



Jiangxi University of Science and Technology

DIGITAL DESIGN

Lecture 2 Signed Number





Binary Addition

- Single Bit Addition Table

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 10 \quad \text{Note “carry”}$$



Example:

Add the following binary numbers

$$\begin{array}{r} 1101 \\ +0111 \\ \hline \end{array}$$

- Solution: Add the following binary numbers

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 10 \quad \text{Note "carry"}$$

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



Binary Subtraction

- Single Bit Subtraction Table

$$0 - 0 = 0$$

$$1 - 0 = 1$$

$$1 - 1 = 0$$

$$0 - 1 = 1 \text{ with a "borrow"}$$



Example

Subtract the following binary numbers

$$0 - 0 = 0$$

$$1 - 0 = 1$$

$$1 - 1 = 0$$

$$0 - 1 = 1 \text{ with a “borrow”}$$

$$\begin{array}{r} 1101 \\ - 0111 \\ \hline \end{array}$$

$$\begin{array}{r} 1101 \\ \overrightarrow{0}\overrightarrow{0}01 \\ - 0111 \\ \hline 0110 \end{array}$$

Arithmetic

- Decimal

$$\begin{array}{r} 11 \\ 1234 \\ + 5678 \\ \hline 6912 \end{array} \qquad \begin{array}{r} 5274 \\ - 1638 \\ \hline 3636 \end{array}$$

- Binary

$$\begin{array}{r} 1\ 1 \\ 1011 \\ + 1010 \\ \hline 10101 \end{array} \qquad \begin{array}{r} 1011 \\ - 0110 \\ \hline 0101 \end{array}$$

Negative Binary Numbers

- In decimal we are quite familiar with placing a “-” sign in front of a number to denote that it is negative
- The same is true for binary numbers a computer won’t understand that
- What happens in memory then?

2.4 Signed Integer Representation



- Example:
 - Using signed magnitude binary arithmetic, find the sum of 75 and 46.
 - First, convert 75 and 46 to binary, and arrange as a sum, but separate the (positive) sign bits from the magnitude bits.
- Example:
 - Using signed magnitude binary arithmetic, find the sum of 75 and 46.
 - Just as in decimal arithmetic, we find the sum starting with the rightmost bit and work left.

$$\begin{array}{r} 0 \quad 1001011 \\ 0 + 0101110 \\ \hline \end{array}$$

$$\begin{array}{r} 0 \quad 1001011 \\ 0 + 0101110 \\ \hline 1 \end{array}$$

2.4 Signed Integer Representation



- Example:

- Using signed magnitude binary arithmetic, find the sum of 75 and 46.
- In the second bit, we have a carry, so we note it above the third bit.

$$\begin{array}{r} & & 1 \\ & 0 & 1001011 \\ 0 & + & 0101110 \\ \hline & & 01 \end{array}$$

- Example:

- Using signed magnitude binary arithmetic, find the sum of 75 and 46.
- The third and fourth bits also give us carries.

$$\begin{array}{r} & & 1 & 1 & 1 \\ & 0 & 1001011 \\ 0 & + & 0101110 \\ \hline & & 1001 \end{array}$$

- Example:

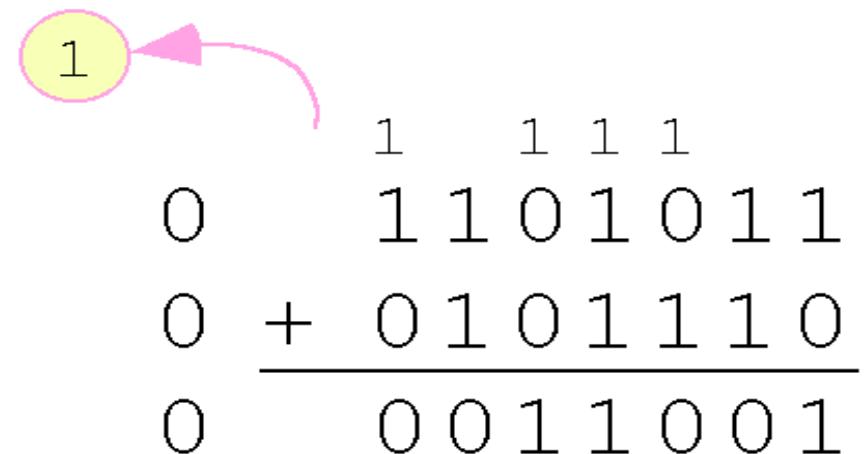
- Using signed magnitude binary arithmetic, find the sum of 75 and 46.
- Once we have worked our way through all eight bits, we are done.

$$\begin{array}{r} & & 1 & 1 & 1 \\ & 0 & 1001011 \\ 0 & + & 0101110 \\ \hline 0 & 1111001 \end{array}$$

In this example, we were careful to pick two values whose sum would fit into seven bits. If that is not the case, we have a problem.

2.4 Signed Integer Representation

- Example:
 - Using signed magnitude binary arithmetic, find the sum of 107 and 46.
 - We see that the carry from the seventh bit *overflows* and is discarded, giving us the **erroneous result**: $107 + 46 = 25$.



