Section1-4 Negative Binary Numbers

Monday, January 18, 2021 2:26 PM

Unsigned numbers:

if you have n bits, can represent numbers from 0 to $2^n - 1$

(e.g. for n=3, can represent 0, 1, 2, 3, 4, 5, 6, 7

For signed numbers, we reserve the first bit to tell us the sign

There are three systems for this, but let's examine the simplest → "sign and magnitude"

Let the first bit represent the sign

"0" → number is positive

"1" → number is negative

Example: 0111 → +7

1111 → −7

note that here the "word length" n=4

we can only represent −7 to +7

or $-(2^{n-1}-1)$ to $+(2^{n-1}-1)$

Quiz: what are these?

0101

1101

0000

1000

∴ have ±0

There are THREE systems for representing signed binary numbers

- sign and magnitude
- 2's complement
- 1's complement

2's complement

If the number is POSITIVE, binary 2's comp. is the SAME as sign + magnitude.

If the number (N) is NEGATIVE, the 2's complement is

$$N^* = 2^n - N$$

(where "N" is positive, so $N^* \Rightarrow -N$ in 2's comp.)

Example: let $N = 6_{10}$ and we want to represent $-6 = -N$ in 2's complement:

$$N^* = 2^n - N = 2^n - [6_{10}]$$

$$= 2^n - [110_2]$$

and suppose n = word length = 4

$$N^* = 2^n - N = 2^4 - \quad = 16 - 6 = 10_{10}$$

$$= 1010_2$$

$$\underset{\text{"}-\text{"}}{\underset{\uparrow}{1}}\ \underbrace{010}_{\text{not "6"}} = -6$$ in two's complement

Suppose your computer recieved "1010"

$$N^* = 2 - N \Rightarrow N = 2^n - N^*$$

$$N = 2^4 - 1010$$

$$= 16_{10} - 1010_2$$

$$= 10000 - 1010$$

$$\begin{array}{r} 10000 \\ 1010 \\ \hline \end{array}$$

So, how is this different from sign + magnitude?

+ N	All systems	− N	sign + mag	2's comp
0	0000	−0		
+1	0001	−1		
2	0010	−2		
3	0011	−3		
4	0100	−4		
5	0101	−5		
6	0110	−6		
7	0111	−7		

Now, what is $N^* = 1000$? $(n=4)$

$$N^* = 2^n - N \quad \Rightarrow \quad N = 2^n - N^*$$

$$2^4 = 16_{10} = 10000_2$$

$$\therefore \begin{array}{r} 10000 \\ -\ 1000 \\ \hline \end{array}$$

Why do we even bother?

Consider subtracting two numbers in sign + mag: $2_{10} - 5_{10}$ (let $n=4$)

same as adding -5_{10} and $+2_{10}$

$-5_{10} = ?$ $+2 = 0010_{2's}$

So.

$$\begin{array}{r} 1101 \\ +\ 0010 \\ \hline \end{array}$$

You'd get the wrong answer. You could still design a circuit to do this (the same way you would do it) but it is complicated, and in circuitry "complicated" = "expensive"

But if you do it in 2's complement: (n=4)

$-5_{2's} = ?$ $+2 = 0010$

$N^* = 2^n - N$

$N = 5 = 0101$

$2^4 = 16 =$

$$\begin{array}{r} 10000 \\ -\quad 101 \\ \hline \end{array}$$

= -5 in 2's complement

$\therefore$ $-5 + 2$ =

$$\begin{array}{r} 1011 \\ +\,0010 \\ \hline \end{array}$$

Now here's the cod part:

Recall $N^* = 2^n - N$ we are subtracting (n+1) bit #'s

e.g. if $n = 4$, $2^n = 16 = 10000$

Need 5 bits to represent that

but we don't have 5 bits (n+1 bits)

BUT $N^* = 2^n - N$ add $0 = -1 + 1$

$N^* = 2^n - N - 1 + 1$

$=(2^n-1) - N + 1$

we can therefer subtract any 4-bit number from this without borrowing

Now, recall $1-0=1$ we have no $0-1$ or $0-0$ because all 1's

$1-1=0$

Example

suppose $n=6$ and we want the 2's complement of $N = 010110$
(e.g. we want 2's complement of -22_{10})

$$N^* = \underset{\substack{\uparrow \\ \text{all 1's}}}{(2^n-1)} - N + 1$$

two steps: ① $(2^n-1) - N =$

$$\begin{array}{r} 111111 \\ -\,010110 \\ \hline \end{array}$$

result is the complement digit by digit!

