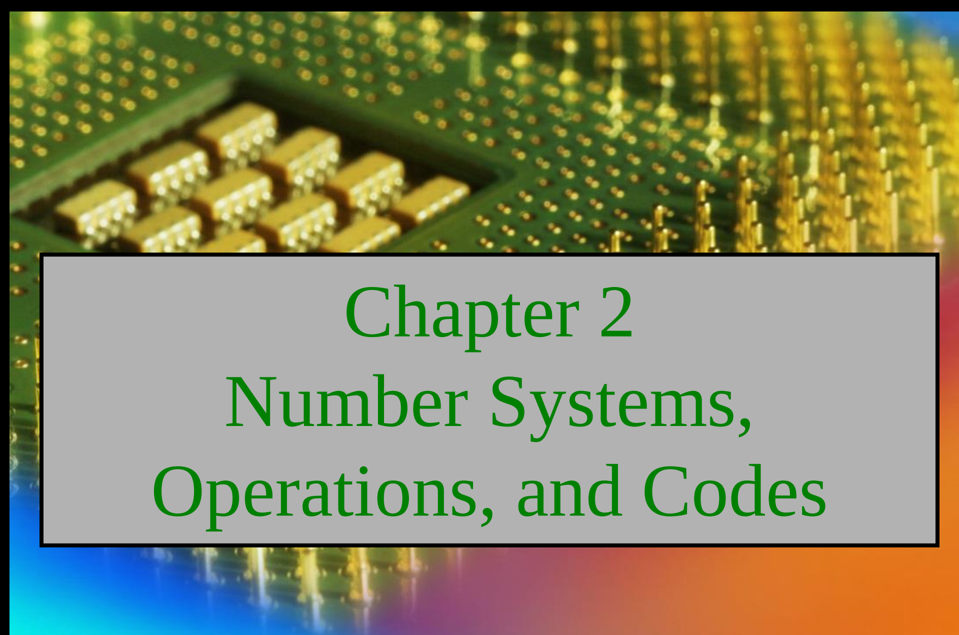
EE-227

Digital Logic Design

Spring-2019



Chapter 2
Number Systems, Operations, and Codes

Course Instructor: Engr. Khalid Iqbal Soomro

CHAPTER OBJECTIVES

- Review the decimal number system
- Count in the binary number system
- Convert from decimal to binary and from binary to decimal
- Apply arithmetic operations to binary numbers
- Determine the 1's and 2's complements of a binary number
- Express signed binary numbers in sign-magnitude, 1's complement, 2's complement, and floating-point format
- Carry out arithmetic operations with signed binary numbers
- Convert between the binary, and hexadecimal number systems
- Convert between the binary and octal number systems
- Add numbers in hexadecimal form
- Express decimal numbers in binary coded decimal (BCD) form

Decimal Numbers

The position of each digit in a weighted number system is assigned a weight based on the **base** or **radix** of the system. The radix of decimal numbers is 10, because only ten symbols (0 through 9) are used to represent any number.

The column weights of decimal numbers are powers of 10 that increase from right to left beginning with $10^0 = 1$:

$$\ldots 10^5\ 10^4\ 10^3\ 10^2\ 10^1\ 10^0.$$

For fractional decimal numbers, the column weights are negative powers of 10 that decrease from left to right:

$$10^2\ 10^1\ 10^0.\ 10^{-1}\ 10^{-2}\ 10^{-3}\ 10^{-4}\ \ldots$$

Decimal Numbers

Decimal numbers can be expressed as the sum of the products of each digit times the column value for that digit. Thus, the number 9240 can be expressed as

$$(9 \times 10^3) + (2 \times 10^2) + (4 \times 10^1) + (0 \times 10^0)$$

or

$$9 \times 1{,}000 + 2 \times 100 + 4 \times 10 + 0 \times 1$$

Example

Express the number 480.52 as the sum of values of each digit.

Solution

$$480.52 = (4 \times 10^2) + (8 \times 10^1) + (0 \times 10^0) + (5 \times 10^{-1}) + (2 \times 10^{-2})$$

Binary Numbers

For digital systems, the binary number system is used. Binary has a radix of 2 and uses the digits 0 and 1 to represent quantities.

