Signed Magnitude Notation

Wednesday, April 7, 2021 1:54 PM

Unsigned Binary - what you worked with before (ex 0010 as 2)

Signed Magnitude Notation - leftmost bit represents the sign (+/-); other bits represent the magnitude

Ex: Represent +5 using signed magnitude notation (using 4-bits)

sign	Magnitude		
0	1	0	1

+ is 0

Ex: Represent -5 using s.m.n.

sign	Magnitude		
1	1	0	1

-5 is 1101

Ex: Represent 0 using signed magnitude notation.

sign	magnitude			
0	0	0	0	← +0
1	0	0	0	← -0

} two zeros; not ideal

Ex: Add -5 and +5 using S.M.N.

$$
\begin{array}{lcccccc}
 & & & 1 & & 1 & \\
+5 & \longrightarrow & & 0 & 1 & 0 & 1 \\
-5 & \longrightarrow & + & 1 & 1 & 0 & 1 \\
\hline
 & & 1 & \multicolumn{4}{c}{\underbrace{0 \quad 0 \quad 1 \quad 0}_{+2}}
\end{array}
$$

drop since we only have 4-bits

$\Rightarrow 5 + (-5) = 2$

Not good for simple arithmetic

One's Complement Notation

Wednesday, April 7, 2021 2:19 PM

<u>Def</u>: The complement of a binary number is formed by reversing every bit.

<u>Ex</u>: Complement the binary value 0100.

<u>Sol</u>: 1011

<u>Ex</u>: Represent +5 using one's complement notation

<u>Sol</u>: $\underset{\text{sign}}{0}\ \underbrace{1\ 0\ 1}_{\text{magnitude}}$

<u>Ex</u>: Represent −5 using O.C.N.

<u>Sol</u>: $\overset{+5}{0\ 1\ 0\ 1} \xrightarrow[\text{complement}]{} \overset{-5}{\underset{\text{sign}}{1}\ \underbrace{0\ 1\ 0}_{\text{magnitude}}}$

<u>Ex</u>: Represent +2 and −2 using O.N.C

<u>Sol</u>: $\overset{+2}{0\ 0\ 1\ 0} \xrightarrow{\text{complement}} \overset{-2}{1\ 1\ 0\ 1}$

<u>Ex</u>: Represent 0.

<u>Sol</u>:

$$\left.\begin{array}{ll} 0000 & +0 \\ 1111 & -0 \end{array}\right\} \text{still have two zeros}$$

Ex: Add +5 and −5.

Sol:

$$\begin{array}{rr} +5 & 0101 \\ -5 & +\ 1010 \\ \hline & 1111 \end{array} \longrightarrow -0_{10}$$

Try this: Add +5 and −2

Sol: +5 is 0101 +2 is 0010

−2 is 1101

$$\begin{array}{rrr} & & 11 \\ +5 & \rightarrow & 0101 \\ +\ -2 & \rightarrow & +\ 1101 \\ \hline & & 10010 \end{array}$$

$\underset{\text{drop}}{1}\underbrace{0010}$ → $\left.\underset{\text{sign}}{0}\ \underbrace{010}_{\text{mag}}\right\} +2$

Two's Complement Notation

Wednesday, April 7, 2021 2:35 PM

Positive values are the same as unsigned binary.

To negate values:

Step 1: Start with given value in binary

Step 2: Complement every bit.

Step 3: Add 1

Ex: Write −5 using two's complement

1.) +5 is 0101

2.) Complement 1010

3.) Add 1

$$\begin{array}{r} 1010 \\ +\ 0001 \\ \hline 1011 \end{array} \leftarrow \boxed{1011 \text{ is } -5 \text{ using T.C.}}$$

Try this: Write −2 using two's complement

Sol: 1.) +2 is 0010

2.) Complement 1101

3.) Add 1 1110

$$\begin{array}{r} 1101 \\ +\ 0001 \\ \hline \end{array}$$

Try this: Convert −7 (1001) to +7

Sol: 1.) -7 is 1001

2.) Complement 0110

3.) Add 1 0111 $\longrightarrow$ $\boxed{+7 \text{ is } 0111}$

Ex: Find -0 from $+0$.

Sol: 1.) $+0$ is 0000

2.) complement 1111

3.) Add 1 0000

$$\begin{array}{r} 111 \\ 1111 \\ +\ 0001 \\ \hline 10000 \end{array}$$

drop

We only have 1 zero!!

Ex: Add $+5$ and -5

Sol: $+5$ is 0101
-5 is 1011

$$\Rightarrow \begin{array}{r} 111 \\ 0101 \\ +\ 1011 \\ \hline 1\underbrace{0000}_{0_{10}} \end{array}$$

drop

Ex: Add $+5$ and -2

Sol: $+5$ is 0101
-2 is 1110

$$\Rightarrow \begin{array}{r} 1 \\ 0101 \\ +\ 1110 \\ \hline 1\underbrace{0011} \end{array}$$

drop +3

Arithmetic works easily!!

We've worked with 4-bits of storage:

Min value is $-2^{4-1} = -2^{3} = -8$

Max value is $2^{4-1} - 1 = 2^{3} - 1 = 8-1 = 7$

In general with n-bits.

Min value is -2^{n-1}

Max value is $2^{n-1} - 1$

Overflow

Wednesday, April 7, 2021 3:00 PM

Def: Overflow – occurs when the value that is to be stored is outside the range of permissible values.

