PDHonline Course E377 (5 PDH)

Digital Logic Systems Volume I - Digital Number Systems

Instructor: Lee Layton, PE

2020

PDH Online | PDH Center

5272 Meadow Estates Drive
Fairfax, VA 22030-6658
Phone: 703-988-0088
www.PDHonline.com

An Approved Continuing Education Provider

Digital Logic Circuits
Volume I
Digital Number Systems

Lee Layton, P.E

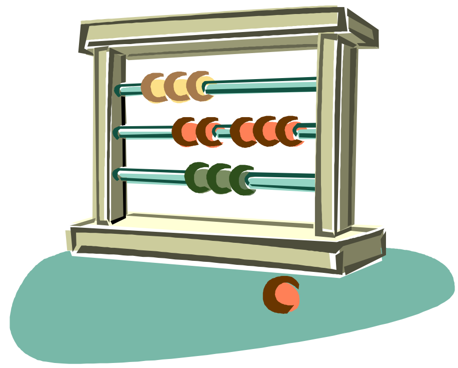
Table of Contents

This series of courses are based on the Navy Electricity and Electronics Training Series (NEETS) section on Logic systems. The NEETS material has been reformatted for readability and ease of use as a continuing education course. The NEETS series is produced by the Naval Education and Training Professional Development and Technology Center.

Introduction

Ever since people discovered that it was necessary to count objects, they have been looking for easier ways to count them. The abacus, developed by the Chinese, is one of the earliest known calculators. It is still in use in some parts of the world.

A Frenchman, Blaise Pascal, invented the first adding machine in 1642. Twenty years later, an Englishman, Sir Samuel Moreland, developed a more compact device that could multiply, add, and subtract. In Germany, around 1672, Gottfried Wilhelm von Leibniz perfected a machine that could perform all the basic operations (add, subtract, multiply, divide), as well as extract the square root. Modern electronic digital computers still use von Liebniz's principles.



Computers are now employed wherever repeated calculations or the processing of huge amounts of data is needed. The greatest applications are found in the military, scientific, and commercial fields. They have applications that range from mail sorting, through engineering design, to the identification and destruction of enemy targets. The advantages of digital computers include speed, accuracy, and manpower savings. Often computers are able to take over routine jobs and release personnel for more important work - work that cannot be handled by a computer.

People and computers do not normally speak the same language. Methods of translating information into forms that are understandable and usable to both are necessary. Humans generally speak in words and numbers expressed in the decimal number system, while computers only understand coded electronic pulses that represent digital information.

In this course you will learn about number systems in general and about binary, octal, and hexadecimal number systems specifically. Methods for converting numbers in the binary, octal, and hex systems to equivalent numbers in the decimal system (and vice versa) will also be described. You will see that these number systems can be easily converted to the electronic signals necessary for digital equipment.

This course is the first in a series of courses and lays out the basics needed to comprehend digital logic circuits. Subsequent courses address fundamental logic circuits and special logic circuits.

Chapter 1
Types of Number Systems

Most people only use one number system, the decimal system and some are familiar with the Roman numeral system, even though they rarely use it, except maybe to translate how many Super Bowls have occurred! In this chapter we will look at the different types of number systems.

The Decimal Number System

We all know and understand the decimal number system and therefore this course uses the decimal number system to explain other bases. The examples may seem simplistic, but they will help to understand other number systems. You should realize that these systems have certain things in common. These common terms will be defined using the decimal system as our base. Each term will be related to each number system as that number system is introduced.

Each of the number systems you will study is built around the following components: the *unit*, *number*, and *base* (radix).

Unit and Number

The terms *unit* and *number* when used with the decimal system are almost self-explanatory. By definition the unit is a single object; that is, an apple, a dollar, a day. A number is a symbol representing a unit or a quantity. The figures 0, 1, 2, and 3 through 9 are the symbols used in the decimal system. These symbols are called *Arabic numerals* or figures. Other symbols may be used for different number systems.

For example, the symbols used with the Roman numeral system are letters, for instance, “V” is the symbol for 5, “X” for 10, “M” for 1,000, and so forth. We will use Arabic numerals and letters in the number system discussions in this chapter.

Base (Radix)

The *base*, or *radix*, of a number system tells you the number of symbols used in that system. The base of any system is always expressed in decimal numbers. The base, or radix, of the decimal system is 10. This means there are 10 symbols - 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 - used in the system. A number system using three symbols - 0, 1, and 2 - would be base 3; four symbols would be base 4; and so forth. Remember to count the zero or the symbol used for zero when determining the number of symbols used in a number system.

The base of a number system is indicated by a subscript (decimal number) following the value of the number. The following are examples of numerical values in different bases with the subscript to indicate the base:

$$7592_{10} \quad 214_5 \quad 123_4 \quad 656_7$$

You should notice the highest value symbol used in a number system is always one less than the base of the system. In base-10 the largest value symbol possible is 9; in base-5 it is 4; in base-3 it is 2.

Positional Notation and Zero

You must observe two principles when counting or writing quantities or numerical values. They are the *positional notation* and the *zero* principles.

Positional notation is a system where the value of a number is defined not only by the symbol but by the symbol's position. Let's examine the decimal (base-10) value of 427.5. You know from experience that this value is four hundred twenty-seven and one-half. Now examine the position of each number:

4 2 7 . 5

This represents ½ unit
This represents 7 units
This represents 20 units
This represents 400 units

If 427.5 is the quantity you wish to express, then each number must be in the position shown. If you exchange the positions of the 2 and the 7, then you change the value.

Each position in the positional notation system represents a power of the base, or radix. A *power* is the number of times a base is multiplied by itself. The power is written above and to the right of the base and is called an *exponent*. Examine the following base-10 line graph,

$$\mathbf{10^3 \quad 10^2 \quad 10^1 \quad 10^0 \quad . \quad 10^{-1} \quad 10^{-2} \quad 10^{-3}}$$

Where,

10^3 = 10 *100 = 1,000

10^2 = 10*10 = 100

10^1 = 10 * 1 = 10

10^0 = 1

10^{-1} = 1/10 = 0.1

10^{-2} = 1/100 = 0.01

10^{-3} = 1/1000 = 0.001

Now let's look at the value of the base-10 number 427.5 with the positional notation line graph in the chart below,

10^2	**10^1**	**10^0**	**.**	**10^{-1}**
4	2	7	.	5
4*100 = 400	2*10 = 20	7*1 = 7	.	5*0.1 = 0.5
427.5				

You can see that the power of the base is multiplied by the number in that position to determine the value for that position.

All numbers to the left of the decimal point are whole numbers, and all numbers to the right of the decimal point are fractional numbers. A whole number is a symbol that represents one, or more, complete objects, such as one apple or $5. A fractional number is a symbol that represents a portion of an object, such as half of an apple (.5 apples) or a quarter of a dollar ($0.25). A mixed number represents one, or more, complete objects, and some portion of an object, such as one and a half apples (1.5 apples). When you use any base other than the decimal system, the division between whole numbers and fractional numbers is referred to as the *radix point*. The decimal point is actually the radix point of the decimal system, but the term radix point is normally not used with the base-10 number system. Remember, in a base-10 system,

Decimal Point = Radix Point

Just as important as positional notation is the use of the zero. The placement of the zero in a number can have quite an effect on the value being represented. Sometimes a position in a number does not have a value between 1 and 9. Consider how this would affect your next paycheck. If you were expecting a check for $605.47, you wouldn't want it to be $65.47.

Leaving out the zero in this case means a difference of $540.00. In the number 605.47, the zero indicates that there are no tens. If you place this value on a bar graph, you will see that there are no multiples of 10^1.

10^2	10^1	10^0	.	10^{-1}	10^{-2}
6	0	5	.	4	7

Most Significant Digit and Least Significant Digit (MSD and LSD)

Other important factors of number systems that you should recognize are the *most significant digit (MSD)* and the *least significant digit (LSD).*

The MSD in a number is the digit that has the *greatest* effect on that number. The LSD in a number is the digit that has the *least* effect on that number. Look at the following examples:

187**6**.0
473.02**2**
0.0**3**26**9**

In this example, the MSD is shown in red and the LSD is shown in green. You can easily see that a change in the MSD will increase or decrease the value of the number the greatest amount. Changes in the LSD will have the smallest effect on the value.

The nonzero digit of a number that is the farthest *left* is the MSD, and the nonzero digit farthest RIGHT is the LSD, as in the following example:

0 0 **4** 0 1 . 0 0 **2** 0 0

In a *whole number* the LSD can be zero and will always be the digit immediately to the left of the radix point.

5 7 9 3 **0** . 0

Carry and Borrow Principles

Soon after you learned how to count, you were taught how to add and subtract. At that time, you learned some concepts that you use almost every day. Those concepts will be reviewed using the decimal system. They will also be applied to the other number systems.