The column weights of binary numbers are powers of 2 that increase from right to left beginning with $2^0 = 1$:

$\ldots 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0.$

For fractional binary numbers, the column weights are negative powers of 2 that decrease from left to right:

$2^2\ 2^1\ 2^0.\ 2^{-1}\ 2^{-2}\ 2^{-3}\ 2^{-4}\ \ldots$

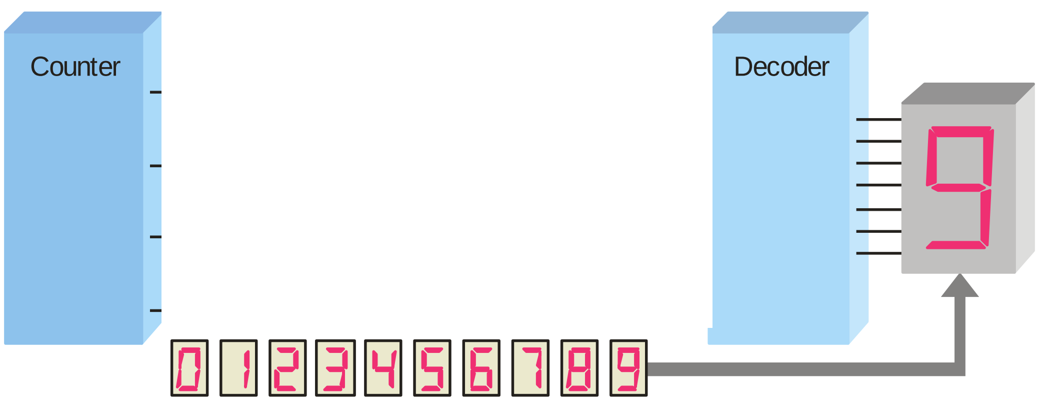
Microsoft PowerPoint Presentation
Microsoft PowerPoint Presentation

Binary Numbers

A binary counting sequence for numbers from zero to fifteen is shown.

Notice the pattern of zeros and ones in each column.

Digital counters frequently have this same pattern of digits:



Decimal Number	Binary Number
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1
10	1 0 1 0
11	1 0 1 1
12	1 1 0 0
13	1 1 0 1
14	1 1 1 0
15	1 1 1 1

Binary Conversions

The decimal equivalent of a binary number can be determined by adding the column values of all of the bits that are 1 and discarding all of the bits that are 0.

Example

Convert the binary number 100101.01 to decimal.

Solution

Start by writing the column weights; then add the weights that correspond to each 1 in the number.

2^5	2^4	2^3	2^2	2^1	2^0.	2^{-1}	2^{-2}	
32	16	8	4	2	1 .	½	¼	
1	0	0	1	0	1.	0	1	
32			+4		+1		+¼	= 37.25

Binary Conversions

You can convert a decimal whole number to binary by reversing the procedure. Write the decimal weight of each column and place 1's in the columns that sum to the decimal number.

Example Convert the decimal number 49 to binary.

Solution The column weights double in each position to the right. Write down column weights until the last number is larger than the one you want to convert.

2^6	2^5	2^4	2^3	2^2	2^1	2^0.
64	32	16	8	4	2	1.
0	1	1	0	0	0	1.

Binary Conversions

You can convert a decimal fraction to binary by repeatedly multiplying the fractional results of successive multiplications by 2. The carries form the binary number.

Example

Convert the decimal fraction 0.188 to binary by repeatedly multiplying the **fractional results** by 2.

Solution

0.188 x 2 = 0.376	carry = 0	MSB
0.376 x 2 = 0.752	carry = 0	
0.752 x 2 = 1.504	carry = 1	
0.504 x 2 = 1.008	carry = 1	
0.008 x 2 = 0.016	carry = 0	

Answer = .00110 (upto five significant digits)

Binary Conversions

You can convert decimal to any other base by repeatedly dividing by the base. For binary, repeatedly divide by 2:

Example Convert the decimal number 49 to binary by repeatedly dividing by 2.

Solution You can do this by “reverse division” and the answer will read from left to right. Put quotients to the left and remainders on top.

Answer: 1 1 0 0 0 1 ← remainder

0 1 3 6 12 24 49 ⟌ 2

Continue until the last quotient is 0

Quotient

Decimal number

base

Binary Addition

The rules for binary addition are

0 + 0 = 0	Sum = 0, carry = 0
0 + 1 = 1	Sum = 1, carry = 0
1 + 0 = 1	Sum = 1, carry = 0
1 + 1 = 10	Sum = 0, carry = 1

When an input carry = 1 due to a previous result, the rules are

1 + 0 + 0 = 1	Sum = 1, carry = 0
1 + 0 + 1 = 10	Sum = 0, carry = 1
1 + 1 + 0 = 10	Sum = 0, carry = 1
1 + 1 + 1 = 11	Sum = 1, carry = 1

Binary Addition

Example Add the binary numbers 00111 and 10101 and show the equivalent decimal addition.

