3. 2's complement

1. Sign & Magnitude:

* We write negative number by writing a minus sign in the front.

$$-(15)_{10} = -(1111)_2$$

* In computer, we assign a sign bit in front.

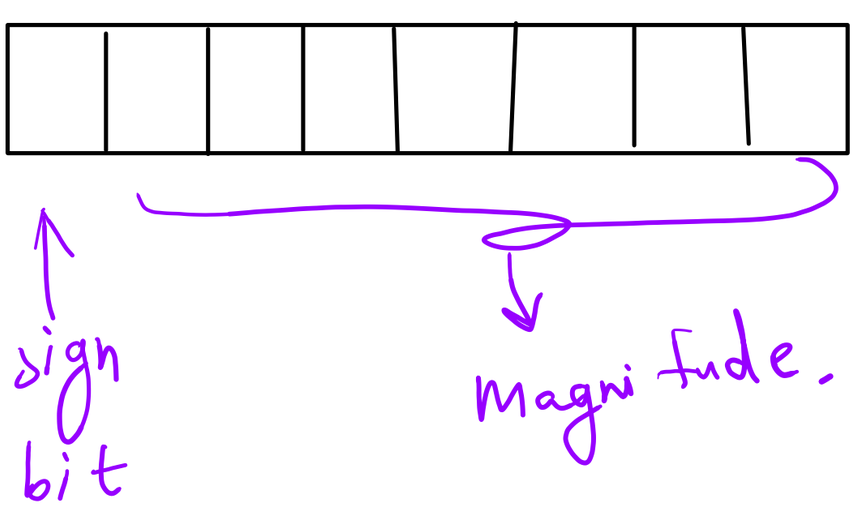
0 = +ve sign = +ve number

1 = -ve " = -ve "

As Memory have a fixed width in Computer, Assigning a sign bit reduces the bits for magnitude. This affects the range, which will be discussed shortly.

Example:

An 8 bit number can have 1 bit sign & 7 bit magnitude.



$13 = 1101$ ← A 4 bit unsigned number

$+13 = 01101$ ← A 5 bit positive "

$-13 = 11101$ ← A 5 bit negative "

In 8 bit,

Range of unsigned Binary number $[0 \rightarrow (2^8 - 1)]_{10}$

$[0 \rightarrow 255]_{10}$

$[00000000 \rightarrow 11111111]_2$

In case of sign & Magnitude,

Largest positive number $= (0\ 1111111)_2 = (+127)_{10} = 2^{(8-1)} - 1$

Largest negative " $= (1\ 1111111)_2 = (-127) = -2^{(8-1)} - 1$

zeros $= (0\ 0000000)_2 = (+0)_{10}$

$(1\ 0000000)_2 = (-0)_{10}$

} Same thing different Representation

∴ Range for 8 bit, $[-2^8 - 1$ to $2^8 - 1]$

∴ Range for n bit, $[(-2^{n-1}-1) \text{ to } (2^{n-1}-1)]$

1's complement:

= Here, positive value remains same.

= In case of negative value, bits are flipped/inversed.

= First bit / MSB still represents sign. 0 = +ve, 1 = -ve

Example:

+7 = 0111		−7 = −(0111)	= (1000)1's comp
+6 = 0110		−6 = −(0110)	= (1001)1's comp.
⋮		⋮	
+1 = 0001		−1 = −(0001)	= (1110)1's comp
+0 = 0000		−0 = −(0000)	= (1111) 1's comp.

* Still 2 representation for 0.

In case of 8 bit 1's comp:

Largest positive number = $(0\ 1111111)_2 = (+127)_{10} = 2^{(8-1)}-1$

Largest negative " = $(1\ 0000000)_2 = (-127) = -2^{(8-1)}-1$

zeros = $(0\ 0000000)_2 = (+0)_{10}$ } same thing different Representation

$(1\ 1111111)_2 = (-0)_{10}$

∴ Range for 8 bit, $[-2^8-1 \text{ to } 2^8-1]$

∴ Range for n bit, $[(-2^{n-1}-1) \text{ to } (2^{n-1}-1)]$

2's complement:

= Positive number remains the same.

= In case of negative number, we

Method 1 {
1. Find out 1's complement
2. Add 1 to get the 2's complement.
}

Method 2 {
= A negative number can also be obtained in 2's complement by,

$$-x = 2^n - x$$
}

Example:

In an 8 bit system, Find out $-(12)_{10}$ in 2's comp representation.

Solve: (Method 1)

$$(-12) = -(00001100)_2$$

$$(-12) = (11110011)_{1's}$$

$$+1$$

$$= (11110100)_{2's}$$

(Method 2)

Here $n = 8, \quad x = 12$

We know, $-x = 2^n - x$

$$\therefore (-12)_{10} = (2^8 - 12)_{10}$$

$$= (256 - 12)_{10}$$

$$= (244)_{10}$$

$$= (11110100)_{2's}$$

[Any method is fine, but remember $-x = 2^n - x$ formula]

- Example: 4-bit signed number (*positive values*)

Value	Sign-and-Magnitude	1s Comp.	2s Comp.
+7	0111	0111	0111
+6	0110	0110	0110
+5	0101	0101	0101
+4	0100	0100	0100
+3	0011	0011	0011
+2	0010	0010	0010
+1	0001	0001	0001
+0	0000	0000	0000

- Example: 4-bit signed number (*negative values*)

Value	Sign-and-Magnitude	1s Comp.	2s Comp.
-0	1000	1111	-
-1	1001	1110	1111
-2	1010	1101	1110
-3	1011	1100	1101
-4	1100	1011	1100
-5	1101	1010	1011
-6	1110	1001	1010
-7	1111	1000	1001
-8	-	-	1000

Note that, 2's comp doesn't has 2 different representation of 0.