Addition is a form of counting in which one quantity is added to another. The following definitions identify the basic terms of addition:

- *Augend* - the quantity to which an addend is added
- *Addend* - a number to be added to a preceding number
- *Sum* - the result of an addition (the sum of 5 and 7 is 12)
- *Carry*- a carry is produced when the sum of two or more digits in a vertical column equals or exceeds the base of the number system in use

How do we handle the carry; that is, the two-digit number generated when a carry is produced? The lower order digit becomes the sum of the column being added; the higher order digit (the carry) is added to the next higher order column. For example, let's add 15 and 7 in the decimal system:

1	Carry
15	Augend
+7	Addend
22	Sum

Starting with the first column, we find the sum of 5 and 7 is 12. The 2 becomes the sum of the lower order column and the 1 (the carry) is added to the upper order column. The sum of the upper order column is 2. The sum of 15 and 7 is, therefore, 22.

The rules for addition are basically the same regardless of the number system being used. Each number system, because it has a different number of digits, will have a unique digit addition table. These addition tables will be described during the discussion of the adding process for each number system.

A decimal addition table is shown in Table 1. The numbers in row X and column Y may represent either the addend or the augend. If the numbers in X represent the augend, then the numbers in Y must represent the addend and vice versa. The sum of X + Y is located at the point in array Z where the selected X row and Y column intersect.

Table 1
Decimal Addition Table

		Y									
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
X	**0**	0	1	2	3	4	5	6	7	8	9
	1	1	2	3	4	5	6	7	8	9	10
	2	2	3	4	5	6	7	8	9	10	11
	3	3	4	5	6	7	8	9	10	11	12
	4	4	5	6	7	8	9	10	11	12	13
	5	5	6	7	8	9	10	11	12	13	14
	6	6	7	8	9	10	11	12	13	14	15
	7	7	8	9	10	11	12	13	14	15	16
	8	8	9	10	11	12	13	14	15	16	17
	9	9	10	11	12	13	14	15	16	17	18

To add 5 and 7 using the table, first locate one number in the X row and the other in the Y column. The point in field where the row and column intersect is the sum. In this case the sum is 12.

Subtraction

The following definitions identify the basic terms you will need to know to understand subtraction operations:

To *subtract* is to take away, as a part from the whole or one number from another. The terms of subtraction are:

- *Minuend* - The number from which another number is to be subtracted
- *Subtrahend* - The quantity to be subtracted
- *Remainder*, or *Difference* - That which is left after subtraction
- *Borrow* - To transfer a digit (equal to the base number) from the next higher order column for the purpose of subtraction.

Use the rules of subtraction and subtract 8 from 25. The form of this problem is probably familiar to you:

1 15	Carry
~~2~~~~5~~	Minuend
-8	Subtrahend
17	Difference

It requires the use of the *borrow*; that is, you cannot subtract 8 from 5 and have a positive difference. You must borrow a 1, which is really one group of 10. Then, one group of 10 plus five groups of 1 equal 15, and 15 minus 8 leaves a difference of 7. The 2 was reduced by 1 by the borrow; and since nothing is to be subtracted from it, it is brought down to the difference.

Since the process of subtraction is the opposite of addition, the addition Table 1 may be used to illustrate subtraction facts for any number system we may discuss.

As we know, when adding,

$$X + Y = Z$$

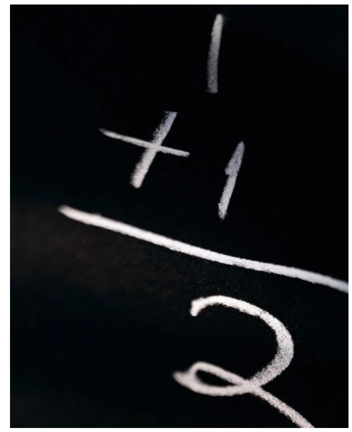
In subtraction, the reverse is true; that is,

$$Z - Y = X \qquad \text{or} \qquad Z - X = Y$$

Thus, in subtraction the minuend is always found in array Z and the subtrahend in either row X or column Y. If the subtrahend is in row X, then the remainder will be in column Y. Conversely, if the subtrahend is in column Y, then the difference will be in row X. For example, to subtract 8 from 15, find 8 in either the X row or Y column. Find where this row or column intersects with a value of 15 for Z; then move to the remaining row or column to find the difference.

Chapter 2
Binary Number System

The simplest possible number system is the *binary*, or base-2, system. You will be able to use the information just covered about the decimal system to easily relate the same terms to the binary system.



Unit and Number

The *base*, or radix -you should remember from our decimal section - is the number of symbols used in the number system. Since this is the base 2 system, only two symbols, 0 and 1, are used. The base is indicated by a subscript, as shown in the following example:

$$1_2$$

When you are working with the decimal system, you normally don't use the subscript. Now that you will be working with number systems other than the decimal system, it is important that you use the subscript so that you are sure of the system being referred to. Consider the following two numbers:

11 and 11

With no subscript you would assume both values were the same. If you add subscripts to indicate their base system, as shown below, then their values are quite different:

11_{10} is not the same as 11_2

The base ten number 11_{10} is eleven, but the base two number 11_2 is only equal to three in base ten.

There will be occasions when more than one number system will be discussed at the same time, so you MUST use the proper Subscript.

Positional Notation

As in the decimal number system, the principle of positional notation applies to the binary number system. You should recall that the decimal system uses powers of 10 to determine the value of a position.

The binary system uses powers of 2 to determine the value of a position. A bar graph showing the positions and the powers of the base is shown in the chart below:

2^4 2^3 2^2 2^1 2^0 . 2^{-1} 2^{-2} 2^{-3}
$2^4 = 2 * 2 * 2 * 2 = 16_{10}$ $2^3 = 2 * 2 * 2 = 8_{10}$ $2^2 = 2 * 2 = 4_{10}$ $2^1 = 2 * 1 = 2_{10}$ $2^0 = 1_{10}$ $2^{-1} = ½ = .5_{10}$ $2^{-2} = ½ * ½ = .25_{10}$ $2^{-3} = ½ * ½ * ½ = .125_{10}$

All numbers or values to the left of the radix point are whole numbers, and all numbers to the right of the radix point are fractional numbers.

Let's look at the binary number 101.1 on a bar graph:

2^2	2^1	2^0	.	2^{-2}
1	0	1	.	1

Working from the radix point to the right and left, you can determine the decimal equivalent:

$1 * 2^{-1} = 0.5_{10}$
$1 * 2^0 = 1.0_{10}$
$1 * 2^1 = 0.0_{10}$
$\underline{1 * 2^2 = 4.0_{10}}$
5.5_{10}

Table 2 provides a comparison of decimal and binary numbers. Notice that each time the total number of binary symbol positions increase, the binary number indicates the next higher power of 2. By this example, you can also see that more symbol positions are needed in the binary system to represent the equivalent value in the decimal system.

Table 2
Decimal and Binary Comparison

	Decimal	Binary	
10^0	0	0	2^0
	1	1	
	2	10	2^1
	3	11	
	4	100	2^2
	5	101	
	6	110	
	7	111	
	8	1000	2^3
	9	1001	
10^1	10	1010	
	11	1011	
	12	1100	
	13	1101	
	14	1110	
	15	1111	
	16	10000	2^4
	17	10001	
	18	10010	
	19	10011	
	20	10100	

MSD and LSD

When you're determining the MSD and LSD for binary numbers, use the same guidelines you used with the decimal system. As you read from left to right, the first nonzero digit you encounter is the MSD, and the last nonzero digit is the LSD.

0 **1** 0 1 0 0 1 1 . 0 0 **1** 0$_2$

If the number is a whole number, then the first digit to the left of the radix point is the LSD, *even if it is a zero*.

1 0 1 1 0 0 **1** .$_2$

1 0 1 0 1 0 1 **0** .$_2$

Here, as in the decimal system, the MSD is the digit that will have the most effect on the number; the LSD is the digit that will have the least effect on the number.

The two numerals of the binary system (1 and 0) can easily be represented by many electrical or electronic devices. For example, 1_2 may be indicated when a device is active (on), and 0_2 may be indicated when a device is non-active (off).

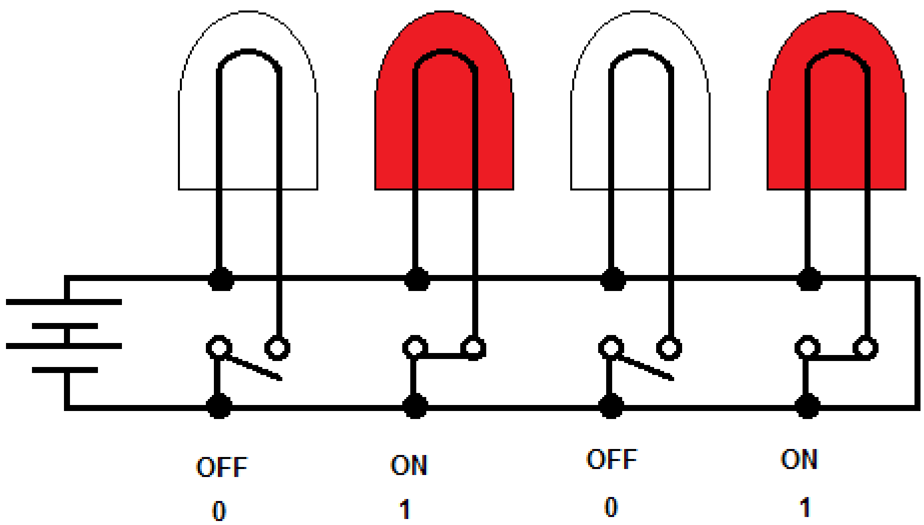


Figure 1

Look at the preceding figure. It illustrates a very simple binary counting device. Notice that 1_2 is indicated by a lighted lamp and 0_2 is indicated by an unlighted lamp. The reverse will work equally well.