Solution

$$\begin{array}{rcr} {\scriptstyle 0\,1\,1\,1} & & \\ 00111 & & 7 \\ 10101 & & 21 \\ \hline 11100 & = & 28 \end{array}$$

Binary Subtraction

The rules for binary subtraction are

0 – 0 = 0
1 – 1 = 0
1 – 0 = 1
10 – 1 = 1 with a borrow of 1

Example Subtract the binary number 011 from 101 and show the equivalent decimal subtraction.

Solution

$$\begin{array}{lr} \;1 & \\ \;1 & \\ \cancel{1}01 & 5 \\ 011 & -3 \\ \hline 010 & 2 \end{array}$$

Binary Multiplication

The rules for binary multiplication are

$$0 \times 0 = 0$$
$$0 \times 1 = 0$$
$$1 \times 0 = 0$$
$$1 \times 1 = 1$$

Example Multiply the binary number 11 with 11.

Solution

Binary	Decimal
11	3
x 11	x 3
11	9
+11	
1001	9

Microsoft
werPoint Presentati

Binary Division

Division in binary follows the same procedure as division in decimal.

Example Perform the following binary division: 110 / 11

Solution

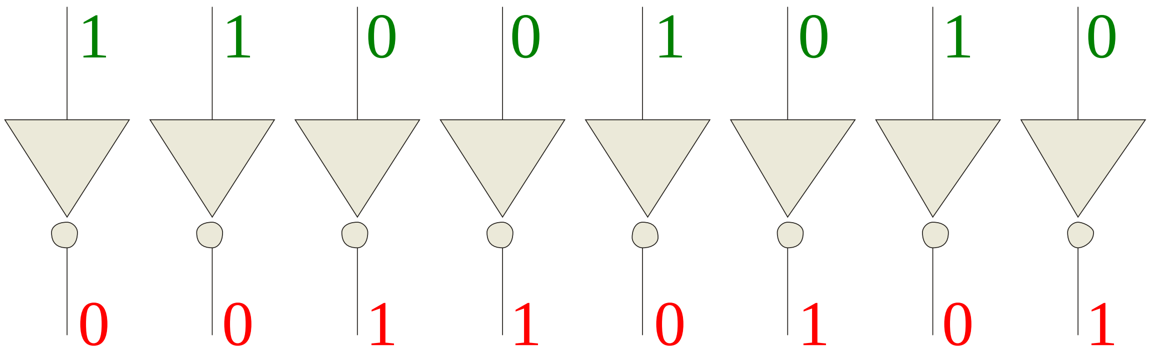
Binary	Decimal
10	2
11 /110	3 /6
110	6
000	0

1's Complement

The 1's complement of a binary number is just the inverse of the digits. To form the 1's complement, change all 0's to 1's and all 1's to 0's.

For example, the 1's complement of 11001010 is
00110101

In digital circuits, the 1's complement is formed by using inverters:



2’s Complement

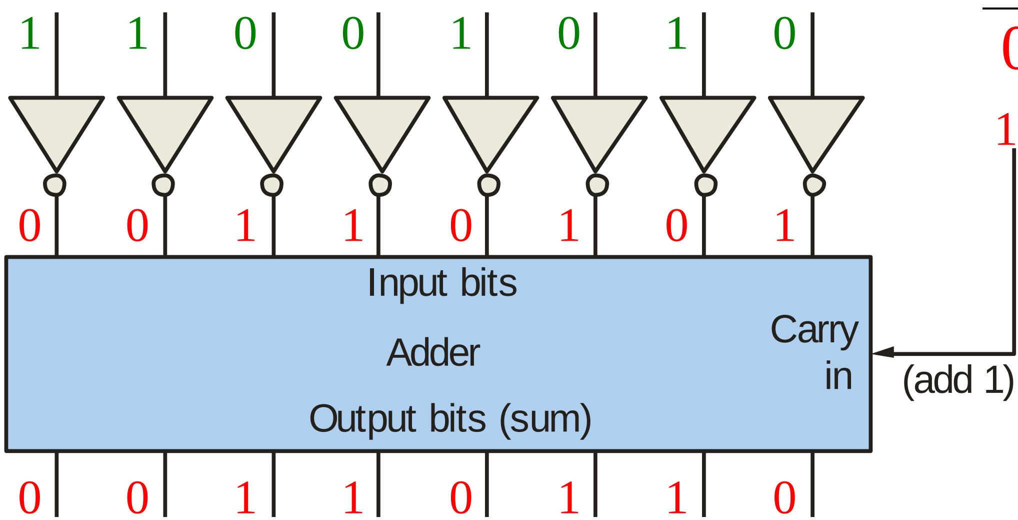
The 2’s complement of a binary number is found by adding 1 to the LSB of the 1’s complement.