This happens because we add +1 in the conversion process.

$\therefore$ Range in 8 bits $(-128)_{10}$ to $(+127)_{10}$

$\therefore$ Range in n bits,

$$\left[-2^{n-1} \text{ to } +(2^{n-1}-1)\right]$$

* MSB of 2's comp number still represents the sign.

Two's comp to binary:

1. To convert a negative 2's comp. we need to flip the bits & add 1.

Ex: $(11100)_{2's} = -(00011 + 1)$

$= -(00100)_2$

$= -4$

Unsigned Number Arithmatic Operation

Similar as we learnt in Lecture 1

Overflow

* When output of addition or subtraction requires more bits to represent properly than allocated memory.

How to find out :

$\Rightarrow$ (+ve) + (+ve) = (+ve) $\longrightarrow$ This is correct. **No Overflow**

(+ve) + (+ve) = (-ve) $\longrightarrow$ Incorrect. Overflow

$\Rightarrow$ (-ve) + (-ve) = (-ve) $\longrightarrow$ No overflow

(-ve) + (-ve) = (+ve) $\longrightarrow$ Overflow

$\therefore$ In case of addition, same sign number should produce same signed output.

different signed number will never cause an overflow.

In case of subtraction, the opposite happens.

* Same signed number will never cause an overflow
* For Different signed number,

if $(+A) - (-B) = A+B$ = +ve, no overflow

= -ve, overflow

if $(-A) - (B) = (-A) + (-B)$ = -ve, no overflow

= +ve, overflow.

2's complement Addition

* In case of two positive number, procedure remains same as binary addition.

Example: (15) + (27) in 7 bit

$\Rightarrow$ $+15 = \underline{0}0001111$ (carries: 1 1 1 1)

$+27 = \underline{0}011011$

$\underline{0}101010 = (\pm 42)$

Note: sign bits are 0. No overflow

(+) + (+) = +

Example: (15) + (27) in 6 bit. Check for overflow:

$+15 = \underline{0}01111$

$+27 = \underline{0}11011$

$\underline{1}01010$

$= -(010101+1)$

$= -(010110)$

$= -(22)$

Here, we added two +ve number and got -ve result. Therefore, overflow.

Another analysis for over flow would be, the range of 6 bit $[-2^{6-1}$ to $2^{6-1}-1] = [-32$ to $31]$ but the result is (+42) which is out of range. So, overflow

2's complement Subtraction

* Subtract 22 from 48 using 2's comp in 8 bits. Check for overflow.

Solve: $48 - 22 = 48 + (-22)$

$+48 = 0110000$

$+48$ in 8 bits $= (00110000)_{2's}$

$22 = 00010110$

$-22 = (11101001)_{1's}$

$+1$

$= (11101010)_{2's}$

$+48 = 00110000$

$-22 = 11101010$

$1\,|\,00011010$

Carry ignore

$= (00011010)_{2's}$

$= +26$

(Ans)

On the above addition,

We added a +ve & a -ve number. So, no overflow.

* Subtract 14 from (-20) in 6 bit. Check for overflow:

Solve: $(-20) - (14) = (-20) + (-14)$

$20 = 010100$

$-20 = (101011)_{1's}$

$= (101100)_{2's}$

$14 = 001110$

$-14 = (110001)_{1's}$

$+1$

$= (110010)_{2's}$

$-20 \rightarrow 101100$

$(+)\ -14 \rightarrow 110010$

$1\,|\,011110$

carry ignore

Here, we added 2 (-ve) number and got 1 positive number.

$\therefore$ Overflow.

* Subtract 14 from (-20) in 7 bit. Check for overflow:

Solve: $(-20)-(14) = (-20)+(-14)$

$20 = 0010100$

$-20 = (1101011)_{1's}$

$\overset{+1}{=}(1101100)_{2's}$

$14 = 0001110$

$-14 = (1110001)_{1's}$

$\overset{+1}{=}(1110010)_{2's}$

$$\begin{array}{r} {\scriptstyle 1\ \ 1\ \ \ \ \ \ \ } \\ 1101100 \\ +\ 1110010 \\ \hline 1\,|\,1011110 \end{array}$$

carry ignore

$= (1011110)_{2's}$

$= -(0100001)$

$\overset{+1}{}$

$= -(0100010)_2$

$= -34$

We got -ve number by adding 2(+ve) numbers.

∴ No overflow.

1's complement Addition & Subtraction

Addition is similar as Binary Addition

Subtraction is done by the following steps:

1. Convert $A-B$ to $A+(-B)$
2. Perform the addition.
3. If the final carry is 1:

   Add 1 to the result

   If the final carry is 0:

   result is a negative number in 1's complement form.

Example: $(1100)_2 - (0101)_2$

Step 1:

$A = (1100)_2 = (1100)_{1's}$

$B = 0101$

$-B = -(0101)_2 = (1010)_{1's}$

Step 2:

$$\begin{array}{r} 1100 \\ +\ 1010 \\ \hline 10110 \end{array} \qquad \begin{array}{l} \rightarrow (12)_{10} \\ \rightarrow (-5)_{10} \\ \rightarrow (6) \end{array}$$

Step 3:

$+1$

$0111 \rightarrow (7)$ ✓

Example 2:

$(0101)_2 - (1100)_2$

$A = 0101 = (0101)_{1's}$

$B = 1100$

$-B = -(1100)_2 = (0011)_{1's}$

$$\begin{array}{r} 0101 \\ 0011 \\ \hline 01000 \end{array} \qquad \begin{array}{l} \rightarrow 5 \\ \rightarrow -12 \end{array}$$

0 → number is -ve in 1's form.

$= (1000)_{1's} = -(0111)$

$= -7$