The unlighted state of the lamp can be used to represent a binary 1 condition, and the lighted state can represent the binary 0 condition. Both methods are used in digital computer applications. Many other devices are used to represent binary conditions. They include switches, relays, diodes, transistors, and integrated circuits (ICs).

Addition of Binary Numbers

Addition of binary numbers is basically the same as addition of decimal numbers. Each system has an augend, an addend, a sum, and carries. The following example will refresh your memory:

1	Carry
15	Augend
+7	Addend
22	Sum

Since only two symbols, 0 and 1, are used with the binary system, only four combinations of addition are possible.

$$0 + 0$$
$$1 + 0$$
$$0 + 1$$
$$1 + 1$$

The sum of each of the first three combinations is obvious:

$$0 + 0 = 0_2$$
$$0 + 1 = 1_2$$
$$1 + 0 = 1_2$$

The fourth combination presents a different situation. The sum of 1 and 1 in any other number system is 2, but the numeral 2 does not exist in the binary system. Therefore, the sum of 1_2 and 1_2 is 10_2 (spoken as one zero base two), which is equal to 2_{10}.

1	Carry
1_2	Augend
$+1_2$	Addend
10_2	Sum

Study the following examples using the four combinations mentioned above:

101_2	Augend
$+010_2$	Addend
111_2	Sum

1	Carry
101_2	Augend
$+101_2$	Addend
1010_2	Sum

When a carry is produced, it is noted in the column of the next higher value or in the column immediately to the left of the one that produced the carry.

Example: Add 1011_2 and 1101_2.

Solution: Write out the problem as shown:

1011_2 Augend
$+1101_2$ Addend

As we noted previously, the sum of 1 and 1 is 2, which cannot be expressed as a single digit in the binary system. Therefore, the sum of 1 and 1 produces a carry:

1 Carry
1011_2 Augend
$+1101_2$ Addend
0_2

The following steps, with the carry indicated, show the completion of the addition:

1 ~~1~~ Carry
1011_2 Augend
$+1101_2$ Addend
00_2

When the carry is added, it is marked through to prevent adding it twice.

1 ~~1~~ ~~1~~ Carry
1011_2 Augend
$+1101_2$ Addend
000_2

1 ~~1~~ ~~1~~ ~~1~~ Carry
1011_2 Augend
$+1101_2$ Addend
11000_2

In the final step the remaining carry is brought down to the sum. In the following example you will see that more than one carry may be produced by a single column. This is something that does not occur in the decimal system.

Example: Add 1_2, 1_2, 1_2, and 1_2

1_2 Augend
1_2 1^{st} Addend
1_2 2^{nd} Addend
$\underline{+1_2}$ 3^{rd} Addend

The sum of the augend and the first addend is 0 with a carry. The sum of the second and third addends is also 0 with a carry. At this point the solution resembles the following example:

1 Carry
1 Carry
1_2 Augend
1_2 1^{st} Addend
1_2 2^{nd} Addend
$\underline{+1_2}$ 3^{rd} Addend
0_2

The sum of the carries is 0 with a carry, so the sum of the problem is as follows:

1 Carry
11 Carry
1_2 Augend
1_2 1^{st} Addend
1_2 2^{nd} Addend
$\underline{+1_2}$ 3^{rd} Addend
100_2

The same situation occurs in the following example: Add 100_2, 101_2, and 111_2

100_2 Augend
101_2 Addend
$\underline{+111_2}$ Addend

```
    1     Carry
  100₂  Augend
  101₂  Addend
 +111₂  Addend
    0₂  Sum
```

```
  1 1     Carry
  100₂  Augend
  101₂  Addend
 +111₂  Addend
   00₂  Sum
```

As in the previous example, the sum of the four 1s is 0 with two carries, and the sum of the two carries is 0 with one carry. The final solution will look like this:

```
    1      Carry
  1111     Carry
    100₂  Augend
    101₂  Addend
   +111₂  Addend
  10000₂  Sum
```

In the addition of binary numbers, you should remember the following binary addition rules:

Rule 1: $\mathbf{0_2 + 0_2 = 0_2}$

Rule 2: $\mathbf{1_2 + 0_2 = 1_2}$

Rule 3: $\mathbf{0_2 + 1_2 = 1_2}$

Rule 4: $\mathbf{1_2 + 1_2 = 10_2}$

Subtraction of Binary Numbers

Now that you are familiar with the addition of binary numbers, subtraction will be easy. The following are the four rules that you must observe when subtracting:

Rule 1: $\mathbf{0_2 - 0_2 = 0_2}$

Rule 2: $\mathbf{1_2 - 0_2 = 1_2}$

Rule 3: 1_2 - 1_2 = 0_2

Rule 4: 0_2 - 1_2 = 1_2 with a borrow

The following example (10110_2 - 1100_2) demonstrates the four rules of binary subtraction:

10110_2 Minuend
<u>- 1100_2</u> Subtrahend
?010_2 Difference

Rule 1: $0_2 - 0_2 = 0_2$
Rule 2: $1_2 - 0_2 = 1_2$
Rule 3: $1_2 - 1_2 = 0_2$
Rule 4: (See explanation below)

Rule 4 presents a different situation because you cannot subtract 1 from 0. Since you cannot subtract 1 from 0 and have a positive difference, you must borrow the 1 from the next higher order column of the minuend. The borrow may be indicated as shown below:

10 Borrow (fourth position is now 10)
0 After Borrow (fifth position is 0 after the borrow_
~~1~~0110_2 Minuend
<u>- 1100_2</u> Subtrahend
1010_2 Difference

Now, for another example, observe the following method of borrowing across more than one column for the case of 1000_2 -1_2:

10 Borrow
0 ~~10~~ ~~10~~ 10 After Borrow (base-2)
~~1~~000_2 Minuend
<u>- 1_2</u> Subtrahend
0111_2 Difference

Complementary Subtraction

Most digital computer systems cannot subtract - they can only add. Therefore, we need a method of adding that gives the results of subtraction. Does that sound confusing? Really, it is quite simple. A *complement* is used for our subtractions. A complement is something used to complete something else.

In most number systems you will find two types of complements. The first is the amount necessary to complete a number up to the highest number in the number system. In the decimal system, this would be the difference between a given number and all 9s. This is called the *nines complement* or the radix-1 or *R's-1 complement*. As an example, the nines complement of 254 is 999 minus 254, or 745.

The second type of complement is the difference between a number and the next higher power of the number base. As an example, the next higher power of 10 above 999 is 1,000. The difference between 1,000 and 254 is 746. This is called the *tens complement* in the decimal number system. It is also called the radix or *R's complement*. We will use complements to subtract. Let's look at the magic of this process.

There are three important points we should mention before we start:

1. Never complement the minuend in a problem,
2. Always disregard any carry beyond the number of positions of the largest of the original numbers, and
3. Add the R's complement of the original subtrahend to the original minuend. This will have the same effect as subtracting the original number.

Let's look at a base ten example in which we subtract 38 from 59:

```
 59
- 38
 21
```

Since the subtrahend is -38, the R's-complement is 100 – 38 = 62. Using +62, we have,

```
  59
+ 62   Add the R's Complement of 38
 121
```

Removing the MSD from the result we have the solution, 21.

Now let's look at the number system that most computers use, the binary system. Just as the decimal system, had the nines (R's-1) and tens (R's) complement, the binary system has two types of complement methods. These two types are the ones (R's-1) complement and the twos

(R’s) complement. The binary system R’s-1 complement is the difference between the binary number and all 1s. The R’s complement is the difference between the binary number and the next higher power of 2.

Let’s look at a quick and easy way to form the R’s-1 complement. To do this, change each 1 in the original number to 0 and each 0 in the original number to 1 as has been done in the example below.

The original number is, 1011011_2

The R’s-1 complement is, 0100100_2

There are two methods of achieving the R’s complement. In the first method we perform the R’s-1 complement and then add 1. This is much easier than subtracting the original number from the next higher power of 2. If you had subtracted, you would have had to borrow. Saying it another way, to reach the R’s complement of any binary number, change all 1s to 0s and all 0s to 1s, and then add 1.

To find the ***R’s complement*** of any binary number, change all 1s to 0s and all 0s to 1s, and then add 1.

As an example let’s determine the R’s complement of 10101101_2:

Step 1 – R’s – 1 complement: 01010010_2
Step 2 – Add 1: $+ \underline{\qquad 1_2}$
01010011_2

The second method of obtaining the R’s complement will be demonstrated on the binary number 00101101100_2.

Step 1—Start with the LSD, working to the MSD, writing the digits as they are up to and including the first one.