A diagram illustrating a signed magnitude binary addition. A yellow circle contains the number '1', with a pink arrow pointing to it from the left. Below the circle is a horizontal line. To the left of the line, there are two columns of binary digits: '1' and '0'. To the right of the line, there are two columns of binary digits: '1' and '0'. The line itself has a '+' sign in its center. Below the line, the result of the addition is shown: '0' followed by '0011001'. This represents the erroneous result of 25 in binary form.

$$\begin{array}{r} & \textcircled{1} \\ & \curvearrowleft \\ \begin{array}{r} 1 \\ 0 \end{array} & \begin{array}{r} 1 \\ 1 \end{array} & \begin{array}{r} 1 \\ 0 \end{array} & \begin{array}{r} 1 \\ 1 \end{array} & \begin{array}{r} 1 \\ 1 \end{array} \\ 0 & + & 0 & 1 & 0 & 1 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{array}$$

Negative numbers

- How do we write negative binary numbers?
 - Prefix numbers with minus symbol?
 - 3 approaches:
 - Sign and magnitude
 - Ones-complement
 - Twos-complement
 - All 3 approaches represent positive numbers in the same way
- Two's complement is the system used in microprocessors
 - Most significant bit becomes important

Signed Magnitude

Represent the decimal number as binary

Left bit (MSB) used as the sign bit

Only have 7 bits to express the number

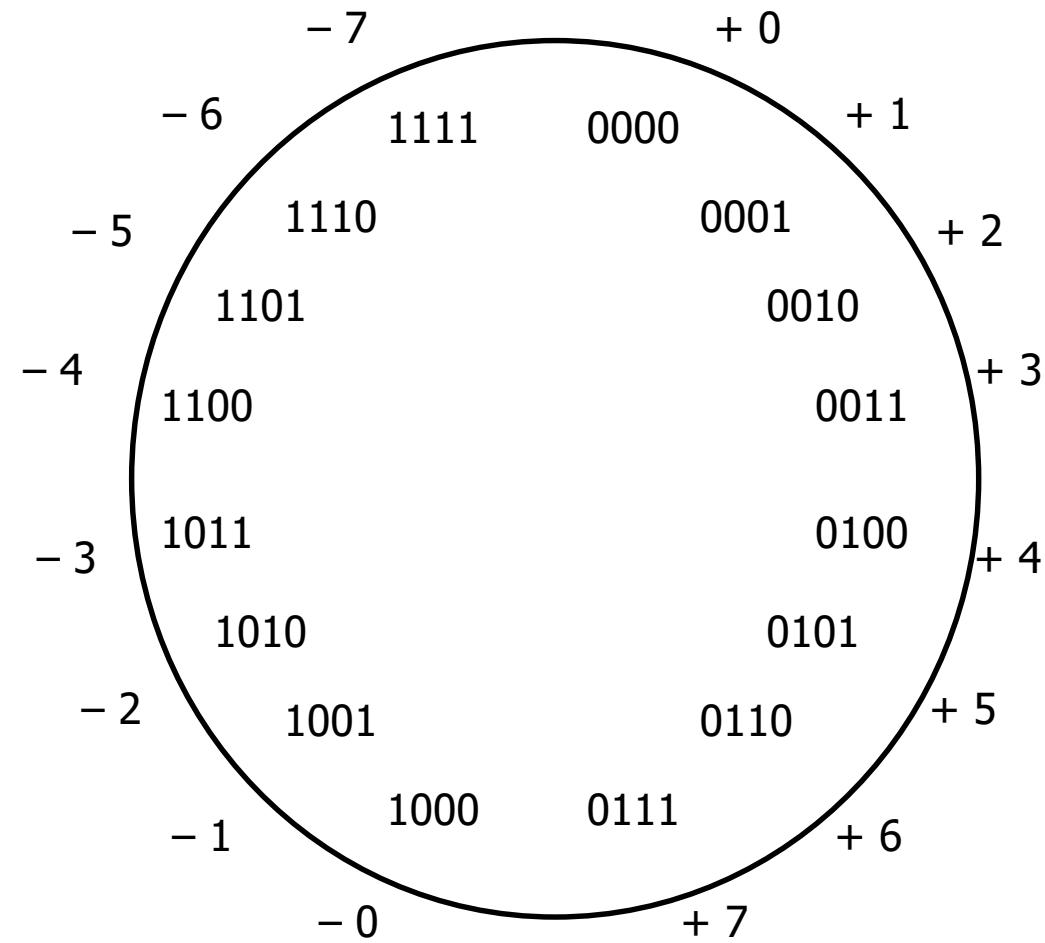
$$12_{10} = 00001100$$

$$-12_{10} = 10001100$$

How many representations are there for zero?

Sign and magnitude

- Most significant bit (MSB) is the sign bit
 - 0 ≡ positive
 - 1 ≡ negative
- Remaining bits are the number's magnitude



Sign and magnitude

- Problem 1: Two representations of for zero
 - $+0 = 0000$ and also $-0 = 1000$
- Problem 2: Arithmetic is cumbersome
 - $4 - 3 \neq 4 + (-3)$

Add	Subtract	Compare and subtract
4 0100 + 3 + 0011 = 7 = 0111	4 0100 0100 - 3 + 1011 - 0011 = 1 ≠ 1111 = 0001	- 4 1100 1100 + 3 + 0011 - 0011 - 1 ≠ 1111 = 1001

