Chapter 2

Number systems

Dinh Duc Anh Vu
International University – VNU HCM

Number Systems

- Number representations
- Common codes
- Computations (Digital arithmetic)

Objectives

- Recognize the basic characteristics of the binary number system.
- Convert a binary number to its decimal equivalent.
- Count in the binary number system
- Convert a number from one number system (decimal, binary, hexadecimal) to its equivalent in one of the other number systems.
- Cite the advantages of the hexadecimal number system.
- Count in hexadecimal.
- Represent decimal numbers using the BCD code; cite the pros and cons of using BCD.
- Understand the difference between BCD and straight binary.
- Understand the purpose of alphanumeric codes such as the ASCII code.
- Perform binary addition, subtraction, multiplication, and division on two binary numbers.
- Compare the advantages and disadvantages among three different systems of representing signed binary numbers.
- Manipulate signed binary numbers using the 2's-complement system.
- Understand the BCD addition process

Number Representations

- Decimal system
- Binary system
- Hexadecimal system

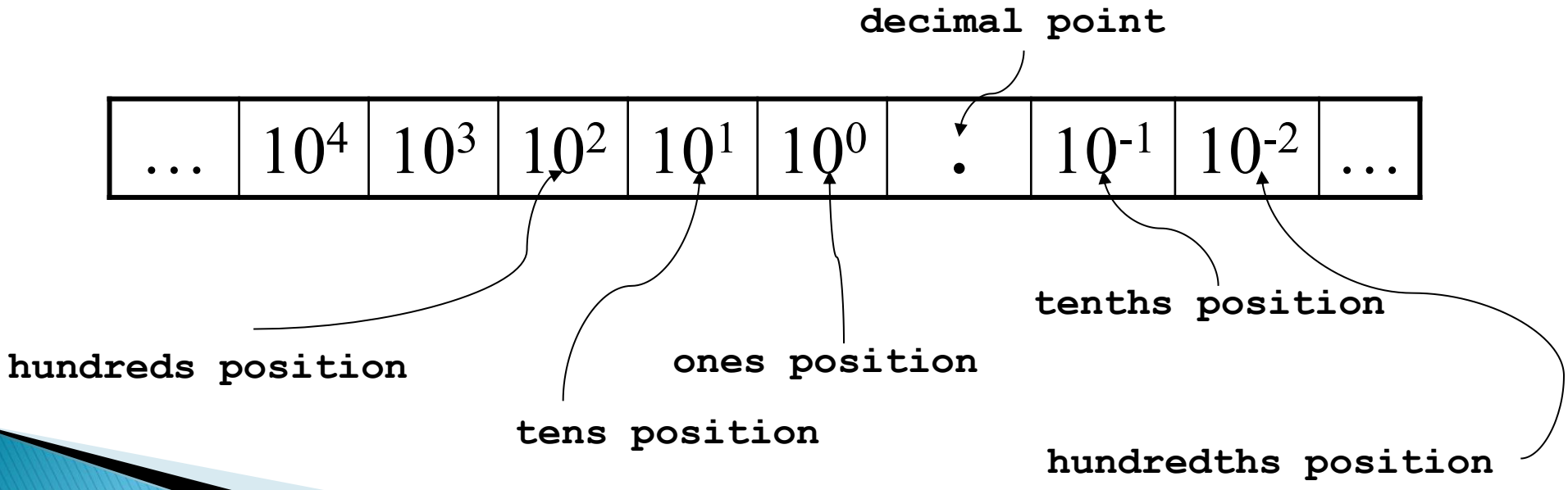
Number representations

453

1011_2

25.3_{10}

- A number in the number system is:
  - created by one or more digits.
  - comprised of the integer part and fractional part separated by the radix point (or base point)
  - Positional number system: Each digit carries a certain weight based on its position



Positional number system

- The weight of each digit: The weight = $base^{position}$
- The last digit on the left: the most significant digit (MSD)
- The last digit on the right: the least significant digit (LSD)
- The value of the number = Σ(digit * its weight)