0010110100_2

Which yields, 100_2

Step 2—Now R's-1 complement the remaining digits:

1101001100_2

For another example, let's R's complement 1001100_2 the same number using both methods:

Method 1

First switch zeros and ones. The original number, 1001100_2, becomes, 0110011_2. Next add 1_2. See below,

1001100_2
0110011_2
$+ \quad 1_2$
0110100_2

Method 2

1001100_2
0110100_2 R's-1 complement

Now let's do some subtracting by using the R's complement method. We will go through the subtraction of 3_{10} from 9_{10} (0011_2 from 1001_2):

9_{10} 1001_2 Minuend
-3_{10} -0011_2 Subtrahend

Step 1- Leave the minuend alone:

1001_2 remains 1001_2

Step 2 - Using either method, R's complement of the subtrahend:

1101_2 R's complement of subtrahend

Step 3 -Add the R's complement found in step 2 to the minuend of the original problem:

1001_2
$+ 1101_2$
10110_2

Step 4 - Remember to discard any carry beyond the size of the original number. Our original problem had four digits, so we discard the carry that expanded the difference to five digits.

$1001_2 = 9_{10}$
$-0011_2 = -3_{10}$
$1\ 0110_2 = 6_{10}$

This carry we disregard is significant to the computer. It indicates that the difference is positive. Because we have a carry, we can read the difference directly without any further computations.

If we do *not* have a carry, it indicates the difference is a negative number. In that case, the difference must be R's complemented to produce the correct answer.

Let's look at an example that will explain this for you.

Subtract 9_{10} from 5_{10} (1001_2 from 0101_2):

5_{10} 0101_2 Minuend
-9_{10} -1001_2 Subtrahend
-4_{10}

Step 1 - Leave the minuend alone:

0101_2 remains 0101_2

Step 2 - R's complement the subtrahend:

0111_2 R's complement of subtrahend

Step 3 - Add the R's complement found in step 2 to the minuend of the original problem:

0101_2 Original minuend
$+0111_2$ *Two's* complement
1100_2 Difference of original problem

Step 4 - We do *not* have a carry; and this tells us, and any computer, that our difference (answer) is negative. With no carry, we must R's complement the difference in step 3. We will then have arrived at the answer (difference) to our original problem. R's complement of difference in step 3 is 0100_2.

Remember, we had no carry in step 3. That showed us our answer was going to be negative. Make sure you indicate the difference is negative.

Chapter 3
Octal Number System

The *octal*, or base-8, number system is a common system used with computers. Because of its relationship with the binary system, it is useful in programming some types of computers. Look closely at the comparison of binary and octal number systems in Table 3. You can see that one octal digit is the equivalent value of three binary digits. The following examples of the conversion of octal 225_8 to binary and back again further illustrate this comparison:

Octal to Binary

2	2	5_8
010	010	101_2

Binary to Octal

010	010	101_2
2	2	5_8

Table 3
Binary and Octal Comparison

	Binary	Octal	
2^0	0	0	8^0
	1	1	
2^1	10	2	
	11	3	
2^2	100	4	
	101	5	
	110	6	
	111	7	
2^3	1000	10	8^1
	1001	11	
	1010	12	
	1011	13	
	1100	14	
	1101	15	
	1110	16	
	1111	17	
2^4	10000	20	8^2
	10001	21	
	10010	22	
	10011	23	
	10100	24	
	10101	25	

	10110	26	
	10111	27	
	11000	30	8^3

Unit and Number

The terms that you learned in the decimal and binary sections are also used with the octal system. The unit remains a single object, and the number is still a symbol used to represent one or more units.

Base (Radix)

As with the other systems, the radix, or base, is the number of symbols used in the system. The octal system uses eight symbols - 0 through 7. The base, or radix, is indicated by the subscript 8.

Positional Notation

The octal number system is a positional notation number system. Just as the decimal system uses powers of 10 and the binary system uses powers of 2, the octal system uses power of 8 to determine the value of a number's position. The following bar graph shows the positions and the power of the base:

$$8^3\ 8^2\ 8^1\ 8^0 \bullet 8^{-1}\ 8^{-2}\ 8^{-3}$$

Remember, that the power, or exponent, indicates the number of times the base is multiplied by itself. The value of this multiplication is expressed in base 10 as shown below: All numbers to the left of the radix point are whole numbers, and those to the right are fractional numbers.

$8^3 = 8 * 8 * 8 = 512_{10}$
$8^2 = 8 * 8 = 64_{10}$
$8^1 = 8_{10}$
$8^0 = 1_{10}$
$8^{-1} = \frac{1}{8} = 0.125^{10}$
$8^{-2} = \frac{1}{(8*8)} = 0.15625^{10}$
$8^{-3} = \frac{1}{(8*8*8)} = 1/512 = 0.0019531^{10}$

MSD and LSD

When determining the most and least significant digits in an octal number, use the same rules that you used with the other number systems. The digit farthest to the left of the radix point is the MSD, and the one farthest right of the radix point is the LSD.

Example:

$$4\ 7\ 3\ 2\ .\ 2\ 6\ 1_8$$

If the number is a whole number, the MSD is the nonzero digit farthest to the left of the radix point and the LSD is the digit immediately to the left of the radix point. Conversely, if the number is a fraction only, the nonzero digit closest to the radix point is the MSD and the LSD is the nonzero digit farthest to the right of the radix point.

Addition of Octal Numbers

The addition of octal numbers is not difficult provided you remember that anytime the sum of two digits exceeds 7, a carry is produced. Compare the two examples shown below:

$$\begin{array}{r} 4_8 \\ \underline{+2_8} \\ 6_8 \end{array} \qquad \begin{array}{r} 4_8 \\ \underline{+4_8} \\ 10_8 \end{array}$$

The octal addition table in Table 4 will be of benefit to you until you are accustomed to adding octal numbers.

Table 4
Octal Addition Table

		Y							
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
	0	0	1	2	3	4	5	6	7
	1	1	2	3	4	5	6	7	10
	2	2	3	4	5	6	7	10	11
X	**3**	3	4	5	6	7	10	11	12
	4	4	5	6	7	10	11	12	13
	5	5	6	7	10	11	12	13	14
	6	6	7	10	11	12	13	14	15
	7	7	10	11	12	13	14	15	16

To use the table, simply follow the directions used in this example:

Add: 6_8 and 5_8

Locate the 6 in the X row of the figure. Next locate the 5 in the Y column. The point where these two intersect is the sum. Therefore,

$$\begin{array}{r} 6_8 \\ \underline{+5_8} \\ 13_8 \end{array}$$

If you use the concepts of addition you have already learned, you are ready to add octal numbers.

Work through the solutions to the following problems:

1 1	Carry
456_8	Augend
$+123_8$	Addend
601_8	Sum

11111	Carry
77714_8	Augend
$+ \quad 76_8$	Addend
100012_8	Sum

As was mentioned earlier in this section, each time the sum of a column of numbers exceeds 7, a carry is produced. More than one carry may be produced if there are three or more numbers to be added, as in this example:

7_8	Augend
7_8	Addend
$+7_8$	Addend

The sum of the augend and the first addend is 6_8 with a carry. The sum of 6_8 and the second addend is 5_8 with a carry. You should write down the 5_8 and add the two carries and bring them down to the sum, as shown below:

1	Carry
1	Carry
7_8	Augend
$+ 7_8$	1st Addend
6_8	Subsum
$+ 7_8$	2nd Addend
25_8	Sum

Subtraction of Octal Numbers

The subtraction of octal numbers follows the same rules as the subtraction of numbers in any other number system. The only variation is in the quantity of the borrow. In the decimal system, you had to borrow a group of 10_{10}. In the binary system, you borrowed a group of 2_{10}. In the octal system you will borrow a group of 8_{10}.

Consider the subtraction of 1 from 10 in decimal, binary, and octal number systems:

Decimal	**Binary**	**Octal**
10_{10}	10_2	10_8
-1_{10}	-1_2	-1_8
9_{10}	1_2	7_8

In each example, you cannot subtract 1 from 0 and have a positive difference. You must use a borrow from the next column of numbers. Let's examine the above problems and show the borrow as a *decimal* quantity for clarity:

Decimal	**Binary**	**Octal**	
10	2	8	Borrow
~~1~~0_{10}	~~1~~0_2	~~1~~0_8	
-1_{10}	-1_2	-1_8	
9_{10}	1_2	7_8	

When you use the borrow, the column you borrow from is reduced by 1, and the amount of the borrow is added to the column of the minuend being subtracted. The following examples show this procedure:

10	Borrow
2	After Borrow
~~3~~4$_{10}$	Minuend
-9$_{10}$	Subtrahend
25$_{10}$	Difference

10	Borrow
3	After Borrow
46$_8$	Minuend
-7$_8$	Subtrahend
37$_8$	Difference

In the octal example, 7_8 cannot be subtracted from 6_8, so you must borrow from the 4. Reduce the 4 by 1 and add 10_8 (the borrow) to the 6_8 in the minuend. By subtracting 7_8 from 16_8, you get a difference of 7_8. Write this number in the difference line and bring down the 3. You may need to refer to Table 4, the octal addition table, until you are familiar with octal numbers. To use the table for subtraction, follow these directions. Locate the subtrahend in column Y. Now find where this line intersects with the minuend. The remainder, or difference, will be in row X directly above this point.

