

NEC 304

STLD

Lecture 3

More Number Systems

Rajeev Pandey

Department Of ECE

rajeevvce2007@gmail.com

Overview

- **Hexadecimal numbers**
 - Related to binary and octal numbers
- **Conversion between hexadecimal, octal and binary**
- **Value ranges of numbers**
- **Representing positive and negative numbers**
- **Creating the complement of a number**
 - Make a positive number negative (and vice versa)
- **Why binary?**

Understanding Binary Numbers

- Binary numbers are made of binary digits (bits):
 - 0 and 1
- How many items does an binary number represent?
 - $(1011)_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = (11)_{10}$
- What about fractions?
 - $(110.10)_2 = 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2}$
- Groups of eight bits are called a *byte*
 - $(11001001)_2$
- Groups of four bits are called a *nibble*.
 - $(1101)_2$

Understanding Hexadecimal Numbers

- Hexadecimal numbers are made of 16 digits:
 - (0,1,2,3,4,5,6,7,8,9,A, B, C, D, E, F)
- How many items does an hex number represent?
 - $(3A9F)_{16} = 3 \times 16^3 + 10 \times 16^2 + 9 \times 16^1 + 15 \times 16^0 = 14999_{10}$
- What about fractions?
 - $(2D3.5)_{16} = 2 \times 16^2 + 13 \times 16^1 + 3 \times 16^0 + 5 \times 16^{-1} = 723.3125_{10}$
- Note that *each* hexadecimal digit can be represented with four bits.
 - $(1110)_2 = (E)_{16}$
- Groups of four bits are called a *nibble*.
 - $(1110)_2$

Putting It All Together

Decimal	Binary	Octal	Hexadecimal
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

- **Binary, octal, and hexadecimal similar**
- **Easy to build circuits to operate on these representations**
- **Possible to convert between the three formats**

Converting Between Base 16 and Base 2

$$3A9F_{16} = \begin{array}{cccc} \underline{0011} & \underline{1010} & \underline{1001} & \underline{1111} \\ 3 & A & 9 & F \end{array}_2$$

- **Conversion is easy!**
 - **Determine 4-bit value for each hex digit**
- **Note that there are $2^4 = 16$ different values of four bits**
- **Easier to read and write in hexadecimal.**
- **Representations are equivalent!**

Converting Between Base 16 and Base 8

$$3A9F_{16} = \underline{0011} \ \underline{1010} \ \underline{1001} \ \underline{1111}_2$$

3 A 9 F



$$35237_8 = \underline{011} \ \underline{101} \ \underline{010} \ \underline{011} \ \underline{111}_2$$

3 5 2 3 7

1. Convert from Base 16 to Base 2
2. Regroup bits into groups of three starting from right
3. Ignore leading zeros
4. Each group of three bits forms an octal digit.

How To Represent Signed Numbers

- Plus and minus sign used for decimal numbers: 25 (or +25), -16, etc.
- For computers, desirable to represent everything as *bits*.
- Three types of signed binary number representations: **signed magnitude, 1's complement, 2's complement**.
- In each case: **left-most bit indicates sign: positive (0) or negative (1).**

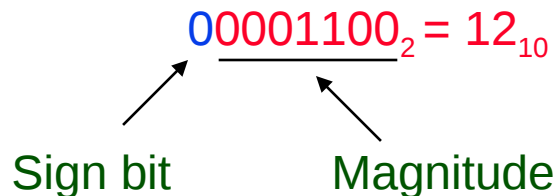
Consider *signed magnitude*:

$$\begin{array}{c} \text{Sign bit} \nearrow \text{0} \text{0001100}_2 = 12_{10} \\ \nwarrow \text{Magnitude} \end{array}$$

$$\begin{array}{c} \text{Sign bit} \nearrow \text{1} \text{0001100}_2 = -12_{10} \\ \nwarrow \text{Magnitude} \end{array}$$