② add 1

$$\begin{array}{r} 101001 \\ +\quad\quad 1 \\ \hline \end{array}$$

$= N^*$

and means -22

Example let $n=7$, $N = 0101100$

$N^* = (2^n - 1) - N + 1$

$n=7$

$$\begin{array}{r} 1111111 \\ -\ 0101100 \\ \hline \\ +\qquad\qquad 1 \\ \hline \end{array} = N^*$$

It works out hat for N, to get N^*

0101100

OK, let's practice using this $(n=4)$

① Addition of two numbers whose sum $< 2^{n-1}$

(15)

e.g.

$$\begin{array}{r} 3 \\ 4 \\ \hline 7 \end{array} \qquad \begin{array}{r} 0011 \\ 0100 \\ \hline 0111 \end{array}$$

correct

② Addition of two positive numbers whose sum $\geq 2^{n-1}$

$$\begin{array}{r} 6 \\ 7 \\ \hline 13 \end{array} \qquad \begin{array}{r} {\scriptstyle 1\,1} \\ 0110 \\ 0111 \\ \hline 1101 \end{array}$$

this equals 13 all right, except we need the first bit to represent the sign

we don't have a 5th bit so this is

an OVERFLOW condition

③ Add a positive and negative number where the negative has greater magnitude

$$\begin{array}{r} +5 \\ -6 \\ \hline -1 \end{array}$$

$$\therefore \begin{array}{r} 0\,1\,0\,1 \\ +\,1\,0\,1\,0 \\ \hline \end{array}$$

④ Add a positive and negative # but positive number is larger in magnitude

$$\begin{array}{r} -5 \\ +6 \\ \hline +1 \end{array}$$

$+5 = 0101$

$-5 =$

$$\therefore \begin{array}{r} \overset{1}{1}\,\overset{1}{0}\,1\,1 \\ 0\,1\,1\,0 \\ \hline 1\,0\,0\,0\,1 \end{array}$$

This answer is correct - if we ignore the extra bit (previous cases did not have an extra bit)

this is NOT an overflow condition.

⑤ Add two negative numbers $|\text{sum}| < 2^{n-1}$

$$\begin{array}{r} -3 \\ -4 \\ \hline -7 \end{array}$$

$+3 = 0011$

$+4 = 0100$

$\therefore$ $-3 =$

$-4 =$

$\overset{1}{1}\,1\,0\,1$

$$\begin{array}{r} +\,1100 \\ \hline \overset{\uparrow}{1}\,\underbrace{1001} \end{array}$$

is this −7? sign bit is right...

ignore

convert back to see
= 0111 ✓

⑥ Add two negative #'s $|sum| > 2^{n-1}$

$$\begin{array}{r} -5 \\ -6 \\ \hline -11 \end{array}$$

$+5 = 0101$

$+6 = 0110$

$\therefore$

$-5 =$

$-6 =$

$$\begin{array}{r} \overset{1}{}\\ 1\,0\,1\,1 \\ +\,1\,0\,1\,0 \\ \hline \overset{}{1}\,0\,1\,0\,1 \end{array}$$

↑
ignore

WRONG! Looks like a positive number

$\boxed{\text{OVERFLOW}}$

How to detect overflow:

If you throw the extra bit (if there is one) and the result should be POSITIVE but looks NEGATIVE (sign bit is "1" but should "0") it's an overflow

If you throw away the extra bit (if there is one) and the result should be NEGATIVE but looks POSITIVE (sign bit is "0" but should be "1") it's an OVERFLOW

1's complement system

Definition: 1's complement = $\bar{N}$

$-N = \bar{N}$ in 1's complement

$\bar{N} = (2^n - 1) - N$

(compare to $N^* = 2^n - N$)

recall $2^n - 1 = 1111\ldots$ e.g. $2^4 = 16 = 10000$

$2^4 - 1 = 1111$

(4 1's)

Example $N = 7$ $n = 4$

$-7 = ?$

$+7 = 0111$

Table $(n=4)$

N	$\bar{N}$	$-N$	$\bar{N}$
+0	0000	−0	1111
+1	0001	−1	1110
+2	0010	−2	1101
+3	0011	−3	1100
+4	0100	−4	1011
+5	0101	−5	1010
+6	0110	−6	1001
+7	0111	−7	1000

↑ "0" = positive

↑ "1" = negative

This system also has a negative and positive zero.

Addition in 1's complement

We'll do the cases again, but in 1's complement we treat the last carry differently – we'll "carry around" – meaning last carry is taken back around to the right-most column and added.