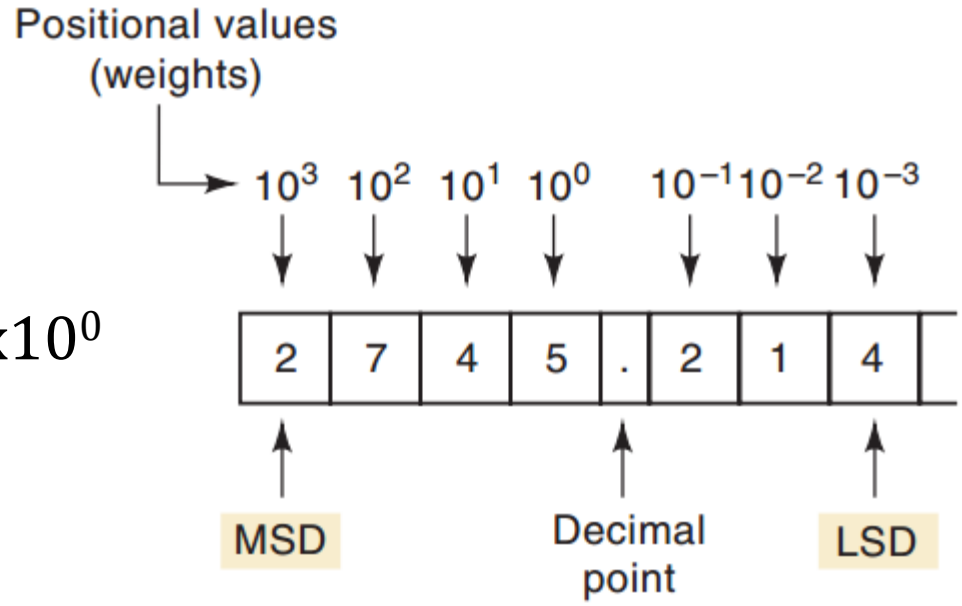
Number representations

1. Decimal system
2. Binary system
3. Hexadecimal system
4. Octal system

Decimal system

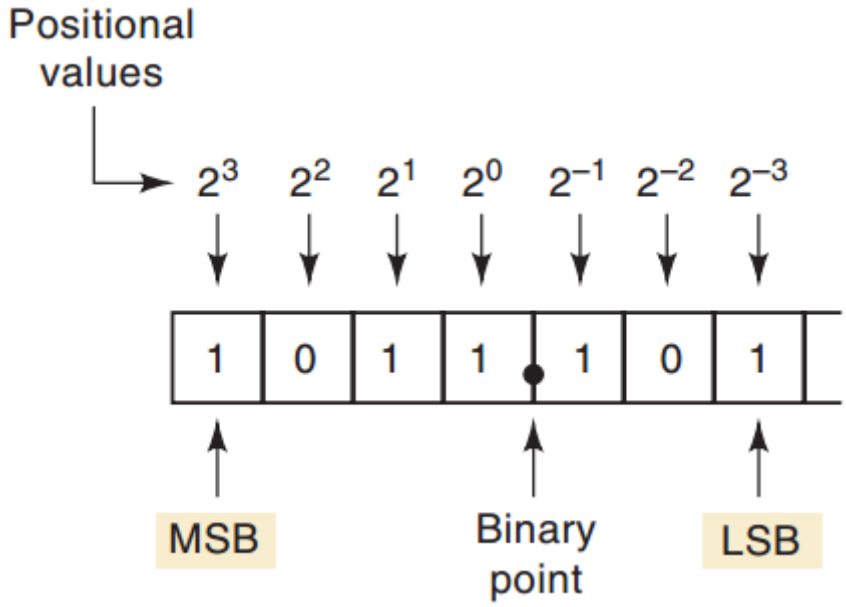
- Symbol: D
- Base: 10
- Digits used: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- The decimal system is a positional-value system in which the value of a digit depends on its position.
- Example:

$$2745.214 = 2x10^{3}+7x10^{2}+4x10^{1}+5x10^{0}+2x10^{-1}+1x10^{-2}+4x10^{-3}$$



Binary system

- Symbol: B
- Base: 2
- Digits used: 0, 1
- Example:
  - 1011.101B



$$1011.101_2 = (1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) + (1 \times 2^{-1}) + (0 \times 2^{-2}) + (1 \times 2^{-3})$$
$$= 8 + 0 + 2 + 1 + 0.5 + 0 + 0.125$$
$$= 11.625_{10}$$

Binary system to Decimal system

- Convert the number in binary system into decimal system, i.e., calculate the value of the binary number
- Any binary number can be converted to its decimal equivalent simply by summing together the weights of the various positions in the binary number that contain a 1
- Example:

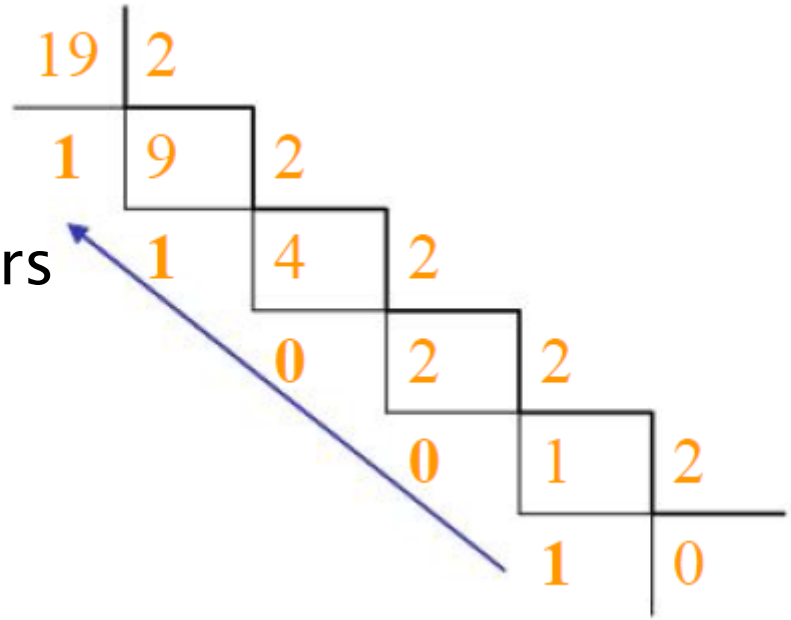
$$\begin{array}{cccccc} 1 & 1 & 0 & 1 & 1_2 & \\ 2^4 + 2^3 + 0 + 2^1 + 2^0 & = & 16 + 8 + 2 + 1 & & & \\ & = & 27_{10} & & & \end{array}$$

Decimal system to Binary system

- The first method is the reverse of the previous process: The decimal number is simply expressed as a sum of powers of 2, and then 1s and 0s are written in the appropriate bit positions

$$76_{10} = 64 + 8 + 4 = 2^6 + 0 + 0 + 2^3 + 2^2 + 0 + 0$$
$$= 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0_2$$

- Another method for converting decimal integers uses repeated division by 2: divide the integer by 2 until the quotient is zero. Then, collect the remainders **upward**
  => 19D = 10011B



Decimal system to Binary system – Fraction

- Convert the number in decimal system into binary system ⇒ For a fraction: multiply the fraction and 2, then taking out the integer part of the product.
- Repeat the multiplication of the fraction and 2, until the fraction is zero or reaches to an accuracy accepted. The binary obtained is the downward collections of removed integers.
- Example: Convert 0.8125D into binary system