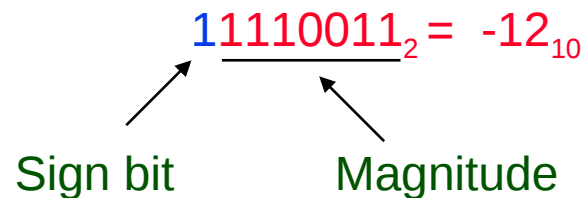
One's Complement Representation

- The one's complement of a binary number involves inverting all bits.
 - 1's comp of 00110011 is **11001100**
 - 1's comp of 10101010 is **01010101**
- For an n bit number **N** the 1's complement is $(2^n - 1) - N$.
- Called diminished radix complement by Mano since 1's complement for base (radix 2).
- To find negative of 1's complement number take the 1's complement.

The diagram shows the binary representation of the decimal number 12 in 1's complement. The sign bit is 0, and the magnitude is 0001100. The entire 8-bit sequence is underlined. Arrows point from the labels 'Sign bit' and 'Magnitude' to their respective parts of the binary number.

$$\underline{00001100}_2 = 12_{10}$$

Sign bit Magnitude

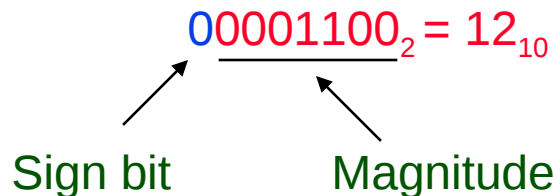
The diagram shows the binary representation of the decimal number -12 in 1's complement. The sign bit is 1, and the magnitude is 1110011. The entire 8-bit sequence is underlined. Arrows point from the labels 'Sign bit' and 'Magnitude' to their respective parts of the binary number.

$$\underline{11110011}_2 = -12_{10}$$

Sign bit Magnitude

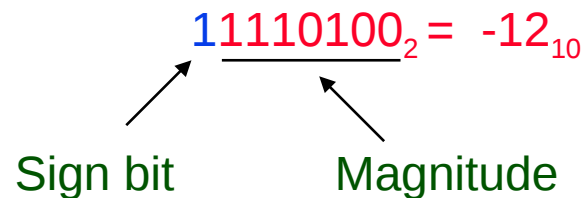
Two's Complement Representation

- The two's complement of a binary number involves inverting all bits **and adding 1**.
 - 2's comp of 00110011 is **11001101**
 - 2's comp of 10101010 is **01010110**
- For an n bit number **N** the 2's complement is $(2^n - 1) - N + 1$.
- Called radix complement by Mano since 2's complement for base (radix 2).
- To find negative of 2's complement number take the 2's complement.

The diagram shows the binary number 00001100₂ = 12₁₀. The first bit '0' is blue and labeled 'Sign bit' with an arrow. The remaining bits '0001100' are red and labeled 'Magnitude' with an arrow.

$$\text{00001100}_2 = 12_{10}$$

Sign bit Magnitude

The diagram shows the binary number 11110100₂ = -12₁₀. The first bit '1' is blue and labeled 'Sign bit' with an arrow. The remaining bits '1110100' are red and labeled 'Magnitude' with an arrow.

$$\text{11110100}_2 = -12_{10}$$

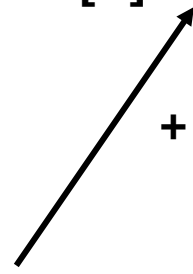
Sign bit Magnitude

Two's Complement Shortcuts

- Algorithm 1 – Simply complement each bit and then add 1 to the result.

- Finding the 2's complement of $(01100101)_2$ and of its 2's complement...

N	=	01100101	[N]	=	10011011
		10011010			01100100
		+ 1			+ 1
		-----			-----
		10011011			01100101



- Algorithm 2 – Starting with the least significant bit, copy all of the bits up to and including the first 1 bit and then complementing the remaining bits.