Recall that the 1’s complement of 11001010 is

To form the 2’s complement, add 1:

00110101 (1’s complement)
+1

00110110 (2’s complement)



Signed Binary Numbers

There are several ways to represent signed binary numbers. In all cases, the MSB in a signed number is the sign bit, that tells you if the number is positive or negative.

Computers use a modified 2's complement for signed numbers. Positive numbers are stored in *true* form (with a 0 for the sign bit) and negative numbers are stored in ***complement*** form (with a 1 for the sign bit).

For example, the positive number 58 is written using 8-bits as 00111010 (true form).

Sign bit

Magnitude bits

Signed Binary Numbers

Negative numbers are written as the 2's complement of the corresponding positive number.

The negative number –58 is written as:

–58 = 11000110 (complement form)

Sign bit Magnitude bits

An easy way to read a signed number that uses this notation is to assign the sign bit a column weight of –128 (for an 8-bit number). Then add the column weights for the 1's.

Example Assuming that the sign bit = –128, show that 11000110 = –58 as a 2's complement signed number:

Solution Column weights: –128 64 32 16 8 4 2 1.

–128	64	32	16	8	4	2	1	
1	1	0	0	0	1	1	0	
–128	+64				+4	+2		= –58

Floating Point Numbers

Microsoft
werPoint Presentati

Floating point notation is capable of representing very large or small numbers by using a form of scientific notation. A 32-bit single precision number is illustrated.

S	E (8 bits)	F (23 bits)

IEEE 754 standard

Sign bit

Biased exponent (+127)

Magnitude with MSB dropped

Example

Express the speed of light, *c*, in single precision floating point notation. c =299,800,000 or (0.2998×10^9)

Solution

In binary, $c = 0001\ 0001\ 1101\ 1110\ 1001\ 0101\ 1100\ 0000_2$.

Dropped

In scientific notation, $c = 1.001\ 1101\ 1110\ 1001\ 0101\ 1100\ 0000 \times 2^{28}$.

S = 0 because the number is positive. E = 28 + 127 = $155_{10} = 1001\ 1011_2$.

F is the next 23 bits after the first 1 is dropped.

In floating point notation, c =

0	10011011	001 1101 1110 1001 0101 1100

Arithmetic Operations with Signed Numbers

Using the signed number notation with negative numbers in 2's complement form simplifies addition and subtraction of signed numbers.

Rules for **addition**: Add the two signed numbers. Discard any final carries. The result is in signed form.

Examples:

00011110 = +30	00001110 = +14	11111111 = –1
00001111 = +15	11101111 = –17	11111000 = –8
00101101 = +45	11111101 = –3	~~1~~ 11110111 = –9

Discard carry

Arithmetic Operations with Signed Numbers

Note that if the number of bits required for the answer is exceeded, overflow will occur. This occurs only if both numbers have the same sign. The overflow will be indicated by an incorrect sign bit.

Two examples are:

01000000 = +128	10000001 = –127
01000001 = +129	10000001 = –127
10000001 = ~~–126~~	Discard carry → ~~1~~00000010 = ~~+2~~

Wrong! The answer is incorrect and the sign bit has changed.

Arithmetic Operations with Signed Numbers

Rules for **subtraction**: 2's complement the subtrahend and add the numbers. Discard any final carries. The result is in signed form.

Repeat the examples done previously, but subtract:

$$\begin{array}{rl} 00011110 & (+30) \\ -\ 00001111 & -(+15) \\ \hline \end{array} \qquad \begin{array}{rl} 00001110 & (+14) \\ -\ 11101111 & -(-17) \\ \hline \end{array} \qquad \begin{array}{rl} 11111111 & (-1) \\ -\ 11111000 & -(-8) \\ \hline \end{array}$$

2's complement subtrahend and add:

$$\begin{array}{rl} 00011110 & = +30 \\ 11110001 & = -15 \\ \hline \cancel{1}00001111 & = +15 \end{array} \qquad \begin{array}{rl} 00001110 & = +14 \\ 00010001 & = +17 \\ \hline 00011111 & = +31 \end{array} \qquad \begin{array}{rl} 11111111 & = -1 \\ 00001000 & = +8 \\ \hline \cancel{1}00000111 & = +7 \end{array}$$

Discard carry

Discard carry

Hexadecimal Numbers

Hexadecimal uses sixteen characters to represent numbers: the numbers 0 through 9 and the alphabetic characters A through F.

Large binary number can easily be converted to hexadecimal by grouping bits 4 at a time and writing the equivalent hexadecimal character.

Example Express $1001\ 0110\ 0000\ 1110_2$ in hexadecimal:

Solution Group the binary number by 4-bits starting from the right. Thus, 960E

Decimal	Hexadecimal	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
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Hexadecimal Numbers

Hexadecimal is a weighted number system. The column weights are powers of 16, which increase from right to left.