Handle by adding more bits or just detecting and reporting

Detecting Overflow:

Ex: Add +4 and +5

Sol:

$$\begin{array}{lll} & & \,{}^{1} \\ +5 & \text{is} & \,0\,1\,0\,1 \\ +4 & \text{is} & +\,0\,1\,0\,0 \\ \hline & & \,1\,0\,0\,1 \end{array} \longrightarrow -7$$

Ex: Add −4 and −5

$$\begin{array}{lll} -4 & \text{is} & \,1\,1\,0\,0 \\ -5 & \text{is} & +\,1\,0\,1\,1 \\ \hline & & \underset{\text{drop}}{1}\,\underbrace{0\,1\,1\,1}_{+7} \end{array}$$

sign is opposite the two original signs.

ASCII

Monday, April 12, 2021 1:34 PM

American Standard Code for Information Interchange

Character group	Range	
	Decimal	**Hexadecimal**
Control characters	0 – 31	0 – 1F
Punctuation	32 – 47	20 – 2F
Digits	48 – 57	30 – 39
More punctuation	58 – 64	3A – 40
Uppercase letters	65 – 90	41 – 5A
More punctuation	91 – 96	5B – 60
Lowercase letters	97 – 122	61 – 7A
More punctuation	123 – 126	7B – 7E
One final control character	127	7F

<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>A</th><th>B</th><th>C</th><th>D</th><th>E</th><th>F</th></tr>
<tr><th>2</th><td>[space]</td><td>!</td><td>“</td><td>#</td><td>$</td><td>%</td><td>&</td><td>'</td><td>(</td><td>)</td><td>*</td><td>+</td><td>,</td><td>-</td><td>.</td><td>/</td></tr>
<tr><th>3</th><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>:</td><td>;</td><td><</td><td>=</td><td>></td><td>?</td></tr>
<tr><th>4</th><td>@</td><td>A</td><td>B</td><td>C</td><td>D</td><td>E</td><td>F</td><td>G</td><td>H</td><td>I</td><td>J</td><td>K</td><td>L</td><td>M</td><td>N</td><td>O</td></tr>
<tr><th>5</th><td>P</td><td>Q</td><td>R</td><td>S</td><td>T</td><td>U</td><td>V</td><td>W</td><td>X</td><td>Y</td><td>Z</td><td>[</td><td>\</td><td>]</td><td>^</td><td>_</td></tr>
<tr><th>6</th><td>`</td><td>a</td><td>b</td><td>c</td><td>d</td><td>e</td><td>f</td><td>g</td><td>h</td><td>I</td><td>j</td><td>k</td><td>l</td><td>m</td><td>n</td><td>o</td></tr>
<tr><th>7</th><td>p</td><td>q</td><td>r</td><td>s</td><td>t</td><td>u</td><td>v</td><td>w</td><td>x</td><td>y</td><td>z</td><td>{</td><td>|</td><td>}</td><td>~</td><td></td></tr>
</table>

$A = 41_{16} = 01000001_2$

$a = 61_{16} = 01100001_2$

Floating Point Notation

Monday, April 12, 2021 1:34 PM

In base 10 we have exponential notation:

$$+3.0 \times 10^{8}$$

8 ← exponent

sign, mantissa, base

In floating Point notation we have 3 parts: (32-bits)

1.) Sign for the mantissa
- 0 for positive
- 1 for negative

2.) Exponent
- 8-bit two's complement notation
- Base 2

3.) Mantissa
- Unsigned binary number
- 23 remaining bits

Ex: The number 56.0 in 32-bit floating point notation.

+/-	Exponent	Mantissa
0	0000 0011	000 0000 0000 0000 0000 0111

In exponential form $+111 \times 2^{11}$ ← exponent

sign, mantissa

Convert to base 10 $7 \times 2^{3} = 7(8) = 56.0$

In unsigned binary 111000

In unsigned binary

$\underbrace{111}_{\text{mantissa}}\underbrace{000}_{\text{3 zeros from exp.}}$

Ex: Convert the 32-bit floating point number to base 10.

+/–	Exponent								Mantissa																						
1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

1.) Write the number in the binary exponential form

Sign is (–)

Mantissa is $11_2 \rightarrow 3_{10}$

Exponent is 11111110_2

$$-11 \times 2^{11111110}$$

2.) Convert mantissa and exponent to base 10

For the mantissa:

$$\overset{1}{2^1} + \overset{1}{2^0} = 2 + 1 = 3$$

For exponent (two's complement, 8-bit)

Given: 1 1 1 1 1 1 1 0

Complement: 0 0 0 0 0 0 0 1

Add 1: 0 0 0 0 0 0 1 0 → 2

exponent is –2

⇒ We have:

$$-3 \times 2^{-2}$$

3.) Evaluate:

$$-3 \times 2^{-2} = -3\left(\frac{1}{2^2}\right) = -3\left(\frac{1}{4}\right) = -\frac{3}{4} = \boxed{-0.75}$$

Try this: Convert to base 10.

+/-	Exponent	Mantissa
1	1 1 1 1 1 1 0 1	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0

1.) Exponential form (binary)

$$-110 \times 2^{11111101}$$

2.) Conversion to base 10 for mantissa and exp.

Mantissa:

$$\begin{array}{ccc} 1 & 1 & 0 \\ 2^2 & 2^1 & \cancel{2^0} \end{array}$$

$$4+2 = 6$$

Exponent:

Given:	1	1	1	1	1	1	0	1
Comp:	0	0	0	0	0	0	1	0
Add 1:	0	0	0	0	0	0	1	1

$\rightarrow 3$

$\Rightarrow$ exp is -3

3.) Eval:

$$-6 \times 2^{-3}$$

$$-6 \times \left(\frac{1}{2^3}\right) = -\frac{6}{8} = -\frac{3}{4} = \boxed{-0.75}$$