Chapter 4
Hexadecimal (Hex) Number System

The *hex* number system is a more complex system in use with computers. The name is derived from the fact the system uses 16 symbols. It is beneficial in computer programming because of its relationship to the binary system. Since 16 in the decimal system is the fourth power of 2 (or 24); one hex digit has a value equal to four binary digits. Table 5 shows the relationship between the two systems.

Table 5
Binary and Hexadecimal Comparison

Binary	Hexadecimal
0	0
1	1
10	2
11	3
100	4
101	5
110	6
111	7
1000	8
1001	A
1010	B
1011	C
1100	D
1110	E
1111	F
10000	10
10001	11
10010	12
10011	13
10100	14
10101	15
10110	16
10111	17
11000	18
11001	19
11010	1A
11011	1B
11100	1C

Unit and Number

As in each of the previous number systems, a unit stands for a single object. A number in the hex system is the symbol used to represent a unit or quantity. The Arabic numerals 0 through 9 are used along with the first six letters of the alphabet. You have probably used letters in math problems to represent unknown quantities, but in the hex system A, B, C, D, E, and F, each have a definite value as shown below:

$$A_{16} = 10_{10}$$

$$B_{16} = 11_{10}$$

$$C_{16} = 12_{10}$$

$$D_{16} = 13_{10}$$

$$E_{16} = 14_{10}$$

$$F_{16} = 15_{10}$$

Base (Radix)

The base, or radix, of this system is 16, which represents the number of symbols used in the system.

A quantity expressed in hex will be annotated by the subscript 16, as shown below:

$A3EF_{16}$

Positional Notation

Like the binary, octal, and decimal systems, the hex system is a positional notation system. Powers of 16 are used for the positional values of a number. The following bar graph shows the positions:

$$16^3\ 16^2\ 16^1\ 16^0\ \bullet\ 16^{-1}\ 16^{-2}\ 16^{-3}$$

Multiplying the base times itself the number of times indicated by the exponent will show the equivalent decimal value:

$16^3 = 16 * 16 * 16 = 4{,}096_{10}$

$16^2 = 16 * 16 = 256_{10}$

$16^1 = 16_{10}$

$16^0 = 1_{10}$

$16^{-1} = \frac{1}{16}$ $1/16 = 0.0625_{10}$

$16^{-2} = \frac{1}{(16*16)} = 0.0039062_{10}$

$16^{-3} = \frac{1}{(16*16*16)} = 0.0002441_{10}$

You can see from the positional values that usually fewer symbol positions are required to express a number in hex than in decimal. The following example shows this comparison:

625_{16} is equal to 1573_{10}

MSD and LSD

The most significant and least significant digits will be determined in the same manner as the other number systems. The following examples show the MSD and LSD of whole, fractional, and mixed hex numbers:

7 9 E **4** . $_{16}$

0 . **1** 8 2 **A**$_{16}$

3 B C . E 4 2 **F**$_{16}$

Addition of Hex Numbers

The addition of hex numbers may seem intimidating at first glance, but it is no different than addition in any other number system. The same rules apply. Certain combinations of symbols produce a carry while others do not. Some numerals combine to produce a sum represented by a letter. After a little practice you will be as confident adding hex numbers as you are adding decimal numbers.

Study the hex addition table in Table 6. Using the table, add 7 and 7. Locate the number 7 in both Row X and column Y. The point where these two intersect is the sum; in this case 7 + 7 = E. As long as the sum of two numbers is 15_{10} or less, only one symbol is used for the sum. A carry will be produced when the sum of two numbers is 16_{10} or greater, as in the following examples:

8_{16} A_{16} D_{16}
$+8_{16}$ $+D_{16}$ $+9_{16}$
10_{16} 17_{16} 16_{16}

Table 6

Hexadecimal Addition Table

		Y															
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
	0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	1	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10
	2	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11
	3	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12
	4	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13
	5	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14
	6	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15
X	**7**	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16
	8	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17
	9	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18
	A	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19
	B	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A
	C	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B
	D	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C
	E	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
	F	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E

Use the addition table and follow the solution of the following problems:

456_{16} Augend
$+784_{16}$ Addend
BDA_{16} Sum

In this example each column is straight addition with no carry.

Now add the addend (784_{16}) and the sum (BDA_{16}) of the previous problem:

11	Carry
784_{16}	Augend
+ BDA_{16}	Addend
$135E_{16}$	Sum

Here the sum of 4 and A is E. Adding 8 and D is 15_{16}; write down 5 and carry a 1. Add the first carry to the 7 in the next column and add the sum, 8, to B. The result is 13_{16}; write down 3 and carry a 1. Since only the last carry is left to add, bring it down to complete the problem.

Now observe the procedures for a more complex addition problem. You may find it easier to add the Arabic numerals in each column first:

1 1 1	Carry
$C14_{16}$	Augend
$19E_{16}$	Addend
571_{16}	Addend
+ $BB3_{16}$	Addend
$1ED6_{16}$	Sum

The sum of 4, E, 1, and 3 in the first column is 16_{16}. Write down the 6 and the carry. In the second column, 1, 1, 9, and 7 equals 12_{16}. Write the carry over the next column. Add B and 2 the sum is D. Write this in the sum line. Now add the final column, 1, 1, 5, and C. The sum is 13_{16}. Write down the carry; then add 3 and B – which yields a sum of E. Write down the E and bring down the final carry to complete the problem.

Subtraction of Hex Numbers

The subtraction of hex numbers looks more difficult than it really is. In the preceding sections you learned all the rules for subtraction. Now you need only to apply those rules to a new number system. The symbols may be different and the amount of the borrow is different, but the rules remain the same.

Use the hex addition table (Table 6) to follow the solution of the following problems:

ABC_{16}	Minuend
- 642_{16}	Subtrahend

Working from left to right, first locate the subtrahend "2" in column Y. Follow this line down until you reach C. The difference is located in row X directly across from the C — which in this case is A. Use this same procedure to reach the solution:

ABC_{16} Minuend
$- \ 642_{16}$ Subtrahend
$47A_{16}$ Difference

Now examine the following problem:

$1E9C4_{16}$ Minuend
$- \ F4A1_{16}$ Subtrahend
$F523_{16}$ Difference

In this example, when F was subtracted from 1E, a borrow was used. Since you cannot subtract F from E and have a positive difference, a borrow of 10_{16} was taken from the next higher value column. The borrow was added to E, and the higher value column was reduced by 1.

The following example shows the use of the borrow in a more difficult problem:

10_{16} Borrow
2 Minuend reduced by 1
4A~~3~~7$_{16}$ Minuend
$- \ 2C4B_{16}$ Subtrahend
C_{16} Difference

In this first step, B cannot be subtracted from 7, so you take a borrow of 10_{16} from the next higher value column. Add the borrow to the 7 in the minuend; then subtract (17_{16} minus B_{16} equals C_{16}). Reduce the number from which the borrow was taken "3" by 1.

To subtract 4_{16} from 2_{16} also requires a borrow, as shown below:

10_{16} 10_{16} Borrow
8 2 Minuend reduced by 1
4~~A3~~7$_{16}$ Minuend
$- \ 2C4B_{16}$ Subtrahend
EC_{16} Difference

Borrow 10_{16} from the A and reduce the minuend by 1. Add the borrow to the 2 and subtract 4_{16} from 12_{16}. The difference is E.

When solved the problem looks like this:

$10_{16}10_{16}\ 10_{16}$	Borrow
392	Minuend reduced by 1
~~4A37~~$_{16}$	Minuend
- $2C4B_{16}$	Subtrahend
$1DEC_{16}$	Difference

Remember that the borrow is 10_{16} not 10_{10}.

There may be times when you need to borrow from a column that has a 0 in the minuend. In that case, you borrow from the next highest value column, which will provide you with a value in the 0 column that you can borrow from.

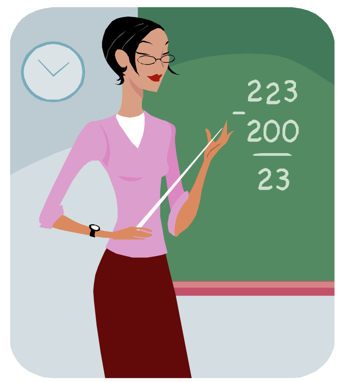
F	Borrow reduced by 1
~~10_{16}~~ 10_{16}	Borrow
1	Minuend reduced by 1
~~2~~07_{16}	Minuend
- A_{16}	Subtrahend
$1FD_{16}$	Difference

To subtract A from 7, you must borrow. To borrow you must first borrow from the 2. The 0 becomes 10_{16}, which can give up a borrow. Reduce the 10_{16} by 1 to provide a borrow for the 7. Reducing 10_{16} by 1 equals F. Subtracting A_{16} from 17_{16} gives you D16. Bring down the 1 and F for a difference of $1FD_{16}$.

Chapter 5
Conversion of Bases

We mentioned in the introduction that digital computers operate on electrical pulses. These pulses or the absence of, are easily represented by binary numbers. A pulse can represent a binary 1, and the lack of a pulse can represent a binary 0 or vice versa.