Column weights $\begin{cases} 16^3 & 16^2 & 16^1 & 16^0. \\ 4096 & 256 & 16 & 1\ . \end{cases}$

Example Express $1A2F_{16}$ in decimal.

Solution Start by writing the column weights:

4096 256 16 1

1 A 2 F_{16}

$1(4096) + 10(256) + 2(16) + 15(1) = 6703_{10}$

Decimal	Hexadecimal	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
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Octal Numbers

Octal uses eight characters the numbers 0 through 7 to represent numbers. There is no 8 or 9 character in octal.

Binary number can easily be converted to octal by grouping bits 3 at a time and writing the equivalent octal character for each group.

Example Express 1 001 011 000 001 110_2 in octal:

Solution Group the binary number by 3-bits starting from the right. Thus, 113016_8

Decimal	Octal	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	10	1000
9	11	1001
10	12	1010
11	13	1011
12	14	1100
13	15	1101
14	16	1110
15	17	1111

Octal Numbers

Octal is also a weighted number system. The column weights are powers of 8, which increase from right to left.

Column weights $\left\{ \begin{array}{cccc} 8^3 & 8^2 & 8^1 & 8^0 \\ 512 & 64 & 8 & 1 \end{array} \right.$.

Example Express 3702_8 in decimal.

Solution Start by writing the column weights:

512 64 8 1

3 7 0 2_8

$3(512) + 7(64) + 0(8) + 2(1) = 1986_{10}$

Decimal	Octal	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	10	1000
9	11	1001
10	12	1010
11	13	1011
12	14	1100
13	15	1101
14	16	1110
15	17	1111

BCD

Binary coded decimal (BCD) is a way to express each of the decimal digits with a binary code.

There are only ten code groups in the BCD system, so it is very easy to convert between decimal and BCD.

Because we like to read and write in decimal, the BCD code provides an excellent interface to binary systems.

Examples of such interfaces are keypad inputs and digital readouts.

The 8421 BCD Code

The 8421 code is a type of BCD (binary coded decimal) code.

Binary coded decimal means that each decimal digit, 0 through 9, is represented by a binary code of four bits.

The designation 8421 indicates the binary weights of the four bits (2^3 , 2^2 ,2^1 , 2^0).

The ease of conversion between 8421 code numbers and the familiar decimal numbers is the main advantage of this code.

All we have to remember are the ten binary combinations that represent the ten decimal digits

BCD

Binary coded decimal (BCD) is a weighted code that is commonly used in digital systems when it is necessary to show decimal numbers such as in clock displays.

The table illustrates the difference between straight binary and BCD. BCD represents each decimal digit with a 4-bit code. Notice that the codes 1010 through 1111 are not used in BCD.

Decimal	Binary	BCD
0	0000	0000
1	0001	0001
2	0010	0010
3	0011	0011
4	0100	0100
5	0101	0101
6	0110	0110
7	0111	0111
8	1000	1000
9	1001	1001
10	1010	00010000
11	1011	00010001
12	1100	00010010
13	1101	00010011
14	1110	00010100
15	1111	00010101

BCD

To express any decimal number in BCD, simply replace each decimal digit with the appropriate 4-bit code, as shown by Example below:

Convert each of the following decimal numbers to BCD:
(a) 35 (b) 98 (c) 170 (d) 2469
Solution

(a)	3	5		(b)	9	8			
	0011	0101			1001	1000			
(c)	1	7	0	(d)	2	4	6	9	
	0001	0111	0000		0010	0100	0110	1001	

BCD

You can think of BCD in terms of column weights in groups of four bits. For an 8-bit BCD number, the column weights are: 80 40 20 10 8 4 2 1.

Question: What are the column weights for the BCD number 1000 0011 0101 1001?

Answer:

8000 4000 2000 1000 800 400 200 100 80 40 20 10 8 4 2 1

Note that you could add the column weights where there is a 1 to obtain the decimal number. For this case:

$$8000 + 200 + 100 + 40 + 10 + 8 + 1 = 8359_{10}$$

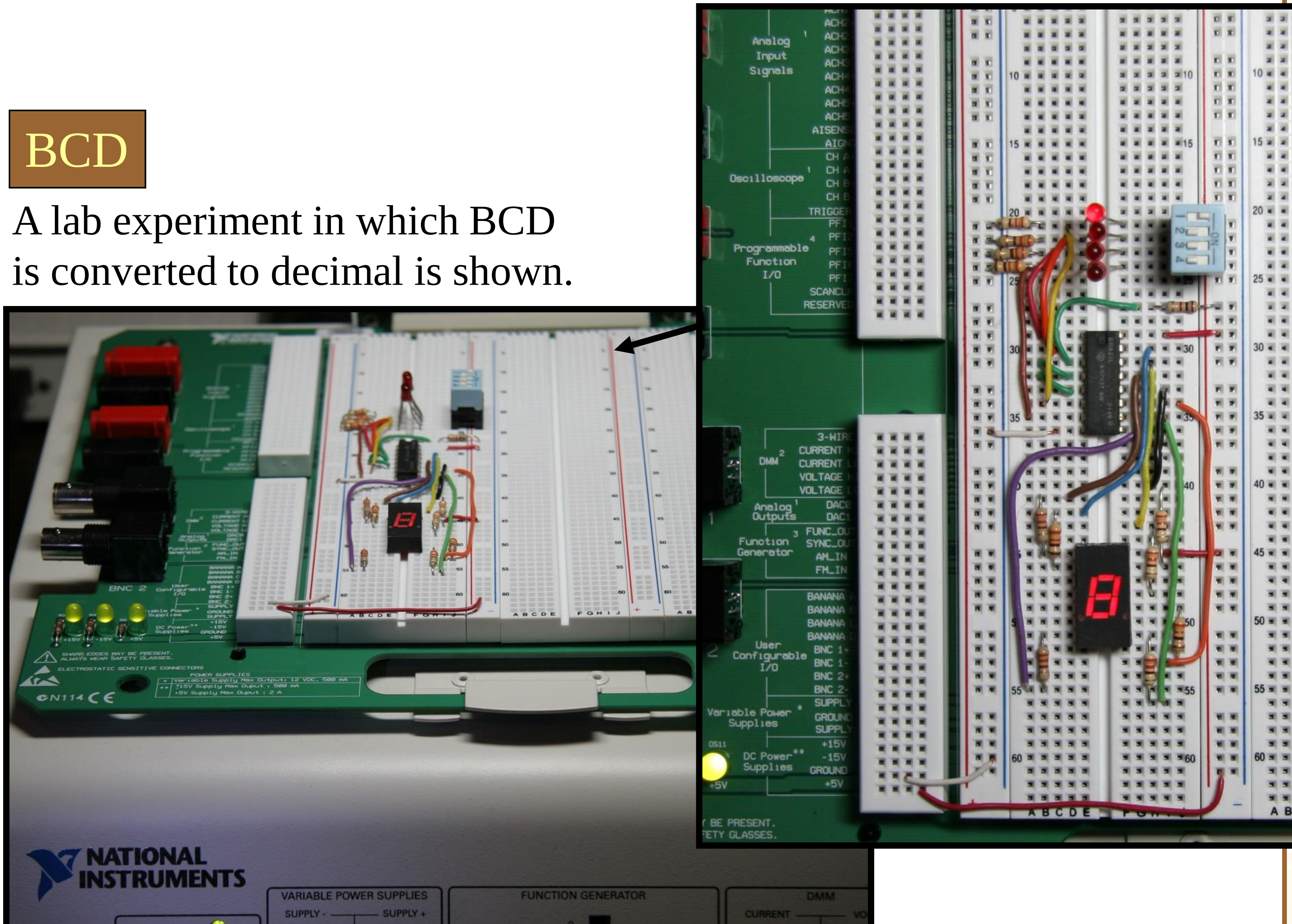
Applications

Digital clocks, digital thermometers, digital meters, and other devices with seven-segment displays typically use BCD code to simplify the displaying of decimal numbers.

BCD is not as efficient as straight binary for calculations, but it is particularly useful if only limited processing is required, such as in a digital thermometer.

BCD

A lab experiment in which BCD is converted to decimal is shown.