  0.8125 x 2 = 1.625 → take out 1
  0.625 x 2 = 1.25 → take out 1
  0.25 x 2 = 0.5 → take out 0
  0.5 x 2 = 1.0 → take out 1
  ⇒ The binary obtained is 0.1101B

Decimal system to Binary system

- Example: Convert 12.69D into Binary system

The integer part: 12D = 1100B

The fraction:

0.69 x 2 = 1.38 → take out 1

0.38 x 2 = 0.76 → take out 0

0.76 x 2 = 1.52 → take out 1

0.52 x 2 = 1.04 → take out 1

0.04 x 2 = 0.08 → take out 0

Result: 12.69D ≈1100.10110B

Hexadecimal system

- Symbol: H
- Base: 16
- Digit used: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
  - The digits A, B, C, D, E, F are worth 10, 11, 12, 13, 14, 15, respectively
- Calculate the value of the hexadecimal number 2A9H

$$
\begin{aligned}
2A9H &= 2\times16^2 + 10\times16^1 + 9\times16^0 \\
&= 2\times256 + 10\times16 + 9\times1 \\
&= 681D
\end{aligned}
$$

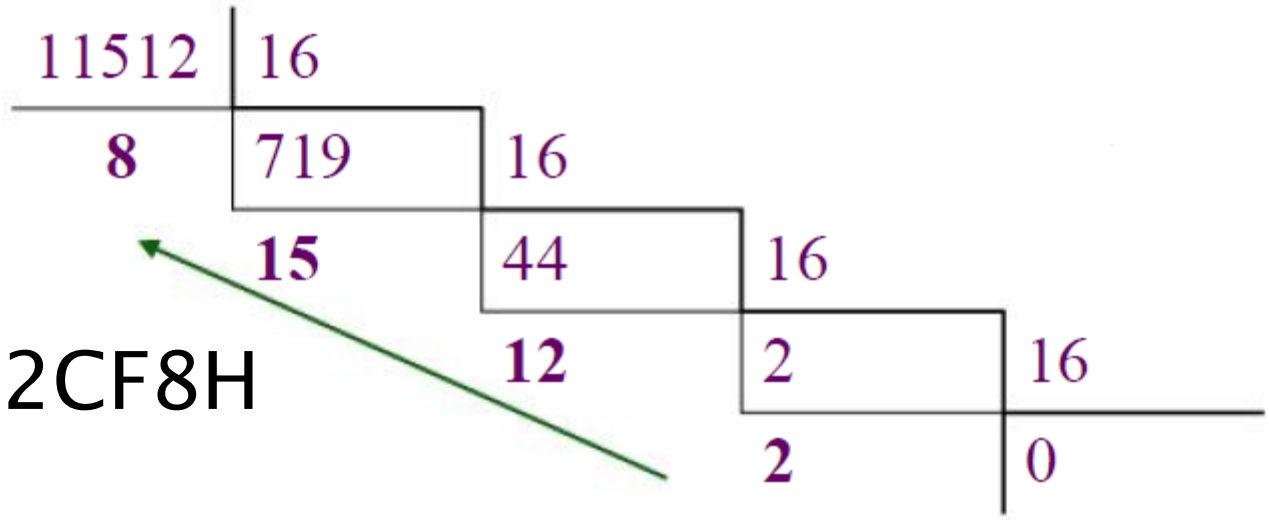
- Hexadecimal numbers are often used to describe a computer's memory address space
  - 0xF000 – 0xFFFF

Decimal system to Hexadecimal system

- Convert the number in decimal system into hexadecimal system

  $\Rightarrow$ Divide the decimal number by 16 until the quotient is zero. The hexadecimal obtained is the upward collections of the remainders

- Example: Convert 11512D into hexadecimal system.



Result: 11512D = 2CF8H

Hexadecimal system to Binary system

- Convert the number in hexadecimal system into binary system: One digit in hexadecimal system is equivalent to 4 bits in binary system

3D6FH = 0011 1101 0110 1111B

Hexadecimal	Decimal	Binary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

Binary system to Hexadecimal system

- Convert the number in binary system into hexadecimal system: From the unit column, group 4 bits of binary number together, then convert each group into the corresponding hexadecimal digit

$$1\,1\,1\,0\,1\,0\,0\,1\,1\,0_2 = \underbrace{0\,0\,1\,1}_{3}\ \underbrace{1\,0\,1\,0}_{A}\ \underbrace{0\,1\,1\,0}_{6}$$

$$= 3A6_{16}$$

Summary of Conversion

- When converting from binary [or hex] to decimal, use the method of taking the weighted sum of each digit position.
- When converting from decimal to binary [or hex], use the method of repeatedly dividing by 2 [or 16] and collecting remainders (Figure 2-1).
- When converting from binary to hex, group the bits in groups of four, and convert each group into the correct hex digit.
- When converting from hex to binary, convert each digit into its four-bit equivalent.

Code: group of special symbols that represent numbers, letters, words

Morse code: series of dots and dashes represents letters of alphabet

Straight binary coding

Common Codes

- BCD Code
- Excess-3 Code
- Gray Code
- Alphanumeric Code

Code word: particular combination of n bit-values that represents different numbers

Common Codes

Decimal digit	BCD (8421)	2421	Excess-3	Biquinary	1-out-of-10
0	0000	0000	0011	0100001	1000000000
1	0001	0001	0100	0100010	0100000000
2	0010	0010	0101	0100100	0010000000
3	0011	0011	0110	0101000	0001000000
4	0100	0100	0111	0110000	0000100000
5	0101	1011	1000	1000001	0000010000
6	0110	1100	1001	1000010	0000001000
7	0111	1101	1010	1000100	0000000100
8	1000	1110	1011	1001000	0000000010
9	1001	1111	1100	1010000	0000000001
		Unused code words			
	1010	0101	0000	0000000	0000000000
	1011	0110	0001	0000001	0000000011
	1100	0111	0010	0000010	0000000101
	1101	1000	1101	0000011	0000000110
	1110	1001	1110	0000101	0000000111
	1111	1010	1111	...	...