The chapters that discussed octal and hex numbers both mentioned that their number systems were beneficial to programmers. You will see later in this section that octal and hex numbers are easily converted to binary numbers and vice versa.



There are many times when it will be necessary to convert decimal numbers to binary, octal, and hex numbers. You will also have to be able to convert binary, octal, and hex numbers to decimal numbers. Converting each number system to each of the others will be explained. This will prepare you for converting from any base to any other base when needed.

Decimal Conversion

Some computer systems have the capability to convert decimal numbers to binary numbers. They do this by using additional circuitry. Many of these systems require that the decimal numbers be converted to another form before entry.

Decimal to Binary

Conversion of a decimal number to any other base is accomplished by dividing the decimal number by the radix of the system you are converting to. The following definitions identify the basic terms used in division:

Dividend - the number to be divided
Divisor - the number by which a dividend is divided
Quotient - the number resulting from the division of one number by another
Remainder - the final undivided part after division that is less or of a lower degree than the divisor

To convert a base-10 whole number (in this example, 5_{10}) to its binary equivalent, first set up the problem for division:

2) 5

Step 1—Divide the base-10 number by the radix (2) of the binary system and extract the remainder (this becomes the binary number's LSD).

```
    2
2 / 5
   -4
    1
```

Step 2—Continue the division by dividing the quotient of step 1 by the radix (2 divided by 2.)

```
    1
2 / 2
   -2
    0
```

Step 3—Continue dividing quotients by the radix until the quotient becomes smaller that the divisor; then do one more division. The remainder is our MSD.

```
    0
2 / 1
   -0
    1
```

```
    0
2 / 0
   -0
    0
```

The remainder in step 1 is our LSD. Now rewrite the solution, and you will see that 5_{10} equals 101_2. For the remainder of this course, we will use the following format to show the math. Using the same problem as above, to convert 5_{10} to binary as follows,

Divisor	Dividend	Quotient	Remainder	
2	5	2	5 - (2*2) = 1	LSD
2	2	1	2 - (2*1) = 0	
2	1	0	1 - (2*0) = 1	MSD
Therefore, $5_{10} = 101_2$				

Now follow the conversion of 23_{10} to binary:

Step 1—Set up the problem for division:

2) 23_{10}

Step 2—Set up the table and work through the division:

Divisor	Dividend	Quotient	Remainder	
2	23	11	23 - (2*11) = 1	LSD
2	11	5	11 - (2*5) = 1	
2	5	2	5 - (2*2) = 1	
2	2	1	2 - (2*1) = 0	
2	1	0	1 - (2*0) = 1	MSD

As you can see, dividing 2 into 23 yields a quotient of 11 and a remainder of 1. This is our Least Significant Digit (LSD.) We continue on with this process until the quotient is less than the divisor.

Step 3—Rewrite the solution from MSD to LSD:

10111_2

No matter how large the decimal number may be, we use the same procedure. Let's try the problem below. It has a larger dividend:

2) 105

Setting up the table we have,

Divisor	Dividend	Quotient	Remainder	
2	105	52	105 - (2*52) = 1	LSD
2	52	26	52 - (2*26) = 0	
2	26	13	26 - (2*13) = 0	
2	13	6	13 - (2*6) = 1	
2	6	3	6 - (2*3) = 0	
2	3	1	3 - (2*1) = 1	
2	1	0	1 - (2*0) = 1	MSD

Therefore, $105_{10} = 1101001_2$

We can convert fractional decimal numbers by multiplying the fraction by the radix and extracting the portion of the product to the *left* of the radix point. Continue to multiply the fractional portion of the previous product until the desired degree of accuracy is attained.

Let's go through this process and convert 0.25_{10} to its binary equivalent:

Base	Value	Product	left of Radix Point		Remainder
2	0.25	0.50	0	MSD	.50
2	.50	1.00	1	LSD	.00

The *first* figure to the left of the radix point is the MSD, and the last figure of the computation is the LSD. Rewrite the solution from MSD to LSD preceded by the radix point as shown:

$.01_2$
Therefore 0.25_{10} is 0.01_2.

Now try converting $.625_{10}$ to binary:

0.625
x 2
MSD 1 ← 1.25

0.25
x 2
0 ← 0.50

0.50
x 2
LSD 1 ← 1.00

0.00
x 2
0 ← 0.00

Therefore, $.635_{10} = 0.101_2$

Just like division, this problem can be set up in a table. The following table shows how the previous problem should be set up and worked.

Base	Number	Resultant	Left of Radix Point		Remainder
2	.625	1.250	1	MSD	.250
2	.250	0.500	0		.500
2	.500	1.000	1	LSD	.000
2	.000	0	0		.000

Therefore, $0.625_{10} = .101_2$

As we mentioned before, you should continue the operations until you reach the desired accuracy. For example, convert $.425_{10}$ to five places in the binary system:

Base	Value	Product	left of Radix Point		Remainder
2	0.425	0.850	0	MSD	.850
2	.850	1.700	1		.700
2	.700	1.400	1		.400
2	.400	0.800	0		.800
2	.800	1.600	1		.600
2	.600	1.200	1		.200
2	.200	0.400	0	LSD	.400

Therefore, $0.425_{10} = .0110110_2$

Although the multiplication was carried out for seven places, you would only use what is required. Write out the solution as shown:

$.01101_2$

To convert a mixed number such as 37.625_{10} to binary, split the number into its whole and fractional components and solve each one separately. Remember the whole number portion is solved using division and the fractional portion is solved using multiplication. In this problem carry the fractional part to four places. When the conversion of each is completed, recombine it with the radix point as shown below:

$37_{10} = 100101_2$

and

$.625_{10} = .1010_2$

Therefore,

$37.625_{10} = 100101.1010_2$

Decimal to Octal

The conversion of a decimal number to its base-8 equivalent is done by the repeated division method.

You simply divide the base-10 number by 8 and extract the remainders. The first remainder will be the LSD, and the last remainder will be the MSD.

Look at the following example. To convert 15_{10} to octal, set up the problem for division:

$$8\overline{)15_{10}}$$

Since 8 goes into 15 one time with a 7 remainder, 7 then is the LSD. Next divide 8 into the quotient (1). The result is a 0 quotient with a 1 remainder. The 1 is the MSD. See the following chart:

Divisor	Dividend	Quotient	Remainder	
8	15_{10}	1	15 - (8*1) = 7	LSD
8	1	0	1 - (8*0) = 1	MSD

Now write out the number from MSD to LSD as shown:

17_8

The same process is used regardless of the size of the decimal number. Naturally, more divisions are needed for larger numbers, as in the following example:

Convert 264_{10} to octal:

$$8\overline{)264_{10}}$$

Divisor	Dividend	Quotient	Remainder	
8	264_{10}	33	264 - (8*33) = 0	LSD
8	33	4	33 - (8*4) = 1	
8	4	0	4 - (8*0) = 4	MSD

By rewriting the solution, you find that the octal equivalent of 264_{10} is as follows:

410_8

To convert a decimal fraction to octal, *multiply* the fraction by 8. Extract everything that appears to the left of the radix point. The first number extracted will be the MSD and will follow the radix point. The last number extracted will be the LSD.

Convert 0.05_{10} to octal:

Base	Value	Product	left of Radix Point		Remainder
8	.05	0.40	0	MSD	.40
8	.40	3.20	3		.20
8	.20	1.60	1		.60
8	.60	4.80	4		.80
8	.80	6.40	6	LSD	.40

Write the solution from MSD to LSD:

$.03146_8$

You can carry the conversion out to as many places as needed, but usually four or five places are enough.

To convert a mixed decimal number to its octal equivalent, split the number into whole and fractional portions and solve as shown below:

Convert 105.589_{10} to octal:

In this case we handle the integer separately from the fractional part of the number. Beginning with the integer, 105_{10},

Divisor	Dividend	Quotient	Remainder	Remaider	
8	105	13	105 – (8*13) = 1	1	LSD
8	13	1	13 – (8*1) = 5	5	
8	1	0	1 – (8*0) = 1	1	MSD

In octal, 105_{10} is $151_{8.}$ Now for the fractional component $.589_{10}$,

Base	Value	Product	Left of Radix Point		Remainder
8	.589	4.712	4	MSD	.712
8	.712	5.696	5		.696
8	.696	5.568	5		.568
8	.568	4.544	4	LSD	.544

In octal, $.589_{10}$ is $.4554_{8.}$

Combining the portions into a mixed number, the base-10 number 105.589_{10} is $151.4554_{8.}$

Decimal to Hex

To convert a decimal number to base 16, follow the repeated division procedures you used to convert to binary and octal, only divide by 16. Let's look at an example:

Convert 63_{10} to hex:

Divisor	Dividend	Quotient	Remainder		
16	63_{10}	3	63 - (16*3) = 15	$15_{10} = F_{16}$	LSD
16	3	0	3 - (16*0) = 3	$3_{10} = 3_{16}$	MSD

Therefore, the hex equivalent of 63_{10} is $3F_{16}$. You have to remember that the remainder is in base-10 and must be converted to hex if it exceeds 9.