Ones-complement

Method: **Invert** the ones and zeros

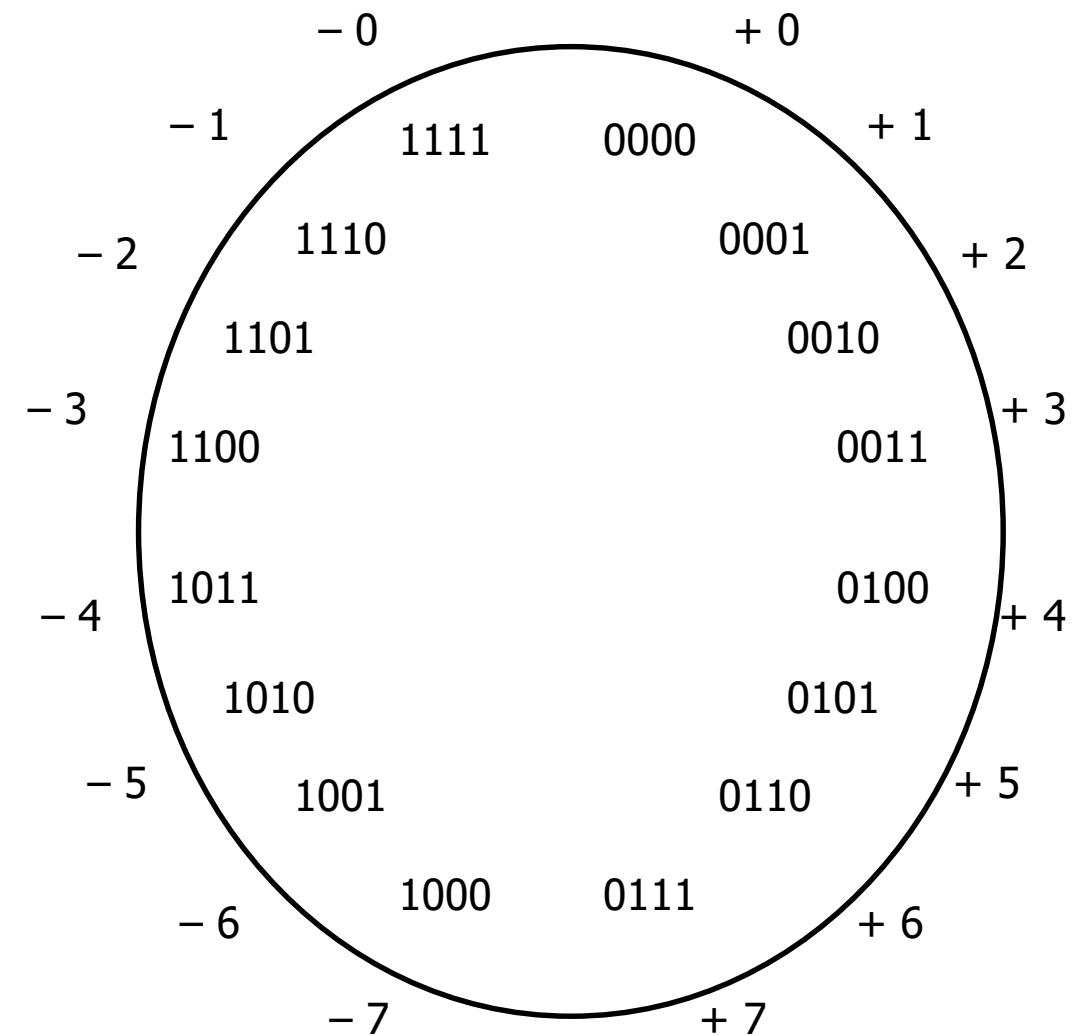
$$11_{10} = 00001011$$

$$-11_{10} = 11110100$$

0 in MSB implies positive

1 in MSB implies negative

- Negative number: Bitwise complement of positive number
 - $0111 \equiv 7_{10}$
 - $1000 \equiv -7_{10}$



Ones-complement

- Solves the arithmetic problem

Add	Invert, add, add carry	Invert and add
$ \begin{array}{r} 4 & 0100 \\ + 3 & + 0011 \\ \hline = 7 & = 0111 \end{array} $	$ \begin{array}{r} 4 & 0100 \\ - 3 & + 1100 \\ \hline = 1 & 1\ 0000 \end{array} $ <p>add carry: \uparrow +1</p>	$ \begin{array}{r} - 4 & 1011 \\ + 3 & + 0011 \\ \hline - 1 & 1110 \end{array} $

↑

end-around carry

Why ones-complement works

- The ones-complement of an 4-bit positive number y is $1111_2 - y$
 - $0111 \equiv 7_{10}$
 - $1111_2 - 0111_2 = 1000_2 \equiv -7_{10}$
- What is 1111_2 ?
 - 1 less than $10000_2 = 2^4 - 1$
 - $-y$ is represented by $(2^4 - 1) - y$



Why ones-complement works

- Adding representations of x and $-y$ where x, y are positive numbers, we get $x + ((2^4 - 1) - y) = (2^4 - 1) + (x - y)$
 - If $x < y$, then $x - y < 0$. There will be no carry from $(2^4 - 1) + (x - y)$. Just add representations to get correct negative number.
 - If $x > y$, then $x - y > 0$. There will be a carry. Performing end-around carry subtracts 2^4 and adds 1, subtracting $(2^4 - 1)$ from $(2^4 - 1) + (x - y)$
 - If $x = y$, then answer should be 0, get $(2^4 - 1) = 1111_2$

So what's wrong?

- Still have two representations for zero!
 - $+0 = 0000$ and also $-0 = 1111$

Twos-complement

- Method: Take the one's complement and add 1

$$1110 = 00001011$$

$$-1110 = 11110100$$

$$-1110 = 11110101$$

one's comp

two's comp

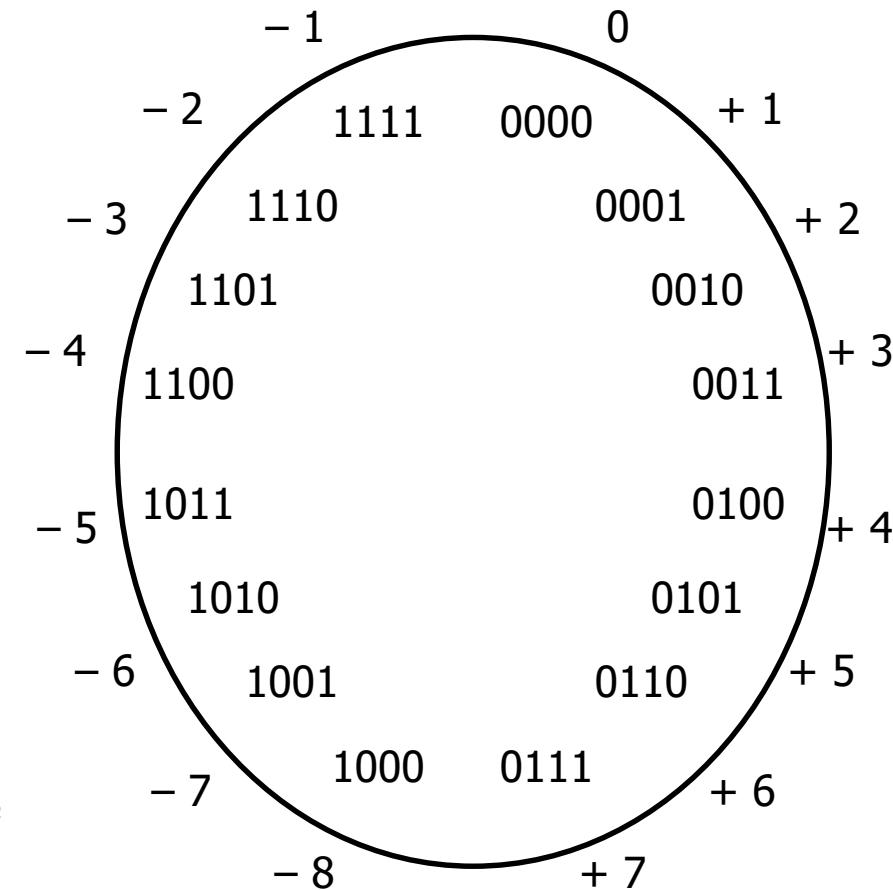
- Negative number: Bitwise complement **plus one**