Gray code

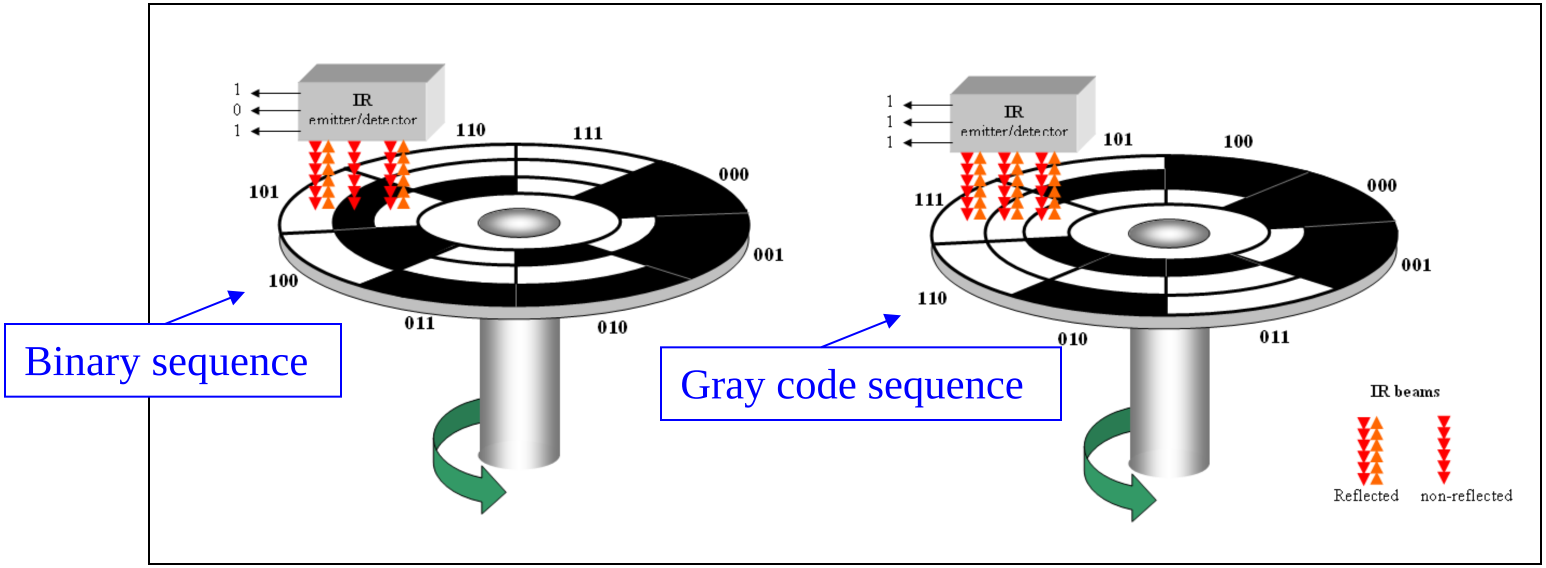
Gray code is an unweighted code that has a single bit change between one code word and the next in a sequence.

Gray code is used to avoid problems in systems where an error can occur if more than one bit changes at a time.

Decimal	Binary	Gray code
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

Gray code

A shaft encoder is a typical application. Three IR emitter/detectors are used to encode the position of the shaft. The encoder on the left uses binary and can have three bits change together, creating a potential error. The encoder on the right uses gray code and only 1-bit changes, eliminating potential errors.



Binary-to-Gray Code Conversion

Conversion between binary code and Gray code is sometimes useful. The following rules explain how to convert from a binary number to a Gray code word:

1. The most significant bit (left-most) in the Gray code is the same as the corresponding MSB in the binary number.

2. Going from left to right, add each adjacent pair of binary code bits to get the next Gray code bit. Discard carries.

For example, the conversion of the binary number 10110 to Gray code is as follows:

$$\begin{array}{ccccccccccccccccc} 1 & - & + & \rightarrow & 0 & - & + & \rightarrow & 1 & - & + & \rightarrow & 1 & - & + & \rightarrow & 0 \\ \downarrow & & & & \downarrow & & & & \downarrow & & & & \downarrow & & & & \downarrow \\ 1 & & & & 1 & & & & 1 & & & & 0 & & & & 1 \end{array} \quad \begin{array}{l} \text{Binary} \\ \\ \text{Gray} \end{array}$$

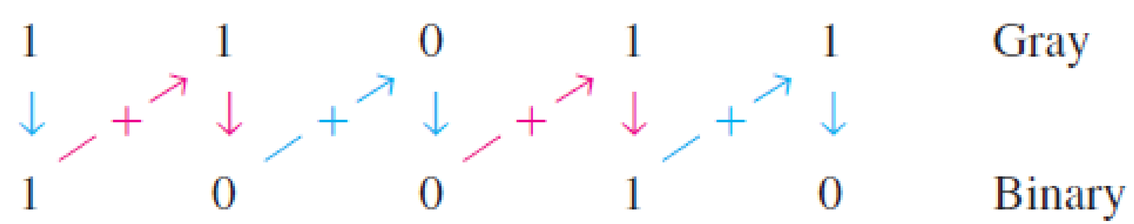
The Gray code is 11101.

Gray-to-Binary Code Conversion

To convert from Gray code to binary, use a similar method; however, there are some differences. The following rules apply:

1. The most significant bit (left-most) in the binary code is the same as the corresponding bit in the Gray code.
2. Add each binary code bit generated to the Gray code bit in the next adjacent position. Discard carries.

For example, the conversion of the Gray code word 11011 to binary is as follows:



The binary number is 10010.