Let's work through another example: Convert 174_{10} to hex:

Divisor	Dividend	Quotient	Remainder		
16	174_{10}	10	174 - (16*10) = 14	$14_{10} = E_{16}$	LSD
16	10	0	10 - (16*0) = 10	$10_{10} = A_{16}$	MSD

Write the solution from MSD to LSD:

AE_{16}
Therefore, $174_{10} = AE_{16}$

There will probably be very few times when you will have to convert a decimal fraction to a hex fraction. If the occasion should arise, the conversion is done in the same manner as binary or octal. Use the following example as a pattern:

Convert 0.695_{10} to hex:

Base	Value	Product	Left of Radix Point	Conversion		Remainder
16	.695	11.120	11	$11_{10} = B_{16}$	MSD	.120
16	.120	1.92	1	$1_{10} = 1_{16}$		.920
16	.920	14.720	14	$14_{10} = E_{16}$		.720
16	.720	11.520	11	$11_{10} = B_{16}$	LSD	.520

Therefore, $0.695_{10} = .B1EB_{16}$

Should you have the need to convert a decimal mixed number to hex, convert the whole number and the fraction separately; then recombine for the solution.

The converting of binary, octal, and hex numbers to their decimal equivalents is covered as a group later in this section.

Binary Conversion

Earlier we mentioned that the octal and hex number systems are useful to computer programmers. It is much easier to provide data to a computer in one or the other of these systems. Likewise, it is important to be able to convert data from the computer into one or the other number systems for ease of understanding the data.

Binary to Octal

Look at the following numbers:

10111001001101_2

27115_8

You can easily see that the octal number is much easier to say. Although the two numbers look completely different, they are equal.

Since 8 is equal to 2^3, then one octal digit can represent three binary digits, as shown below:

$0_8 = 000_2$
$1_8 = 001_2$
$2_8 = 010_2$
$3_8 = 011_2$
$4_8 = 100_2$
$5_8 = 101_2$
$6_8 = 110_2$
$7_8 = 111_2$

With the use of this principle, the conversion of a binary number is quite simple. As an example, follow the conversion of the binary number at the beginning of this section.

Write out the binary number to be converted. Starting at the radix point and moving left, break the binary number into groups of three as shown. This grouping of binary numbers into groups of three is called *binary-coded octal* (BCO). Add 0s to the left of any MSD that will fill a group of three:

$$010\ 111\ 001\ 001\ 101._2$$

Next, write down the octal equivalent of each group:

010	111	001	001	101_2
2	7	1	1	5_8

To convert a binary fraction to its octal equivalent, starting at the radix point and moving right, expand each digit into a group of three:

$$.100\ 001\ 110\ 011$$

Add 0s to the right of the LSD if necessary to form a group of three. Now write the octal digit for each group of three, as shown below:

.100	001	110	011_2
. 2	1	6	3_8

To convert a mixed binary number, starting at the radix point, form groups of three both right and left:

101	101	110 .	001	110_2
5	5	6 .	1	6_8

Binary to Hex

The table below shows the relationship between binary and hex numbers. You can see that four binary digits may be represented by one hex digit. This is because 16 is equal to 2^4.

Binary	Hex
0	0
1	1
10	2
11	3
100	4

101	5
110	6
111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Using this relationship, you can easily convert binary numbers to hex. Starting at the radix point and moving either right or left, break the number into groups of four. The grouping of binary into four bit groups is called binary-coded hexadecimal (BCH).

Convert 111010011_2 to hex:

0001	1101	$0011._2$
1	D	$3._{16}$

Therefore the answer is $1D3_{16}$

Add 0s to the left of the MSD of the whole portion of the number and to the right of the LSD of the fractional part to form a group of four.

Convert $.111_2$ to hex:

Since it is a three digit number, add a zero in the first position as shown below (in red),

$.111\mathbf{0}_2$

$.E_{16}$

Therefore the answer is $.E_{16}$

In this case, if a 0 had not been added, the conversion would have been $.7_{16}$, which is incorrect.

Octal Conversion

The conversion of one number system to another, as we explained earlier, is done to simplify computer programming or interpreting of data.

Octal to Binary

For some computers to accept octal data, the octal digits must be converted to binary. This process is the reverse of binary to octal conversion.

To convert a given octal number to binary, write out the octal number in the following format. We will convert octal 567_8. Next, below each octal digit write the corresponding three-digit binary-coded octal equivalent:

5	6	7_8
101	110	111_2

Therefore, 567_8 equals 101 110 111_2

Remove the conversion from the format:

101110111_2

As you gain experience, it may not be necessary to use the block format.

An octal fraction ($.123_8$) is converted in the same manner, as shown below:

. 1	2	3
.001	010	011_2

Solution: $.123_8$ equals $.001010011_2$

Apply these principles to convert mixed numbers as well.

Convert 32.25_8 to binary:

3	2 .	2	5_8
011	010 .	010	101_2

Therefore, 32.25_8 is 011010.010101_2

Octal to Hex

You will probably not run into many occasions that call for the conversion of octal numbers to hex. Should the need arise, conversion is a two-step procedure. Convert the octal number to binary; then convert the binary number to hex. The steps to convert 53.7_8 to hex are shown below:

5	3.	7_8
101	011.	111_2

Regroup the binary digits into groups of four and add zeros where needed to complete groups; then convert the binary to hex.

0010	1011.	1110_2
2	B.	E_{16}

Therefore, 53.7_8 is $2B.E_{16}$

Hex Conversion

The procedures for converting hex numbers to binary and octal are the reverse of the binary and octal conversions to hex.

Hex to Binary

To convert a hex number to binary, set up the number in the block format you used in earlier conversions. Below each hex digit, write the four-digit binary equivalent. Observe the following example:

Convert ABC_{16} to binary:

A	B	C_{16}
1010	1011	1100_2

Therfore, ABC_{16} is 101010111100_2

Hex to Octal

Just like the conversion of octal to hex, conversion of hex to octal is a two-step procedure. First, convert the hex number to binary; and second, convert the binary number to octal. Let's use the same example we used above in the hex to binary conversion and convert it to octal:

A	B	C_{16}
1010	1011	1100_2

101	010	111	100_2
5	2	7	4_8

Conversion to Decimal

Computer data will have little meaning to you if you are not familiar with the various number systems. It is often necessary to convert those binary, octal, or hex numbers to decimal numbers. The need for understanding is better illustrated by showing you a paycheck printed in binary. A check in the amount of $\$10,010,101.00_2$ looks impressive but in reality only amounts to $\$149.00_{10}$

Binary to Decimal

The computer that calculates your pay probably operates with binary numbers, so a conversion takes place in the computer before the amount is printed on your check. Some computers, however, don't automatically convert from binary to decimal. There may be times when you must convert mathematically.

To convert a base-2 number to base-10, you must know the decimal equivalent of each power of 2.

The decimal value of a power of 2 is obtained by multiplying 2 by itself the number of times indicated by the exponent for whole numbers; for example, $2^4 = 2 * 2 * 2 * 2$ or 16_{10}.

For fractional numbers, the decimal value is equal to 1 divided by 2 multiplied by itself the number of times indicated by the exponent. Look at this example:

$$2^{-3} = \frac{1}{2*2*2} = 0.125_{10}$$

The table below shows a portion of the positions and decimal values of the binary system: Remember, any non-zero number to the 0 power is equal to 1_{10}.

2^5	2^4	2^3	2^2	2^1	2^0	.	2^{-1}	2^{-1}	2^{-3}
32	16	8	4	2	1	.	0.5	0.25	0.125

Another method of determining the decimal value of a position is to multiply the preceding value by 2 for whole numbers and to divide the preceding value by 2 for fractional numbers. Remember,

- Multiply by 2 for whole numbers
- Divide by 2 for fractions

Let's convert a binary number to decimal by using the positional notation method. First, write out the number to be converted; then, write in the decimal equivalent for each position with a 1 indicated. Add these values to determine the decimal equivalent of the binary number. Look at our example in the chart format where we want to convert 101001_2 to decimal:

32	16	**8**	4	2	**1**
1	0	1	0	0	1_2

$32 + 8 + 1 = 41_{10}$

Let's look at a fractional example. Convert 10110.01_2 to decimal. You may want to write the decimal equivalent for each position as we did in the following example.

16	8	**4**	**2**	1	.	0.5	**0.25**
1	0	1	1	0	.	0	1_2

$16 + 4 + 2 + 0.25 = 22.25_{10}$

Add only the values indicated by a 1.

You should make sure that the decimal values for each position are properly aligned before adding.

Octal to Decimal

Conversion of octal numbers to decimal is best done by the positional notation method. This process is the one we used to convert binary numbers to decimal.

First, determine the decimal equivalent for each position by multiplying 8 by itself the number of times indicated by the exponent. Set up a bar graph of the positions and values as shown below:

8^4	8^3	8^2	8^1	8^0	.	8^{-1}	8^{-2}	8^{-3}
4096	512	64	8	1	.	0.125	0.015625	0.0019531

To convert an octal number to decimal, write out the number to be converted, placing each digit under the proper position.

For example, convert 743_8 to base-10.