- $0111 \equiv 7_{10}$

- $100\textcolor{blue}{1} \equiv -7_{10}$

- Benefits:

- Simplifies arithmetic
- Only one zero!



Twos-complement

Add	Invert and add	Invert and add
$ \begin{array}{r} 4 \quad 0100 \\ + 3 \quad + 0011 \\ \hline = 7 \quad = 0111 \end{array} $	$ \begin{array}{r} 4 \quad 0100 \\ - 3 \quad + 1101 \\ \hline = 1 \quad 1\ 0001 \end{array} $ <p style="text-align: center;">drop carry</p>	$ \begin{array}{r} - 4 \quad 1100 \\ + 3 \quad + 0011 \\ \hline - 1 \quad 1111 \end{array} $



Why twos-complement works

- Recall: The ones-complement of a b -bit positive number y is $(2^b - 1) - y$
- Twos-complement adds one to the bitwise complement, thus, $-y$ is $2^b - y$
 - $-y$ and $2^b - y$ are equal mod 2^b (have the same remainder when divided by 2^b)
 - Ignoring carries is equivalent to doing arithmetic mod 2^b
- One representation of zero (Check)
- Enables subtraction operation by considering the addition of a positive number with a two's complement number
- Only need addition
- MSB indicates the sign of the number



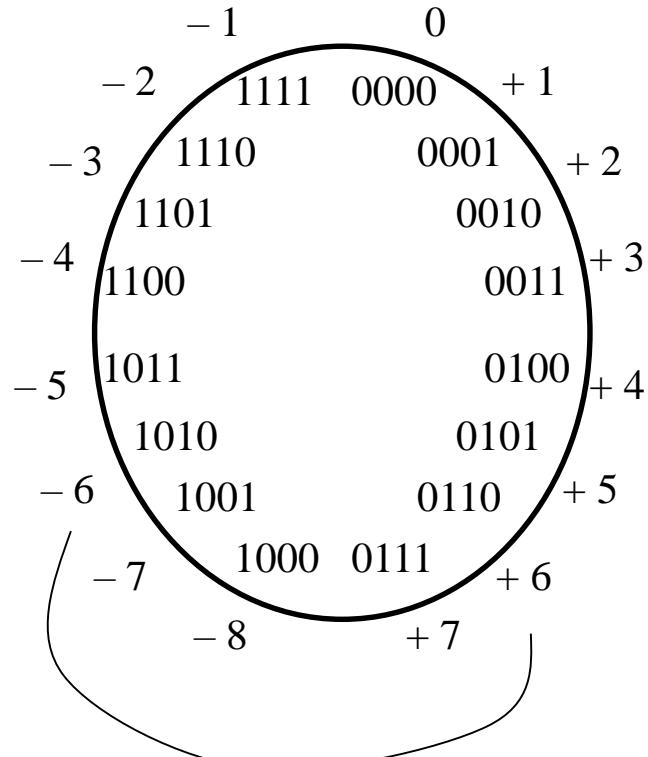
Why twos-complement works

- Adding representations of x and $-y$ where x, y are positive numbers, we get $x + (2^b - y) = 2^b + (x - y)$
 - If there is a carry, that means that $x \geq y$ and dropping the carry yields $x - y$
 - If there is no carry, then $x < y$, then we can think of it as $2^b - (y - x)$, which is the twos-complement representation of the negative number resulting from $x - y$.

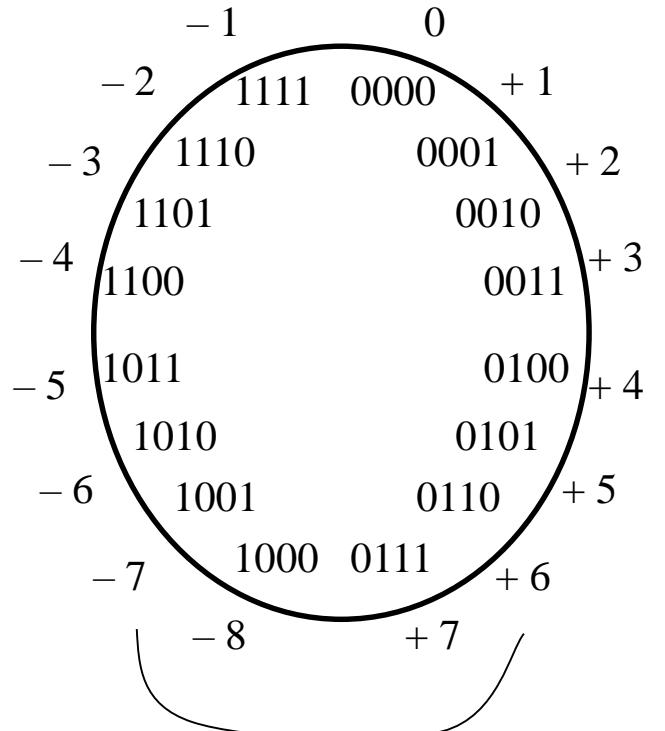
Only need one type of hardware/process to add both signed and unsigned numbers.