BCD Code

$b_3\ b_2\ b_1\ b_0$

$2^3\ 2^2\ 2^1\ 2^0$

$8\ 4\ 2\ 1$

- Binary Coded Decimal (BCD 8421)
  - Used to represent the decimal digits 0 – 9.
  - 4 bits are used.
  - Each bit position has a weight associated with it (**weighted code**).
  - Weights are: 8, 4, 2, and 1 from MSB to LSB (called 8-4-2-1 code).

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

- BCD Codes:
  - Used to encode numbers for output to numerical displays
  - Used in processors that perform decimal arithmetic.
    - Example: $(9750)_{10} = (1001\ 0111\ 0101\ 0000)_{BCD}$

BCD Code

- Each digit of a decimal number is represented by its binary equivalent
- Examples

```
  8       7       4       (decimal)
  ↓       ↓       ↓
1000    0111    0100      (BCD)

  9       4       3       (decimal)
  ↓       ↓       ↓
1001    0100    0011      (BCD)
```

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

BCD – Examples

- Convert 0110100000111001 (BCD) to its decimal equivalent

  0110 1000 0011 1001

  6 8 3 9

- Convert the BCD number 011111000001 to its decimal equivalent

  0111 1100 0001

  7 Error 1

(The forbidden code group indicated an error)

BCD 2421

- is one kind of BCD code
- used to represent a decimal number
- is a 4-bit binary combination with weights 2-4-2-1.
- characterized by complement of "base-1"
  - Number 0 has 2421 code is 0000, complements with number 9 having 2421 code is 1111
  - Number 3 has 2421 code is 0011, complements with number 6 having 2421 code is 1100
- The binary combinations from 0101 to 1010 are not included in BCD 2421

Comparison of BCD and Binary

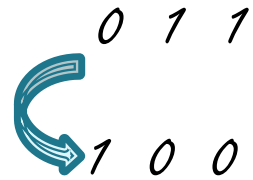
- A straight binary code takes the complete decimal number and represents it in binary
- A BCD code converts each decimal digit to binary individually
- Examples
  - $178_{10} = 10110010_2$ (8 bit)
  - $178_{10} = 0001\ 0111\ 1000_{BCD}$ (12 bit)

- BCD uses more bits, but easier to convert to and from decimal

Excess 3 Code

- Created from 8421 code by adding 3.
- Has **no weight** → not suited for arithmetic operations, but similar to BCD 8421 code
- Represent a decimal digit
- Characterized by base complement
- Example:
  - Number 0 has the Excess 3 code is 0011, complements with number 9 having the Excess 3 code is 1100
  - Number 4 has the Excess 3 code is 0111, complements with number 5 having the Excess 3 code is 1000

Gray Code

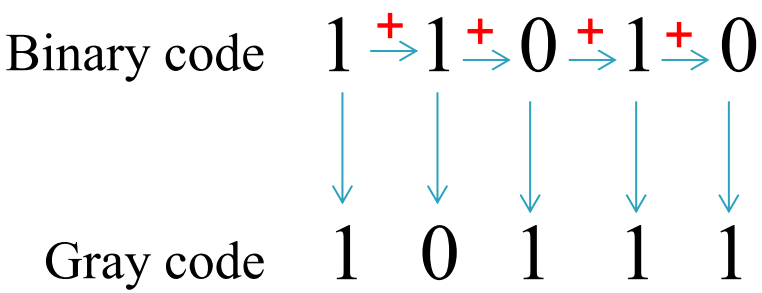


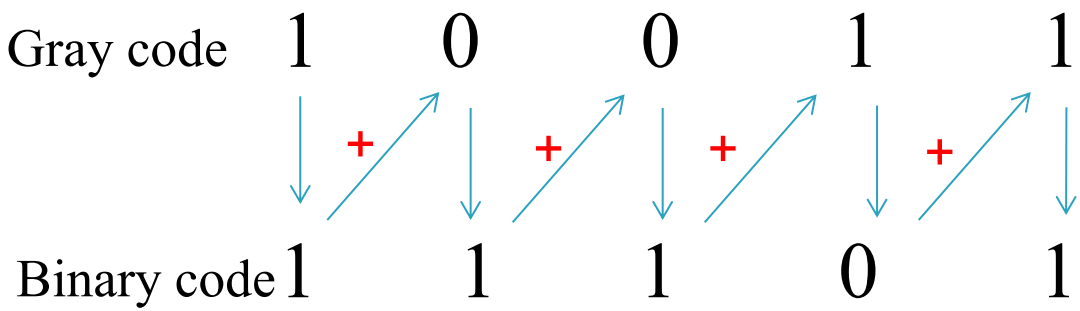
- is not BCD Code.
- has **no weight** → not suited for arithmetic operations
- **only one bit ever changes between two successive numbers in the sequence**
- created from the binary code based on the following principle:
  - MSB of Gray code and that of binary code is the same.
  - Add MSB of binary number into the right bit and write the sum (ignore the carrier)
  - Continue to LSB.
  - The numbers of bits in Gray Code are the same to that in binary code.