To solve this problem, we multiply the decimal equivalent by the corresponding digit of the octal number; then, add this column of figures for the final solution:

Positional Value	**64**	**8**	**1**
Number	7	4	3_8
Product	7*64 = 448	8*4 = 32	1*3 = 3
Solution	448 + 32 + 3 = 483_{10}		

Therefore, 743_8 is equal to 483_{10}.

Now follow the conversion of 26525_8 to decimal:

Positional Value	**4096**	**512**	**64**	**8**	**1**
Number	2	6	5	2	5
Product	4096*2 = 8192	512*6 = 3072	64*5 = 320	8*2 = 16	1*5 = 5
Solution	8192 + 3072 + 320 + 16 + 5 = 11605_{10}				

Therefore, 11605_{10} is the decimal equivalent of 26525_8

To convert a fraction or a mixed number, simply use the same procedure.

Example: Change $.5_8$ to decimal:

Positional Value	**8**	**1**	**.**	**0.125**	**0.015625**
Number	-	-	.	5	-
Product			.	.125*5=0.625	
Solution	$.625_{10}$				

Therefore, $.5_8$ is $.625_{10}$.

For another example, convert 24.36_8 to decimal:

Positional Value	**8**	**1**	**.**	**0.125**	**0.015625**
Number	2	4	.	3	6
Product	8*2 = =16	1*4 = 4	.	0.125*3 = 0.375	0.015625*6 = 0.09375
Solution	$16 + 4 + 0.375 + 0.09375 = 20.46875_{10}$				

Therefore, 24.36_8 equals 20.46875_{10}.

If your prefer or find it easier, you may want to convert the octal number to binary and then to decimal.

Hex to Decimal

It is difficult to comprehend the magnitude of a base-16 number until it is presented in base-10; for instance, $E0_{16}$ is equal to 224_{10}. You must remember that usually fewer digits are necessary to represent a decimal value in base-16.

When you convert from base-16 to decimal, you may use the positional notation system for the powers of 16 (a bar graph). You can also convert the base-16 number to binary and then convert to base-10.

Note in the bar graph below that each power of 16 results in a tremendous increase in the decimal equivalent. Only one negative power (16^{-1}) is shown for demonstration purposes:

16^4	16^3	16^2	16^1	16^0	.	16^{-1}
65,536	4,096	256	16	1	.	0.0625

Just as we did with octal conversions, write out the hex number, placing each digit under the appropriate decimal value for that position. Multiply the decimal value by the base-16 digit and add the values. (Convert A through F to their decimal equivalent before multiplying). Let's take a look at an example.

Convert $2C_{16}$ to decimal. Use the same procedure we used with binary and octal to convert base-16 fractions to decimal. If you choose to convert the hex number to binary and then to decimal, the solution will look like this:

Positional Value	**16**	**1**
Number	2	C
Product	16*2 = 32	C(12) * 1 = 12
Solution	$32 + 12 = 44_{10}$	

The decimal equivalent of $2C_{16}$ is 44_{10}.

Chapter 6
Binary-Coded Decimal

In today's technology, you hear a great deal about microprocessors. A microprocessor is an integrated circuit designed for two purposes: data processing and control.

Computers and microprocessors both operate on a series of electrical pulses called words. A word can be represented by a binary number such as 10110011_2. The word length is described by the number of digits or *bits* in the series. A series of four digits would be called a 4-bit word and so forth. The most common are 4-, 8-, and 16-bit words. Quite often, these words must use binary-coded decimal inputs.

Binary-coded decimal, or BCD, is a method of using binary digits to represent the decimal digits 0 through 9. A decimal digit is represented by four binary digits, as shown below:

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

You should note in the table above that the BCD coding is the binary equivalent of the decimal digit.

Since many devices use BCD, knowing how to handle this system is important. *You must realize that BCD and binary are not the same*. For example, 49_{10} in binary is 110001_2, but 49_{10} in BCD is 01001001_{BCD}. Each decimal digit is converted to its binary equivalent.

BCD Conversion

You can see by the above table, conversion of decimal to BCD or BCD to decimal is similar to the conversion of hexadecimal to binary and vice versa.

For example, let's go through the conversion of 264_{10} to BCD. We'll use the block format that you used in earlier conversions. First, write out the decimal number to be converted; then, below each digit write the BCD equivalent of that digit:

2	6	4_{10}
0010	0110	0100_{BCD}

The BCD equivalent of 264_{10} is 001001100100_{BCD}. To convert from BCD to decimal, simply reverse the process as shown:

1001	1000	0011_{BCD}
9	8	3_{10}

BCD Addition

The procedures followed in adding BCD are the same as those used in binary. There is, however, the possibility that addition of BCD values will result in invalid totals. The following example shows this:

Add 9 and 6 in BCD:

$$\begin{array}{rr} 9_{10} & 1001_{BCD} \\ \underline{+6_{10}} & \underline{+0110_{BCD}} \\ 15_{10} & 1111 \end{array}$$

The sum 1111_2 is the binary equivalent of 15_{10}; however, 1111 is not a valid BCD number. You cannot exceed 1001 in BCD, so a correction factor must be made. To do this, you add 6_{10} (0110_{BCD}) to the sum of the two numbers. The *"add 6" correction factor* is added to any BCD group larger than 1001_2.

Remember, there is no 1010_2, 1011_2, 1100_2, 1101_2, 1110_2, or 1111_2 in BCD:

$$\begin{array}{r} 1111 \\ \underline{+0110_{BCD}} \\ 0001\ 0101 \end{array}$$ - ("add 6" correction factor)

The sum plus the "add 6" correction factor can then be converted back to decimal to check the answer. Put any carries that were developed in the "add 6" process into a new 4-bit word:

0001	0101_{BCD}
1	5_{10}

Now observe the addition of 60_{10} and 55_{10} in BCD:

$$
\begin{array}{rl}
60_{10} = & 0110\ 0000_{BCD} \\
+55_{10} = & \underline{+0101\ 0101_{BCD}} \\
& 1011\ 0101
\end{array}
$$

In this case, the higher order group is invalid, but the lower order group is valid. Therefore, the correction factor is added only to the higher order group as shown:

$$
\begin{array}{r}
1011\ 0101 \\
\underline{+\ 0110\ 0000} \\
0001\ 0001\ 0101_{BCD}
\end{array}
$$

Convert this total to decimal to check your answer:

0001	0001	0101_{BCD}
1	1	5_{10}

Remember that the correction factor is added only to groups that exceed 9_{10} (1001_{BCD}).

Summary

This first course in digital logic circuits is intended to lay the foundation to understand the mathematics of logic gates. This should give you a basic understanding of number systems. The number systems that were dealt with are used extensively in the microprocessor and computer fields. The following is a summary of the emphasized terms and points we have reviewed,

The *unit* represents a single object.

A *number* is a symbol used to represent one or more units.

The *radix* is the base of a positional number system. It is equal to the number of symbols used in that number system.

A *positional notation* is a system in which the value or magnitude of a number is defined not only by its digits or symbol value, but also by its position. Each position represents a power of the radix, or base, and is ranked in ascending or descending order.

The *most significant digit (MSD)* is a digit within a number (whole or fractional) that has the largest effect (weighing power) on that number.

The *least significant digit (LSD)* is a digit within a number (whole or fractional) that has the least effect (weighting power) on that number.

The *binary number system* is a base-2 system. The symbols 1 and 0 can be used to represent the state of electrical/electronic devices. A binary "1" may indicate the device is active; a "0" may indicate the device is inactive.

The *octal number system* is a base-8 system and is quite useful as a tool in the conversion of binary numbers. This system works because 8 is an integral power of 2; that is, $2^3 = 8$. The use of octal numbers reduces the number of digits required to represent the binary equivalent of a decimal number.

The *hex number system* is a base-16 system and is sometimes used in computer systems. A binary number can be converted directly to a base-16 number if the binary number is first broken into groups of four digits.

The basic rules of addition apply to each of the number systems. Each system becomes unique when carries are produced.

Subtraction in each system is based on certain rules of that number system. The *borrow* varies in magnitude according to the number system in use. In most computers, subtraction is accomplished by using the complement (R's or R's-1) of the subtrahend and adding it to the minuend.

To convert a whole base-10 number to another system, divide the decimal number by the base of the number system to which you are converting. Continue dividing the quotient of the previous division until it can no longer be done. Extract the remainders - the remainder from the first computation will yield the LSD; the last will provide the MSD.

To convert decimal fractions, multiply the fraction by the base of the desired number system. Extract those digits that move to the left of the radix point. Continue to multiply the fractional product for as many places as needed. The first digit left of the radix point will be the MSD, and the last will be the LSD.

Binary numbers are converted to octal and hex by the grouping method. Three binary digits equal one octal digit; four binary digits equal one hex digit.

To convert binary, octal, and hex numbers to decimal, use the powers of the base being converted.

Binary-coded decimal (BCD) is a coding system used with some microprocessors. A correction factor is needed to correct invalid numbers.

With the information gleaned from this course, we are now ready to delve into the actual digital logic circuits. The next course in this series covers the basic "gates" used in digital logic circuits and how they can be combined to give different outputs.

+++