Overflow

- Answers only correct mod 2^b
 - Summing two positive numbers can give a negative result
 - Summing two negative numbers can give a positive result



$$6 + 4 \Rightarrow -6$$



$$-7 - 3 \Rightarrow +6$$

Miscellaneous

- Sign-extension
 - Write +6 and -6 as twos-complement
 - 0110 and 1010
 - Sign-extend to 8-bit bytes
 - **0000**0110 and **1111**1010
- Can't infer a representation from a number
 - 11001 is 25 (unsigned)
 - 11001 is -9 (sign and magnitude)
 - 11001 is -6 (ones complement)
 - 11001 is -7 (twos complement)

Compression between Complements / Sign-and-Magnitude

4 bit(signed number)
Negative

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

4 bit(signed number)
(positive)

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

BCD (Binary-Coded Decimal)

<u>Decimal Symbols</u>	<u>BCD Code</u>
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001



Positional Notation

A positive number N can be written as:

$$N = (a_{n-1}a_{n-2}\dots.a_1a_0.a_{-1}a_{-2}\dots.a_{-m})_r$$

Where:

. = radix point

r = radix or base of number system

n = number of integer digits to the left of radix point

m = number of fractional digits to the right of radix point

a_i = integer digit i

a_j = fractional digit j

a_{n-1} = most significant digit

a_{-m} = least significant digit



Radix Complements

Definition

Radix Complements

$$[N]_r = r^n - (N)_r$$

Where

$[N]_r$ = Radix Complement of N

$(N)_r$ = Original Number, N

r = radix or base

n = number of digits



Radix Complements

Definition

Note:

$$[N]_r + (N)_r = r^n - (N)_r + (N)_r = r^n$$

Or,

N added to its complement is equal to r^n

But this is just 0, if we ignore the Most Significant Digit (MSD)

Complements(r- 1)

Diminished-Radix

E.g.: $(r-1)$'s complement, or 9s complement, of $(22)_{10}$ is:

$$(10^2 - 1) - 22 = (77)_{9s} \text{ [This means } -(22)_{10} \text{ is } (77)_{9s}]$$

$(r-1)$'s complement, or 1s complement, of $(0101)_2$ is:

$$(2^4 - 1) - 0101 = (1010)_{1s} \text{ [This means } -(0101)_2 \text{ is } (1010)_{1s}]$$

مکمل $(r-1)$ عدد x با n رقم صحیح و m رقم اعشار برابر است با :

$$(r^n - r^m) - x$$

$$(10^2 - 1) - 22 = 99 - 22 = (77)_{9s}$$

روش نظری : همه رقم ها از $(r-1)$ کم می کنیم:

$$(2^4 - 1) - 0101 = 1111 - 0101 = (1010)_{1s}$$



Radix Complements

Two's Complement

$$[N]_2 = 2^n - (N)_2$$

Ten's Complement

$$[N]_{10} = 10^n - (N)_{10}$$

16's Complement

$$[N]_{16} = 16^n - (N)_{16}$$

r = radix or base

n = number of digits

Summary

- Two's complement calculations
- Method 1

$$[N]_2 = 2^n - (N)_2$$

- Method 2
- Method 3

$$[N]_2 = [N]_1 + 1$$

Leave all of the least significant 0's and first 1 **unchanged** and then “flip” or complement the bits for all other digits

Complement

- Complement 10 of 546700

$$10^6 - 546700 = 1000000 - 546700 = 453300$$

- Complement 9 of 546700

- $(10^6 - 1) - 546700 = 999999 - 546700 = 453299$

- Other way is Complement 9+1

- $453299 + 1 = 453300$

Complement

- Complement 2 of 1101100

$$2^7 - 1101100 = (10000000 - 1101100)_2 = 0010100$$

- Complement 1 of 1101100

$$(2^7 - 1) - 1101100 = (1111111 - 1101100)_2 = 0010011 \cdot$$

- Other way is complement 1+1

$$0010011 + 1 = 0010100$$

1s Complement Subtraction/add

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

$$\begin{array}{r}
 +5 & 0101 \\
 + -5 & + 1010 \\
 \hline
 - & \hline \\
 -0 & 1111 \\
 \hline
 - & \hline
 \end{array}$$

$$\begin{array}{r}
 -2 & 1101 \\
 + -5 & + 1010 \\
 \hline
 - & \hline \\
 -7 & \textcolor{red}{10111} \\
 \hline
 - & + 1 \\
 \hline
 & \hline \\
 1000 &
 \end{array}$$

$$\begin{array}{r}
 -3 & 1100 \\
 + -7 & + 1000 \\
 \hline
 - & \hline \\
 -10 & \textcolor{red}{10100} \\
 \hline
 - & + 1 \\
 \hline
 & \hline \\
 0101 &
 \end{array}$$

2s Complement Subtraction/add

$$\begin{array}{r}
 -2 & 1110 \\
 + -6 & + 1010 \\
 \hline
 -8 & \textcolor{red}{1}1000 \\
 \hline
 \end{array}$$

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

$$\begin{array}{r}
 +4 & 0100 \\
 + -7 & + 1001 \\
 \hline
 -3 & 1101 \\
 \hline
 \end{array}$$

$$\begin{array}{r}
 +6 & 0110 \\
 + -3 & + 1101 \\
 \hline
 +3 & \textcolor{red}{1}0011 \\
 \hline
 \end{array}$$

$$\begin{array}{r}
 +5 & 0101 \\
 + +6 & + 0110 \\
 \hline
 +11 & 1011 \\
 \hline
 \end{array}$$