Case 1 add two positive integers sum $<2^{n-1}$

$$\begin{array}{r} +3 \\ +4 \\ \hline 7 \end{array} \qquad \begin{array}{r} 0011 \\ 0100 \\ \hline 0111 \end{array}$$

Case 2 add two positive #s sum $\geq 2^{n-1}$

$$\begin{array}{r} +4 \\ +5 \\ \hline +9 \end{array} \qquad \begin{array}{r} 0100 \\ 0101 \\ \hline \end{array}$$

wrong because it looks like a negative number

$\boxed{\text{OVER FLOW}}$

Case 3

Add positive and negative number where negative number has greater magnitude

$$\begin{array}{r} +4 \\ -5 \\ \hline -1 \end{array}$$

$+4 = 0100$

$+5 = 0101$ so $-5 =$

$$\begin{array}{r} 0100 \\ +\ 1010 \\ \hline \end{array}$$

Case 4

positive and negative but positive number is larger

$$\begin{array}{r} -5 \\ +6 \\ \hline +1 \end{array}$$

$$\begin{array}{r} 1010 \\ 0110 \\ \hline 10000 \\ 1 \\ \hline 0001 \end{array}$$

carry! take to rightside

✓ correct

Case 5

addition of two negative numbers whose $|\text{sum}| < 2^{n-1}$

-3 $\quad +3 = 0011$ $\quad 1100$

-2 $\quad +2 = 0010$ $\quad 1101$

-5

$$\begin{array}{rr} \hline & \textcircled{1}\;1\;0\;0\;1 \\ \llcorner\!\!\longrightarrow & 1 \\ \hline & 1\;0\;1\;0 \end{array}$$

end around carry

Case 6 Addition of two negative numbers whose $|sum| \geq 2^{n-1}$

$$\begin{array}{r} -4 \\ -7 \\ \hline -11 \end{array} \qquad \begin{array}{l} +4 = 0100 \\ +7 = 0111 \end{array} \qquad \begin{array}{rr} & 1\;0\;1\;1 \\ + & 1\;0\;0\;0 \\ \hline 1 & 0\;0\;1\;1 \\ \llcorner\!\!\longrightarrow & 1 \\ \hline \end{array}$$

wrong! this looks like a positive #!

$\boxed{\text{OVERFLOW}}$

Example let's compare 1's and 2's complement

let $n=8$ add -12 and -19

$+12 = 00001100$ $+19 = 00010011$

One's compl:

now add

$$\begin{array}{lrr} -12 & & 1\;1\;1\;1\;0\;0\;1\;1 \\ -19 & + & 1\;1\;1\;0\;1\;1\;0\;0 \\ \hline \end{array}$$

2's complement

+12 = 00001100 +19 = 00010011

-12 = -19 =

now add

$$\begin{array}{r} 11110100 \\ +\,11101101 \\ \hline \end{array}$$

How to detect overflow:

In 1's or 2's complement:

If when you are done you expected a negative # but got a positive number, or vice versa, you have an $\boxed{\text{OVERFLOW}}$

Special extra check for 2's complement only

If carry bit is opposite value as sign bit, it's an overflow

Example

our previous example. We had

$$\begin{array}{r} 1\;1\;1\;1\;1 \\ 1\;1\;1\;1\;0\;1\;0\;0 \\ 1\;1\;1\;0\;1\;1\;0\;1 \\ \hline \end{array}$$

1 1 1 1 0 0 0 0 1

another: 120 + 8 (should give +128)

but $n=8$ $2^{n-1} = 128$

should overflow

$$\begin{array}{rcccccccc}
 & & 1 & 1 & 1 & 1 & & & \\
120: & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
\hline
 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}$$

$\boxed{\text{OVERFLOW}}$

Summary

- Positive numbers the same for all three systems
- sign + magnitude: first bit is sign, the rest is normal magnitude of the number
- 2's complement: $N^* = 2^n - N$
- 1's complement $\bar{N} = (2^n - 1) - N$

Positive #'s

+0	0000
+1	0001
+2	0010
+3	0011
+4	0100
+5	0101
+6	0110
+7	0111

ALL systems

↑ sign bit

Negative #'s

	sign+mag	2's N^*	1's $\bar{N}$
−0	1000	—	1111
−1	1001	1111	1110
−2	1010	1110	1101
−3	1011	1101	1100
−4	1100	1100	1011
−5	1101	1011	1010
−6	1110	1010	1001
−7	1111	1001	1000
−8	—	1000	—

Addition sign and magnitude - hard to implement in hardware

2's complement easy to implement addition - discard carry
but one more step to convert # to N^*

1's complement easy to convert numbers to $\bar{N}$
but one more step in addition