- $N = 01100101$
 $[N] = 10011011$

Finite Number Representation

- ° Machines that use 2's complement arithmetic can represent integers in the range

$$-2^{n-1} \leq N \leq 2^{n-1}-1$$

where n is the number of bits available for representing N . Note that $2^{n-1}-1 = (011..11)_2$ and $-2^{n-1} = (100..00)_2$

- ° For 2's complement more negative numbers than positive.
- ° For 1's complement two representations for zero.
- ° For an n bit number in base (radix) z there are z^n different **unsigned** values.

$$(0, 1, \dots, z^{n-1})$$

1's Complement Addition

- Using 1's complement numbers, adding numbers is easy.
- For example, suppose we wish to add $+(1100)_2$ and $+(0001)_2$.
- Let's compute $(12)_{10} + (1)_{10}$.
 - $(12)_{10} = +(1100)_2 = 01100_2$ in 1's comp.
 - $(1)_{10} = +(0001)_2 = 00001_2$ in 1's comp.

Step 1: Add binary numbers

Step 2: Add carry to low-order bit

		0	1	1	0	0
		0	0	0	0	1

	0	0	1	1	0	1
Add carry		└───────────────────> 0				

Final Result		0	1	1	0	1

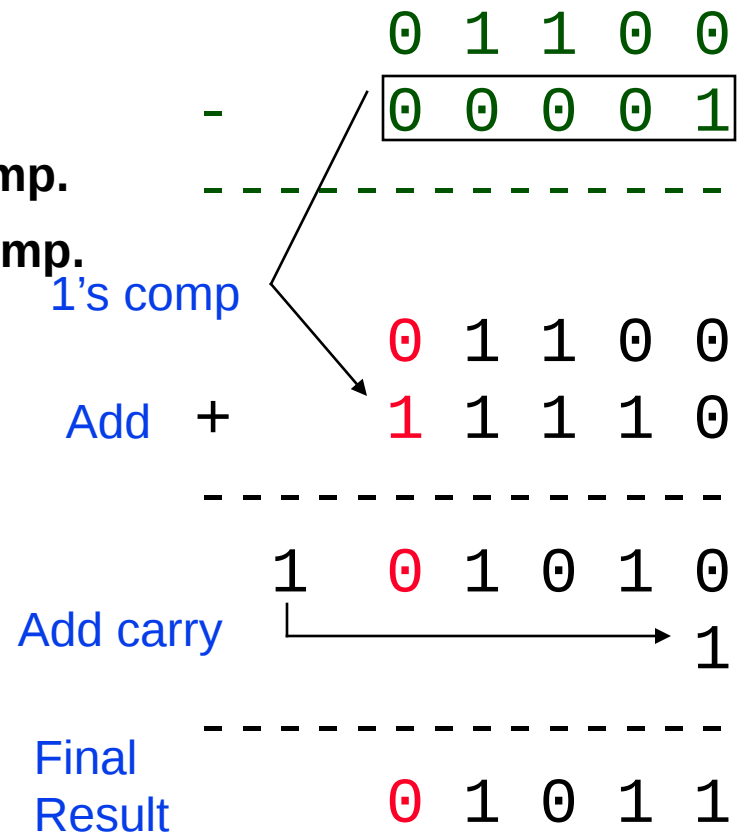
1's Complement Subtraction

- Using 1's complement numbers, subtracting numbers is also easy.
- For example, suppose we wish to **subtract** $+(0001)_2$ from $+(1100)_2$.
- Let's compute $(12)_{10} - (1)_{10}$.
 - $(12)_{10} = +(1100)_2 = 01100_2$ in 1's comp.
 - $(-1)_{10} = -(0001)_2 = 11110_2$ in 1's comp.

Step 1: Take 1's complement of 2nd operand

Step 2: Add binary numbers

Step 3: Add carry to low order bit



2's Complement Addition

- Using 2's complement numbers, adding numbers is easy.
- For example, suppose we wish to add $+(1100)_2$ and $+(0001)_2$.
- Let's compute $(12)_{10} + (1)_{10}$.
 - $(12)_{10} = +(1100)_2 = \textcolor{red}{0}1100_2$ in 2's comp.
 - $(1)_{10} = +(0001)_2 = \textcolor{red}{0}0001_2$ in 2's comp.