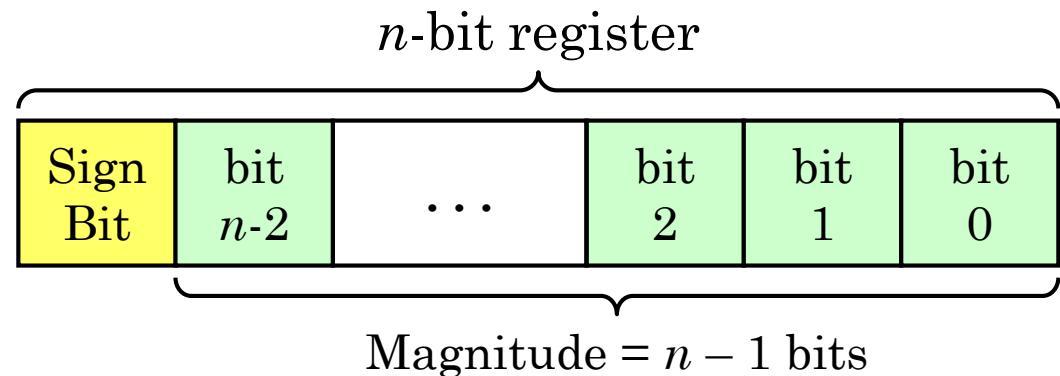
$$\begin{array}{r}
 -3 & 1101 \\
 + -6 & + 1010 \\
 \hline
 -9 & \textcolor{red}{1}0111 \\
 \hline
 \end{array}$$



Signed Numbers

- Several ways to represent a signed number
 - Sign-Magnitude
 - 1's complement
 - 2's complement
- Divide the range of values into 2 equal parts
 - First part corresponds to the positive numbers (≥ 0)
 - Second part correspond to the negative numbers (< 0)
- The 2's complement representation is widely used
 - Has many advantages over other representations

Sign-Magnitude Representation



- Independent representation of the sign and magnitude
- Leftmost bit is the sign bit: 0 is positive and 1 is negative
- Using n bits, largest represented magnitude = $2^{n-1} - 1$

Sign-magnitude
representation of +45
using 8-bit register

0	0	1	0	1	1	0	1
---	---	---	---	---	---	---	---

Sign-magnitude
representation of -45
using 8-bit register

1	0	1	0	1	1	0	1
---	---	---	---	---	---	---	---



Properties of Sign-Magnitude

- Two representations for zero: +0 and -0
- Symmetric range of represented values:
For n-bit register, range is from $-(2^{n-1} - 1)$ to $+(2^{n-1} - 1)$
For example using 8-bit register, range is -127 to +127
- Hard to implement addition and subtraction
 - Sign and magnitude parts have to be processed independently
 - Sign bit should be examined to determine addition or subtraction
Addition is converted into subtraction when adding numbers of different signs
 - Need a different circuit to perform addition and subtraction
Increases the cost of the logic circuit

Computing the 2's Complement

starting value	$00100100_2 = +36$
step1: reverse the bits (1's complement)	11011011_2
step 2: add 1 to the value from step 1	$+ 1_2$
sum = 2's complement representation	$11011100_2 = -36$

2's complement of 11011100_2 (-36) = $00100011_2 + 1 = 00100100_2 = +36$

The 2's complement of the 2's complement of N is equal to N

Another way to obtain the 2's complement:

Start at the least significant 1

Leave all the 0s to its right unchanged

Complement all the bits to its left

Binary Value
= 00100 1 00 least significant 1

2 's Complement
= 11011 1 00



Properties of the 2's Complement

- The 2's complement of N is the negative of N
- The sum of N and 2's complement of N must be zero
The final carry is ignored
- Consider the 8-bit number $N = 00101100_2 = +44$
 $-44 = 2\text{'s complement of } N = 11010100_2$
 $00101100_2 + 11010100_2 = \text{1 } 00000000_2$ (8-bit sum is 0)

↑
Ignore final carry

- In general: Sum of $N + 2\text{'s complement of } N = 2^n$
where 2^n is the final carry (1 followed by n 0's)
- There is only one zero: 2's complement of 0 = 0

Sign Extension

Step 1: Move the number into the lower-significant bits

Step 2: Fill all the remaining higher bits with the sign bit

- This will ensure the correctness of the signed value
- Examples

- Sign-Extend 10110011_2 to 16 bits

$$10110011_2 = -77 \rightarrow$$

$$\boxed{11111111} \circledcirc 10110011 = -77$$

- Sign-Extend 01100010_2 to 16 bits

$$01100010_2 = +98 \rightarrow$$

$$\boxed{00000000} \circledcirc 01100010 = +98$$

- Infinite 0's can be added to the left of a positive number
- Infinite 1's can be added to the left of a negative number

Subtraction with 2's Complement

- When subtracting $A - B$, convert B to its 2's complement
- Add A to (2's complement of B)

borrow	-1 -1	-1		carry: 1 1	1 1	
:						
-	0 1 0 0 1 1 0 1			0 1 0 0 1 1 0 1		
	0 0 1 1 1 0 1 0			+	1 1 0 0 0 1 1 0	(2's complement)
					0 0 0 1 0 0 1 1	(same result)

- Final carry is ignored, because
 - Negative number is sign-extended with 1's
 - You can imagine infinite 1's to the left of a negative number
 - Adding the carry to the extended 1's produces extended zeros



Carry and Overflow

- Carry is important when ...
 - Adding or subtracting **unsigned integers**
 - Indicates that the **unsigned sum** is out of range
 - Either < 0 or $>$ maximum unsigned n -bit value
- Overflow is important when ...
 - Adding or subtracting **signed integers**
 - Indicates that the **signed sum** is out of range
- Overflow occurs when
 - Adding two positive numbers and the sum is negative
 - Adding two negative numbers and the sum is positive
 - Can happen because of the fixed number of sum bits