EXAMPLE 2–37

(a) Convert the binary number 11000110 to Gray code.

(b) Convert the Gray code 10101111 to binary.

Solution

(a) Binary to Gray code:

$$1 - + \rightarrow 1 - + \rightarrow 0 - + \rightarrow 0 - + \rightarrow 0 - + \rightarrow 1 - + \rightarrow 1 - + \rightarrow 0$$

$$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow$$

$$\mathbf{1} \quad \mathbf{0} \quad \mathbf{1} \quad \mathbf{0} \quad \mathbf{0} \quad \mathbf{1} \quad \mathbf{0} \quad \mathbf{1}$$

(b) Gray code to binary:

$$1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1$$

$$\downarrow \; + \nearrow \downarrow \; + \nearrow \downarrow \; + \nearrow \downarrow \; + \nearrow \downarrow \; + \nearrow \downarrow \; + \nearrow \downarrow \; + \nearrow \downarrow$$

$$\mathbf{1} \quad \mathbf{1} \quad \mathbf{0} \quad \mathbf{0} \quad \mathbf{1} \quad \mathbf{0} \quad \mathbf{1} \quad \mathbf{0}$$

ASCII

ASCII is a code for alphanumeric characters and control characters. In its original form, ASCII encoded 128 characters and symbols using 7-bits. The first 32 characters are control characters, that are based on obsolete teletype requirements, so these characters are generally assigned to other functions in modern usage.

In 1981, IBM introduced extended ASCII, which is an 8-bit code and increased the character set to 256. Other extended sets (such as Unicode) have been introduced to handle characters in languages other than English.

Parity Method

The parity method is a method of error detection for simple transmission errors involving one bit (or an odd number of bits). A parity bit is an "extra" bit attached to a group of bits to force the number of 1's to be either even (even parity) or odd (odd parity).

Example The ASCII character for "a" is 1100001 and for "A" is 1000001. What is the correct bit to append to make both of these have odd parity?

Solution The ASCII "a" has an odd number of bits that are equal to 1; therefore the parity bit is 0. The ASCII "A" has an even number of bits that are equal to 1; therefore the parity bit is 1.

Cyclic Redundancy Check

The cyclic redundancy check (CRC) is an error detection method that can detect multiple errors in larger blocks of data.

At the sending end, a checksum is appended to a block of data.

At the receiving end, the check sum is generated and compared to the sent checksum.

If the check sums are the same, no error is detected.

Selected Key Terms

Byte	A group of eight bits
Floating-point number	A number representation based on scientific notation in which the number consists of an exponent and a mantissa.
Hexadecimal	A number system with a base of 16.
Octal	A number system with a base of 8.
BCD	Binary coded decimal; a digital code in which each of the decimal digits, 0 through 9, is represented by a group of four bits.

Selected Key Terms

Alphanumeric	Consisting of numerals, letters, and other characters
ASCII	American Standard Code for Information Interchange; the most widely used alphanumeric code.
Parity	In relation to binary codes, the condition of evenness or oddness in the number of 1s in a code group.
Cyclic redundancy check (CRC)	A type of error detection code.

Quiz

1. For the binary number 1000, the weight of the column with the 1 is

    a. 4

    b. 6

    c. 8

    d. 10

Quiz

2. The 2’s complement of 1000 is

   a. 0111

   b. 1000

   c. 1001

   d. 1010

Quiz

3. The fractional binary number 0.11 has a decimal value of

   a. ¼

   b. ½

   c. ¾

   d. none of the above

Quiz

4. The hexadecimal number 2C has a decimal equivalent value of

a. 14

b. 44

c. 64

d. none of the above

Quiz

5. Assume that a floating point number is represented in binary. If the sign bit is 1, the

a. number is negative

b. number is positive

c. exponent is negative

d. exponent is positive

Quiz

6. When two positive signed numbers are added, the result may be larger that the size of the original numbers, creating overflow. This condition is indicated by

a. a change in the sign bit

b. a carry out of the sign position

c. a zero result

d. smoke

Quiz

7. The number 1010 in BCD is

   a. equal to decimal eight

   b. equal to decimal ten

   c. equal to decimal twelve

   d. invalid

Quiz

8. An example of an unweighted code is

   a. binary

   b. decimal

   c. BCD

   d. Gray code

Quiz

9. An example of an alphanumeric code is

   a. hexadecimal

   b. ASCII

   c. BCD

   d. CRC

Quiz

10. An example of an error detection method for transmitted data is the

a. parity check

b. CRC

c. both of the above

d. none of the above

Quiz

Answers:

1. c	6. a
2. b	7. d
3. c	8. d
4. b	9. b
5. a	10. c