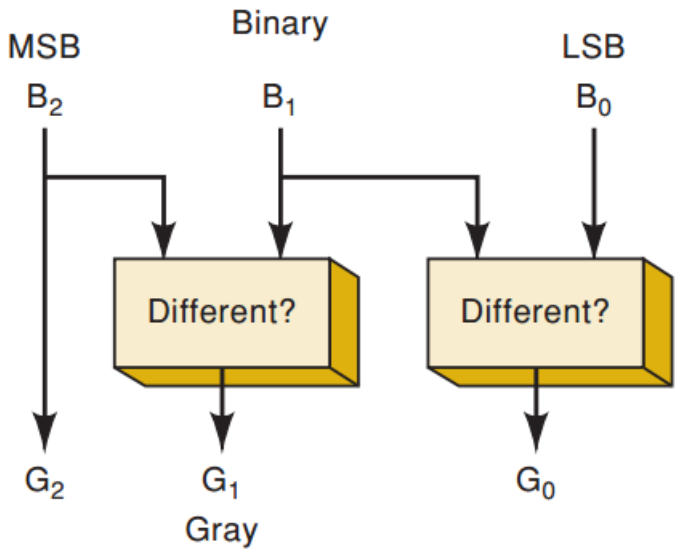
B_2	B_1	B_0	G_2	G_1	G_0
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

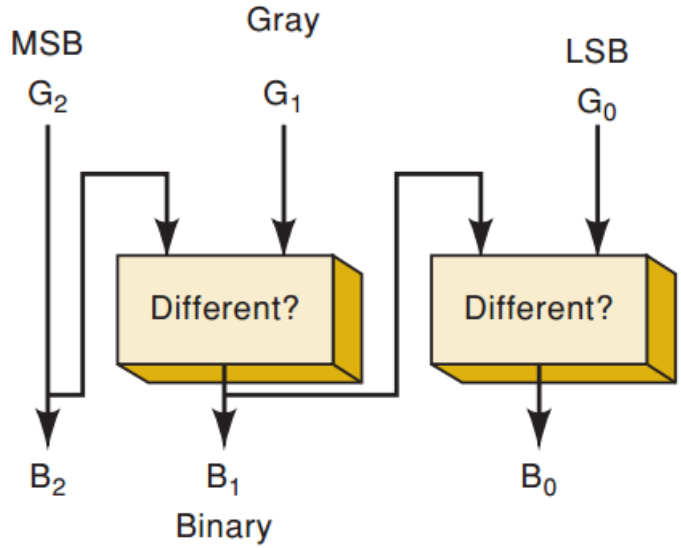
Gray Code conversion

- Different:
  - If inputs are the same, then G=0.
  - If inputs are different, then G=1
- The bit following bit 0 of binary code is not changed, the bit following bit 1 of binary code is reversed









Alphanumeric Code

- Codes representing letters of the alphabet, punctuation marks, and other special characters as well as numbers are called alphanumeric codes
- The most widely used alphanumeric code is the American Standard Code for Information Interchange (ASCII). The ASCII (pronounced “askee”) code is a 7-bit code
- Baudot Code

Baudot Code

Value	LTRS shift	FIGS shift	Value	LTRS shift	FIGS shift
3	A	-	23	Q	1
25	B	?	10	R	4
14	C	:	5	S	.
9	D	Who are u	16	T	5
1	E	3	7	U	7
13	F	!	30	V	;
26	G	&	19	W	2
20	H	#	29	X	/
6	I	8	21	Y	6
11	J	Bell	17	Z	"
15	K	(	0	BLANK	BLANK
18	L	)	31	LTRS	LTRS
28	M	.	27	FIGS	FIGS
12	N	,	4	SPACE	SPACE
24	O	9	8	CR	CR
22	P	0	2	LF	LF

"JAMES BOND 007 SAYS HI!"

J	A	M	E	S		B	O	N	D		0	0	7		S	A	Y	S		H	I	!
31	11	3	28	1	5	4	25	24	12	9	4	27	22	22	7	4	31	5	3	21	5	4	20	6	27	13

ASCII Code

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	TAB	LF	VT	FF	CR	SO	SI
10	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
20		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Review Questions

- The following is a message encoded in ASCII code. What is the message?
  - 1001000 1000101 1001100 1010000
  - HELP
- Encode the following message in ASCII code using the hex representation: “Cost=$72.”
  - 43 6F 73 74 3D 24 37 32 2D
- The following padded ASCII-coded message is stored in successive memory location in a computer: 01010011 01010100 01001111 01010000. What is the message?
  - STOP

Computations

- Representing signed numbers
- Addition & Subtraction in the 2's complement system
- BCD addition

"base-1" Complement

- Given number N consisting of n digits in the base system r, complement of "base -1" of N is defined as $(r^n - 1 - N)$.
- Example:
  - Complement 9 of a decimal number 123D is: $10^3 - 1 - 123 = 999 - 123 = 876D$
  - Complement 1 of a binary number 1100B is: $2^4 - 1 - 12 = 15 - 12 = 3 = 0011B$
  - Complement 15 of a hexadecimal number 2CH is: $16^2 - 1 - 44 = 255 - 44 = 211 = D3H$

9's Complement of a decimal

- To find complement 9 of a decimal number, take 9 minus each digit
- Example: Find complement 9 of 270284D

```
  9  9  9  9  9  9
- 2  7  0  2  8  4
-------------------
  7  2  9  7  1  5
```

Result: Complement 9 of 270284 is 729715

1's Complement of a binary

- To find complement 1 of a binary, change bit 1 to bit 0 and change bit 0 to bit 1
- Example:
  - Complement 1 of 1100110B is 0011001B
  - Complement 1 of 01101110B is 10010001B

Base Complement

- Given number N consisting of n digits in the base system r, base complement of N is defined:
  - $r^n - N$ if $N \neq 0$
  - 0 if $N = 0$
- Example
  - Complement 10 of a decimal number 123D is: $10^3 - 123 = 1000 - 123 = 877D$
  - Complement 2 of a binary number 1100B is: $2^4 - 12 = 16 - 12 = 4 = 0100B$
  - Complement 16 of a hexadecimal number 2CH is: $16^2 - 44 = 256 - 44 = 212 = D4H$