Carry and Overflow Examples



- We can have carry without overflow and vice-versa
- Four cases are possible (Examples are 8-bit numbers)

$$\begin{array}{r} & \text{1} \\ & \hline 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 15 \\ + & \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 8 \\ \hline 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 23 \\ \end{array}$$

Carry = 0 Overflow = 0

$$\begin{array}{r} & \text{1} & \text{1} & \text{1} & \text{1} & \text{1} \\ & \hline 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 15 \\ + & \hline 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 248 (-8) \\ \hline 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 7 \\ \end{array}$$

Carry = 1 Overflow = 0

$$\begin{array}{r} & \text{1} \\ & \hline 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 79 \\ + & \hline 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 64 \\ \hline 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 143 \\ & & & & & & & & (-113) \\ \end{array}$$

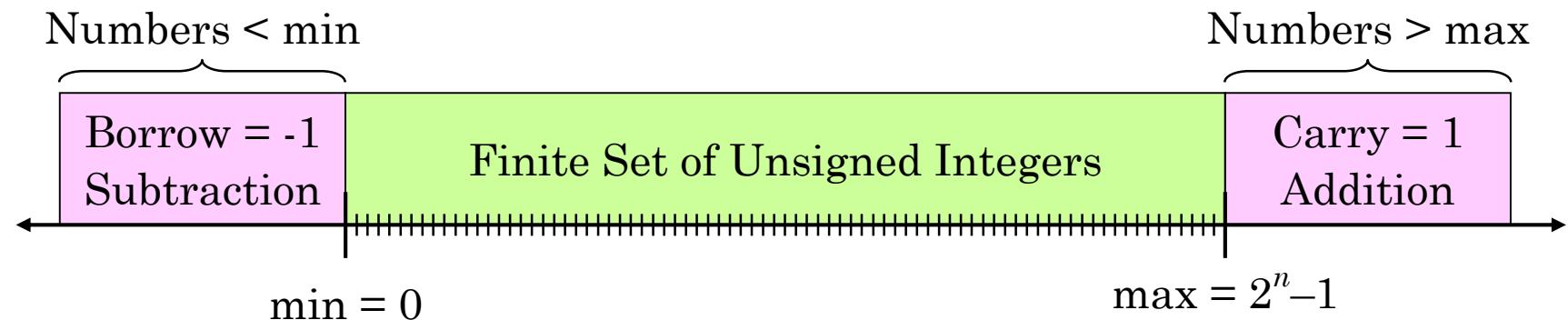
Carry = 0 Overflow = 1

$$\begin{array}{r} & \text{1} & \text{1} & \text{1} \\ & \hline 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 218 (-38) \\ + & \hline 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 157 (-99) \\ \hline 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 119 \\ \end{array}$$

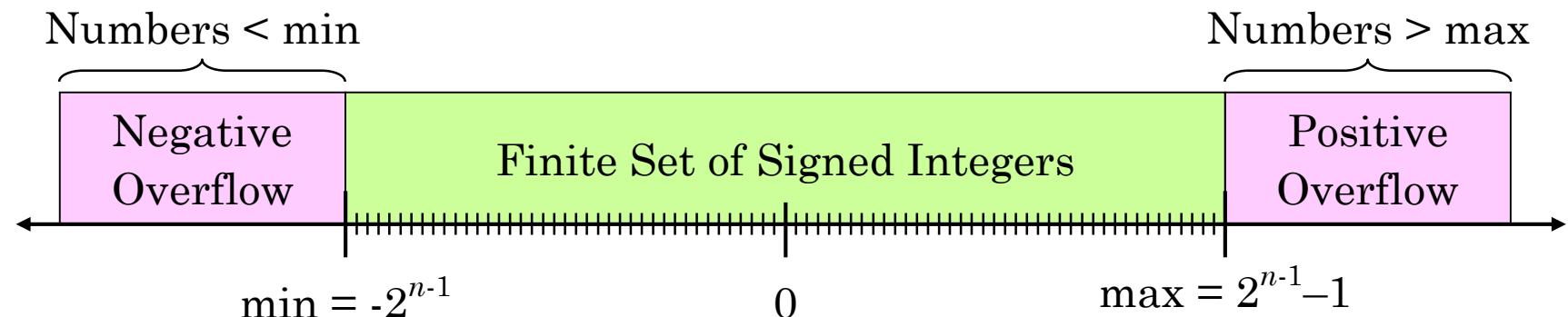
Carry = 1 Overflow = 1

Range, Carry, Borrow, and Overflow

- Unsigned Integers: n -bit representation



- Signed Integers: n -bit 2's complement representation



Signed number ADD

$$6 + 9$$

$$\begin{array}{r} 6 \quad 0110 \\ +) 9 \quad 1001 \\ \hline \end{array}$$

$$15 \quad 1111 \rightarrow 01111$$

$$6 + (-9)$$

$$\begin{array}{r} 9 \quad 1001 \\ -) 6 \quad 0110 \\ \hline \end{array}$$

$$-3 \quad 0011 \rightarrow 10011$$

$$-6 + 9$$

$$\begin{array}{r} 9 \quad 1001 \\ -) 6 \quad 0110 \\ \hline \end{array}$$

$$3 \quad 0011 \rightarrow 00011$$

$$-6 + (-9)$$

$$\begin{array}{r} 6 \quad 0110 \\ +) 9 \quad 1001 \\ \hline \end{array}$$

$$-15 \quad 1111 \rightarrow 11111$$

Overflow

$9 + 9$ or $(-9) + (-9)$

Summation of complement number

$$\begin{array}{r}
 +) \quad 6 \ 0 \ 0110 \\
 9 \ 0 \ 1001 \\
 \hline
 15 \ 0 \ 1111
 \end{array}$$

$$\begin{array}{r}
 +) \quad -6 \ 1 \ 1010 \\
 9 \ 0 \ 1001 \\
 \hline
 3 \ 0 \ 0011
 \end{array}$$

$$\begin{array}{r}
 +) \quad 6 \ 0 \ 0110 \\
 -9 \ 1 \ 0111 \\
 \hline
 -3 \ 1 \ 1101
 \end{array}$$

$$\begin{array}{r}
 +) \quad 9 \ 0 \ 1001 \\
 9 \ 0 \ 1001 \\
 \hline
 18 \ 1 \ 0010
 \end{array}$$

overflow

$$\begin{array}{r}
 +) \quad -9 \ 1 \ 0111 \\
 -9 \ 1 \ 0111 \\
 \hline
 -18 (1)0 \ 1110
 \end{array}$$