Step 1: Add binary numbers

Step 2: Ignore carry bit

Add

+

0	1	1	0	0
0	0	0	0	1

Final Result

0

Ignore

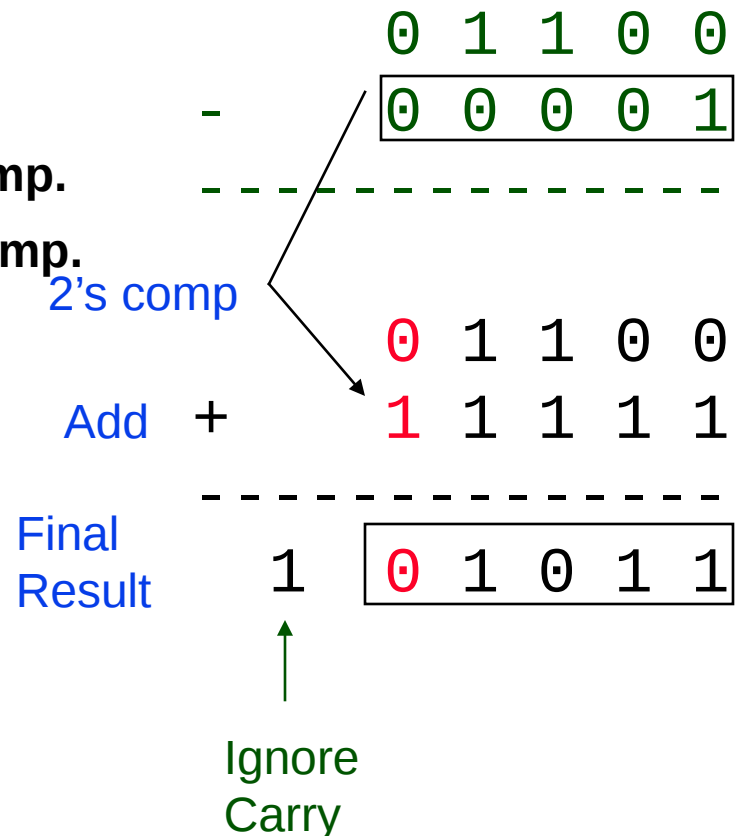
2's Complement Subtraction

- Using 2's complement numbers, follow steps for subtraction
- For example, suppose we wish to subtract $+(0001)_2$ from $+(1100)_2$.
- Let's compute $(12)_{10} - (1)_{10}$.
 - $(12)_{10} = +(1100)_2 = 01100_2$ in 2's comp.
 - $(-1)_{10} = -(0001)_2 = 11111_2$ in 2's comp.

Step 1: Take 2's complement of 2nd operand

Step 2: Add binary numbers

Step 3: Ignore carry bit



2's Complement Subtraction: Example #2

◦ Let's compute $(13)_{10} - (5)_{10}$.

• $(13)_{10} = +(1101)_2 = (01101)_2$

• $(-5)_{10} = -(0101)_2 = (11011)_2$

◦ Adding these two 5-bit codes...

				0	1	1	0	1
	+			1	1	0	1	1
				-	-	-	-	-
carry			→	1	0	1	0	0

◦ Discarding the carry bit, the sign bit is seen to be zero, indicating a correct result. Indeed,

$$(01000)_2 = +(1000)_2 = +(8)_{10}.$$

2's Complement Subtraction: Example #3

- **Let's compute $(5)_{10} - (12)_{10}$.**
 - $(-12)_{10} = -(1100)_2 = (10100)_2$
 - $(5)_{10} = +(0101)_2 = (00101)_2$
- **Adding these two 5-bit codes...**

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

- Here, there is no carry bit and the sign bit is 1. This indicates a negative result, which is what we expect. $(11001)_2 = -(7)_{10}$.

Summary

- **Binary numbers can also be represented in octal and hexadecimal**
- **Easy to convert between binary, octal, and hexadecimal**
- **Signed numbers represented in signed magnitude, 1's complement, and 2's complement**
- **2's complement most important (only 1 representation for zero).**
- **Important to understand treatment of sign bit for 1's and 2's complement.**