Base complement

- Base complement can also be calculated by adding 1 in the complement of "base -1"
- Example
  - Complement 9 of 456D is 543D
    - Complement 10 of 456D is 543 + 1 = 544D
  - Complement 1 of 1011B is 0100B
    - Complement 2 of 1011B is 0100B + 1 = 0101B
  - Complement 1 of 1000B is 0111B
    - Complement 2 of 1000B is 0111B + 1 = 1000B

Computations in binary code systems

- Unsigned binary number
  - The value of unsigned binary number ≥ 0.
  - An unsigned binary number n bits expresses value in range of 0 to $2^n - 1$.
  - Example:
    - 1010B = 10D, in range of 0 to $2^4 - 1$ (15)
    - 01100001B = 97D, in range of 0 to $2^8 - 1$ (255)

Computations in binary code systems

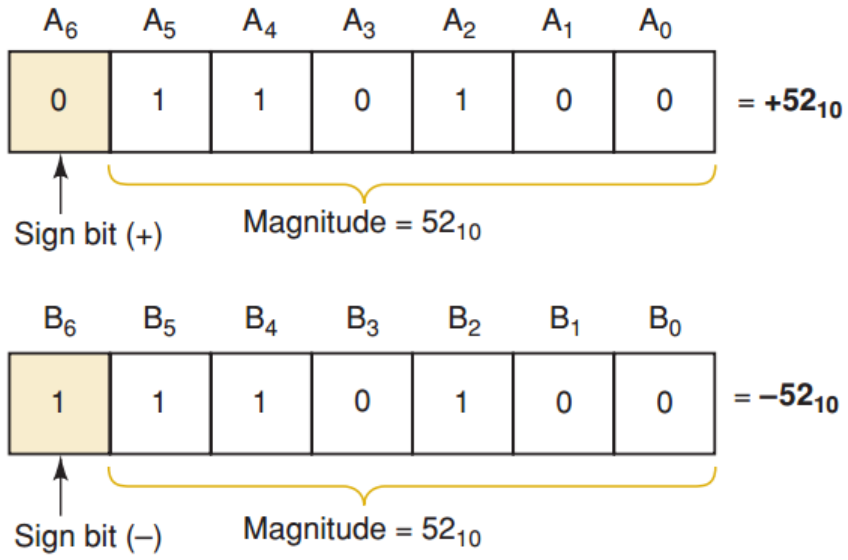
- Signed binary number
  - Three ways to represent signed numbers

1. Sign-Magnitude
2. 1's Complement
3. 2's Complement

Sign-Magnitude

+98 -67 +123.7 -13.72

- A number consists of a magnitude and a symbol indicating whether the magnitude is positive or negative
- For binary system
  - MSB is used as the sign bit (0 – plus, 1 – minus)
  - The magnitude bits are the true binary equivalent of the decimal value being represented
  - Two values for zero: 0000 and 1111
  - Range for n bits:
    $-(2^{n-1}-1)$ through $+(2^{n-1}-1)$



Representing Signed Numbers Using 1's Complement

- The positive number is represented as in the common binary form, and the MSB is "0".
- The negative number is obtained by find the 1's complement of the corresponding positive number.
- Example:

- Two values for zero: 0000 and 1111
- Range for n bits: $-(2^{n-1}-1)$ through $+(2^{n-1}-1)$

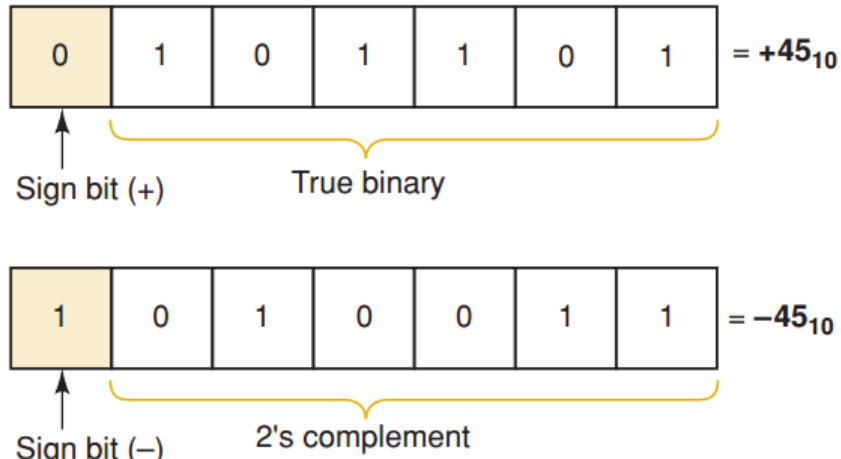
Representing Signed Numbers Using 2's Complement

- The positive number is represented as in the common binary form, and the MSB is "0".
- The negative number is obtained by find the 2's complement of the corresponding positive number.
- Example: 0110B = +6D

⇒2's complement of 0110B is 1010B

$\Rightarrow 1010_B \overset{2's\ complement}{\Longleftrightarrow} -6_D$

- Only one representation of zero
- Range for n bits: -2^{n-1} through $+(2^{n-1}-1)$



2's Complement – example

- Represent –9D in 2's complement form
  - +9D = 01001B
  - 2's complement of 01001B is 10111B
  - So –9D = 10111B

- Convert –14D into signed binary number
  - +14D = 01110B
  - Complement 2 of 01110B is 10010B

⇒ –14D = 10010B

- Convert the signed binary number 11101B into Decimal number
  - Complement 2 of 11101B is 00011B = +3D