$x'_{n-1}y'_{n-1}s_{n-1}$

$x_{n-1}y_{n-1}s'_{n-1}$



Signed number ADD with complement-

- end-around carry

- دو عدد را همراه با بیت علامتشان جمع می کنیم. اگر یک رقم نقلی از پر ارزش ترین بیت خارج شد حاصل جمع را با 1 جمع می کنیم و از رقم نقلی صرفنظر می کنیم.

$$\begin{array}{r}
 +) \quad 6 \ 0 \ 0110 \\
 -9 \ 1 \ 0110 \\
 \hline
 -3 \ 1 \ 1100
 \end{array}$$

$$\begin{array}{r}
 & 11 & 0 & 1011 \\
 +) & -9 & 1 & 0110 \\
 \hline
 & (1) & 0 & 0001 \\
 +) & & \dots & \dots \rightarrow 1 \\
 \hline
 & 2 & 0 & 0010
 \end{array}$$

$$\begin{array}{r}
 +) \quad 9 \quad 0 \quad 1001 \\
 9 \quad 0 \quad 1001 \\
 \hline
 -13 \quad 1 \quad 0010
 \end{array}$$

$$\begin{array}{r}
 +) \quad -9 \quad 1 \quad 0110 \\
 -9 \quad 1 \quad 0110 \\
 \hline
 (1) \quad 0 \quad 1100
 \end{array}$$

$$\begin{array}{r}
 +) \quad \dots \quad \dots \quad \dots \quad \dots \quad \rightarrow 1 \\
 13 \quad 0 \quad 1101
 \end{array}$$

• سریز: $c_{n-1} \oplus c_n$

overflow

Subtraction

- تفریق در سیستم مکمل دو
- مکمل دوی تفریق شونده (همراه با بیت علامت) را گرفته و حاصل را با عملگر دیگر جمع می کنیم.

$$(\pm A) - B = (\pm A) + (-B)$$

$$(\pm A) - (-B) = (\pm A) + B$$

Numbers with the float point

در این نمایش مکان ممیز قسمت کسری ثابت نیست.

طول عدد قابل نمایش در این روش بسیار گسترده است $F = E \cdot M$

m_n	$e_k e_{k-1} \dots e_0$	$m_{n-1} m_{n-2} \dots m_0 \cdot m_{-1} \dots m_{-m}$
نما	علامت	مانتیس

مانتیس: عدد علامت دار با ممیز ثابت (خواه عدد صحیح و یا عدد کسری)

نما: مکان ممیز را مشخص می کند

مقدار معادل دهدهی

$$V(F) = V(M) * RV(E)$$

M: Mantissa, E: Exponent, R: Radix

Numbers with the float point

مثال:

$$\begin{array}{r}
 \text{sign} \\
 0 \quad .1234567 \\
 \hline
 \text{mantissa}
 \end{array}
 \qquad
 \begin{array}{r}
 \text{sign} \\
 0 \quad 04 \\
 \hline
 \text{exponent}
 \end{array}$$

$\Rightarrow +.1234567 \times 10^{+04}$

نکته: در نمایش اعداد به صورت شناور تنها مانتیس (M) و نما (E) صریح‌نمایش داده می‌شوند.
متنا و مکان ممیز به صورت ضمنی استنباط می‌شوند.

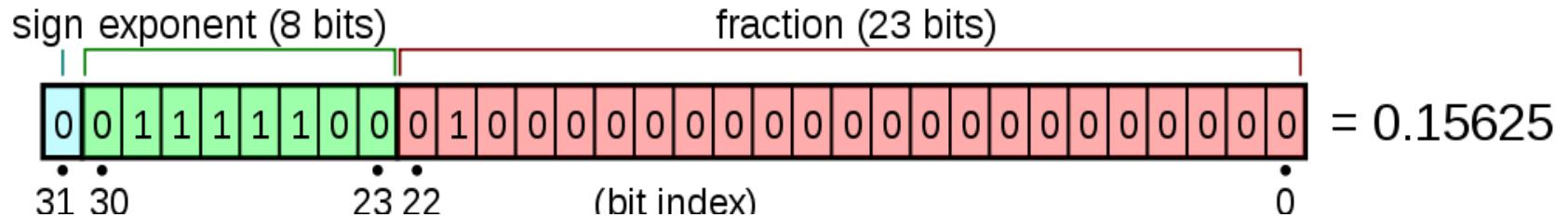
Numbers with the float point

مثال: عدد 1001.11_+ را به صورت یک عدد با ممیز شناور 16 بیتی (6 بیت نما و 10 بیت مانتیس) نمایش دهید.

$$\begin{array}{r}
 & \underline{0} & \underline{0 \ 00100} & \underline{100111000} \\
 & \text{علامة} & \text{نما} & \text{مانتیس} \\
 \underline{+} & \underline{0} & \underline{0 \ 00101} & \underline{010011100}
 \end{array}$$

یا

استاندارد IEEE 754 نمایش



- The real value assumed by a given 32 bit data with a given biased exponent e and a 23 bit fraction is

$$= (-1)^{\text{sign}} (1.b_{-1}b_{-2}\dots b_{-23})_2 \times 2^{e-127}$$

$$\text{value} = (-1)^{\text{sign}} \left(1 + \sum_{i=1}^{23} b_{-i} 2^{-i}\right) \times 2^{(e-127)}$$

Reference