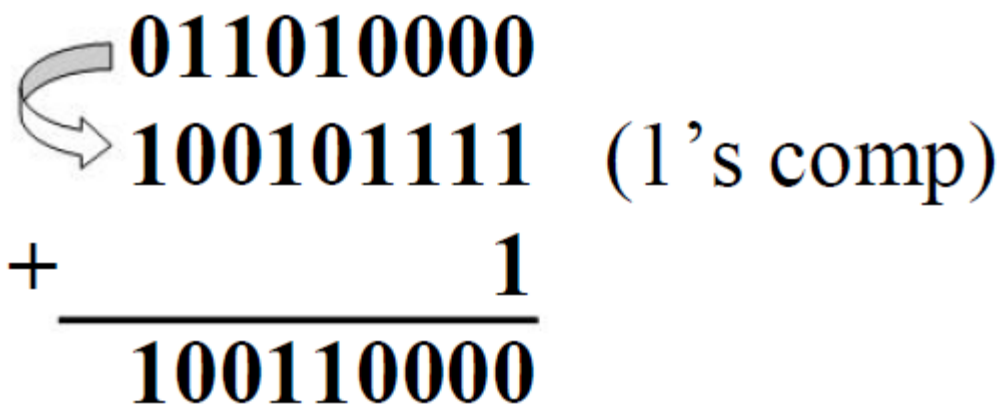
⇒ 11101B = –3D

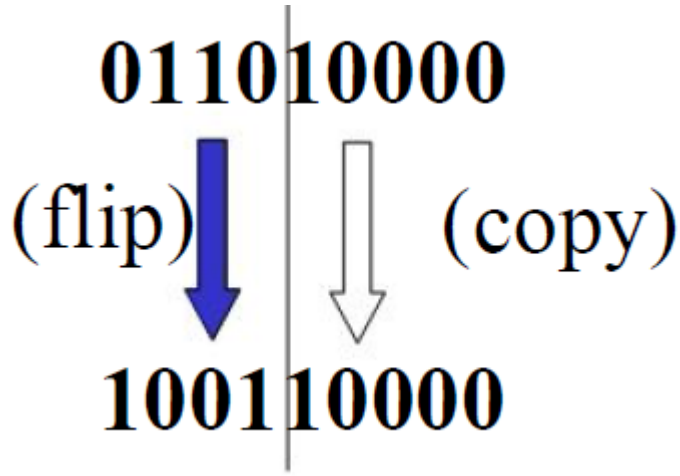
Signed extension

- The MSB of the negative numbers in the signed binary system = 1
- For positive numbers, adding bits 0 to the left (in front of MSB) does not change the number's value.
- For negative numbers, adding bits 1 to the left (in front of MSB) does not change the number's value.
- Copy sign bit to the left is called sign extension
- Example:
  - 0110B = 00110B = 000110B = … = +6D
  - 1010B = 11010B = 111010B = … = -6D

Negation

- Another way to take the two's complement of a number:
  - copy bits from right to left until (and including) the first "1"
  - flip remaining bits to the left
- To negate a signed binary number by 2's-complementing it



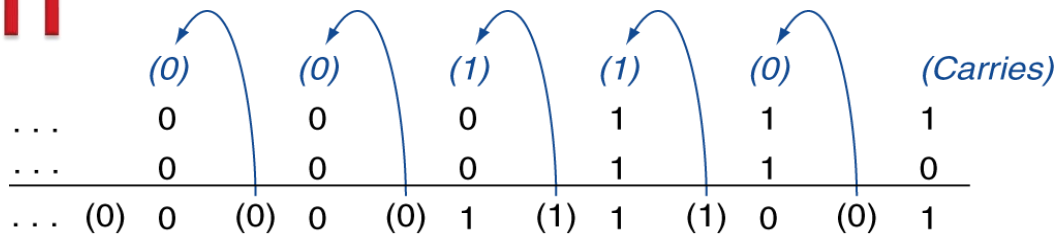


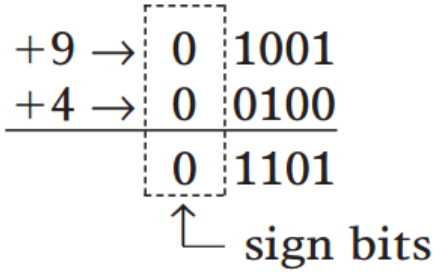
Signed numbers

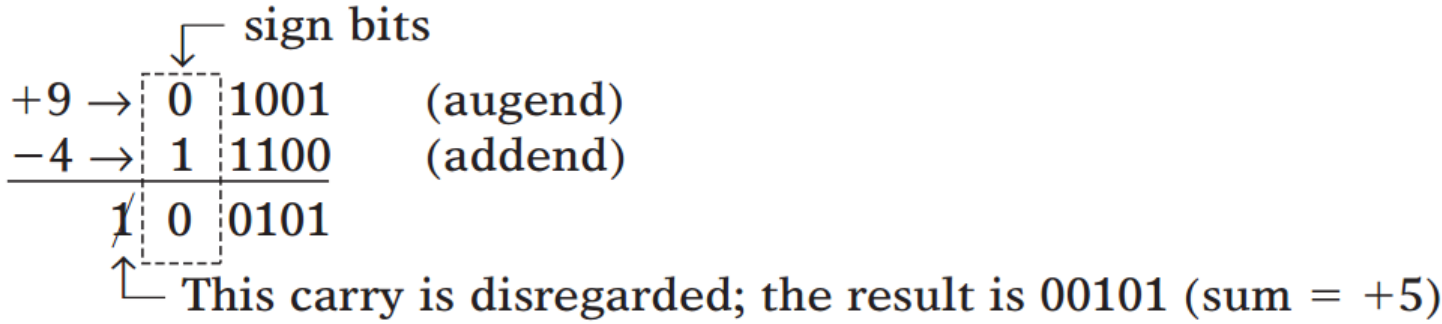
Decimal	Two's Complement	Ones' Complement	Signed Magnitude
−8	1000	—	—
−7	1001	1000	1111
−6	1010	1001	1110
−5	1011	1010	1101
−4	1100	1011	1100
−3	1101	1100	1011
−2	1110	1101	1010
−1	1111	1110	1001
0	0000	1111 or 0000	1000 or 0000
1	0001	0001	0001
2	0010	0010	0010
3	0011	0011	0011
4	0100	0100	0100
5	0101	0101	0101
6	0110	0110	0110
7	0111	0111	0111

Integer Addition

- Example: 7 + 6







- Overflow if result out of range
  - Adding +ve and -ve operands, no overflow
  - Adding two +ve operands
    - Overflow if result sign is 1
  - Adding two -ve operands
    - Overflow if result sign is 0

Integer Subtraction

- Negate the subtrahend and add this to the minuend
- Example: 9 – 4 = 9 + (–4)

minuend (+9) → 01001
subtrahend (+4) → 00100

```
  01001   (+9)
+ 11100   (−4)
-------
1 00101   (+5)
↑
└ Disregard, so the result is 00101 = +5.
```

- Overflow if result out of range
  - Subtracting two +ve or two –ve operands, no overflow
  - Subtracting +ve from –ve operand
    - Overflow if result sign is 0
  - Subtracting –ve from +ve operand
    - Overflow if result sign is 1

Subtraction using 2's complement

Example: Using 2's complement to do the subtraction 111001 - 1010

$$\begin{array}{rr} & 1\ 1\ 1\ 0\ 0\ 1 \\ + & \\ \text{Complement 2 of } 001010 \text{ is:} & 1\ 1\ 0\ 1\ 1\ 0 \\ \hline & 1\ 1\ 0\ 1\ 1\ 1\ 1 \end{array}$$

Ignore the last carry:

$\Rightarrow$111001 - 1010 = 101111

Subtraction using 2's complement

Example:

Using complement 2 to do subtraction 1010 - 111001

$$\begin{array}{rr} & 0\ 0\ 1\ 0\ 1\ 0 \\ + & \\ \text{Complement 2 of 111001 is:} & 0\ 0\ 0\ 1\ 1\ 1 \\ \hline & 0\ 1\ 0\ 0\ 0\ 1 \end{array}$$

There is no last carry. Find 2's complement of 010001 is 101111 $\Rightarrow$ 1010 - 111001 = -101111

Addition/Subtraction rules for binary numbers

Number System	Addition Rules	Negation Rules	Subtraction Rules
Unsigned	Add the numbers. Result is out of range if a carry out of the MSB occurs.	Not applicable	Subtract the subtrahend from the minuend. Result is out of range if a borrow out of the MSB occurs.
Signed magnitude	(same sign) Add the magnitudes; overflow occurs if a carry out of MSB occurs; result has the same sign. (opposite sign) Subtract the smaller magnitude from the larger; overflow is impossible; result has the sign of the larger.	Change the number's sign bit.	Change the sign bit of the subtrahend and proceed as in addition.
Two's complement	Add, ignoring any carry out of the MSB. Overflow occurs if the carries into and out of MSB are different.	Complement all bits of the number; add 1 to the result.	Complement all bits of the subtrahend and add to the minuend with an initial carry of 1.
Ones' complement	Add; if there is a carry out of the MSB, add 1 to the result. Overflow if carries into and out of MSB are different.	Complement all bits of the number.	Complement all bits of the subtrahend and proceed as in addition.

BCD Addition

```
  45        0100 0101   ← BCD for 45
+33       + 0011 0011   ← BCD for 33
  78        0111 1000   ← BCD for 78
```

- Sum Equals 9 or Less
  - none of the sums of the pairs of decimal digits exceeded 9; therefore, no decimal carries were produced.
  - the BCD addition process is straightforward and is actually the same as binary addition
- Sum Greater than 9
  - Sum is one of the six forbidden or invalid four-bit code groups because the sum of the two digits exceeds 9.
  - Whenever this occurs, the sum must be corrected by the addition of six (0110) to take into account the skipping of the six invalid code groups

```
  6        0110   ← BCD for 6
+7       + 0111   ← BCD for 7
+13      ~~1101~~   ← invalid code group for BCD
```

```
          0110   ← BCD for 6
        + 0111   ← BCD for 7
          1101   ← invalid sum
          0110   ← add 6 for correction
  0001    0011   ← BCD for 13
   1       3
```

BCD Addition procedure

- Using ordinary binary addition, add the BCD code groups for each digit position.
- For those positions where the sum is 9 or less, no correction is needed. The sum is in proper BCD form.
- When the sum of two digits is greater than 9, a correction of 0110 should be added to that sum to get the proper BCD result. This case always produces a carry into the next digit position, either from the original addition (step 1) or from the correction addition

275	0010	0111	0101	← BCD for 275
+641	+ 0110	0100	0001	← BCD for 641
916	1000	1011	0110	← perform addition
	+ 1	0110		← add 6 to correct second digit
	1001	0001	0110	← BCD for 916

Summary

- The hexadecimal number system is used in digital systems and computers as efficient ways of representing binary quantities
- In conversion between hex and binary, each hex digit corresponds to 4 bits
- The repeated division method is used to convert decimal numbers to binary or hexadecimal
- Using N bit binary number, we can represent decimal values from 0 to 2^N-1
- The BCD code for a decimal number is formed by converting each digit of the decimal number to its 4-bit binary equivalent
- An alphanumeric code is one that uses groups of bits to represent all of the various characters and functions that are part of a typical computer's keyboard. The ASCII code the most widely used alphanumeric code