What is Computer?

1. A computer is a programmable machine that receives input, stores and manipulates data, and provides output in a useful format.

2. Computer is a machine for performing calculations automatically

3. Computer is an electronic machine capable of performing calculations and other manipulations of various types of data, under the control of a stored set of instructions. The machine itself is the hardware; the instructions are the program or software.

What is Computer Hardware?

Computer hardware refers to the physical parts of a computer and related devices. Internal hardware devices include motherboards, hard drives, and RAM. External hardware devices include monitors, keyboards, mouse, printers, and scanners.

What is Software?

A computer cannot do anything on its own. It must be instructed to do a desired job. Hence, it is necessary to specify a sequence of instruction, which a computer must perform to solve a problem. Such a sequence of instructions, written in a language, which can be understood by a computer, is called a computer program.

The term software refers to the set of computer programs, procedures, and associated, documents (flowcharts, manuals, etc.) which describe the programs, and how they are to be used.

Difference of Hardware and Software:

Hardware	Software
Hardware refers to the physical components of the computer that run the software.	Software refers to things that are used by the hardware, such as programs that you install on your computer including games, word processing programs, spreadsheet programs, graphic design programs and the like.
Hardware includes the physical components, such as the motherboard, chips, memory, and hard drives,	Software includes the programs that run on the hardware.
Hardware starts functioning once software is loaded.	To deliver its set of instructions, Software is installed on hardware.
Hardware is a physical device something that you're able to touch and see. For example, the computer monitor you're viewing this text on or the mouse you're using to navigate is considered computer hardware.	Software is code and instructions that tell a computer and/or hardware how to operate. This code can be viewed and executed using a computer or other hardware device.

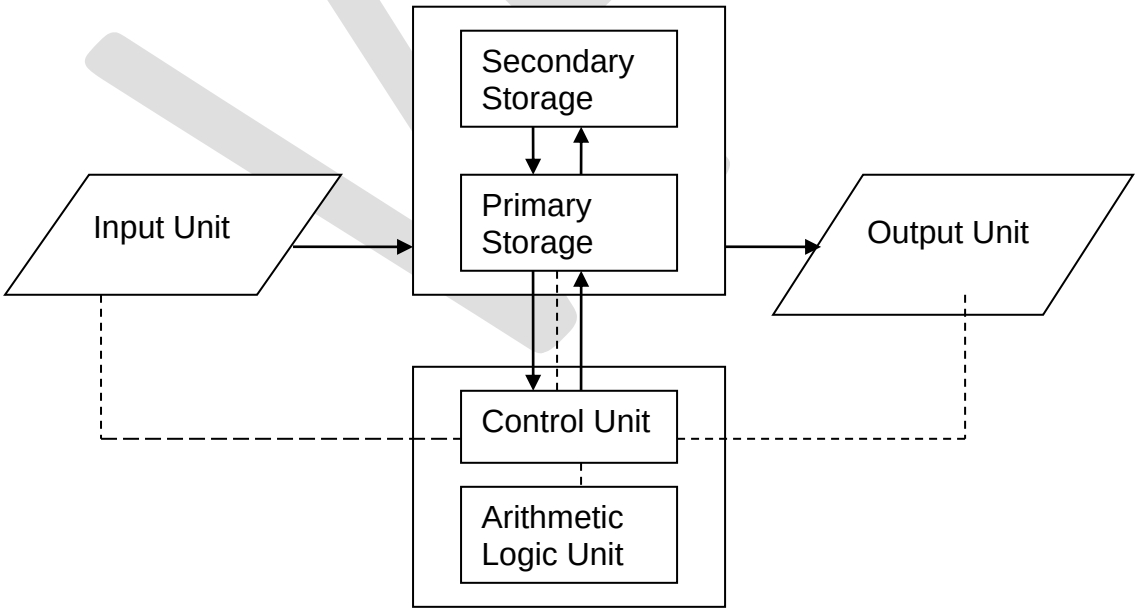
Hardware includes every computer-related object that you can physically touch and handle like disks, screens, keyboards, printers, chips, wires, central processing unit, floppies, USB ports, pen drives etc.	Software includes every computer-related program that you cannot feel with the physical senses for example, operating system, an anti-virus program, the web browser etc.

Basic Computer Organization

All computer systems perform the following five basic operations, for converting raw input data into information, which is useful to their users:

1. **Inputting:** The process of entering data and instructions into the computer system.
2. **Storing:** Saving data instructions to make them readily available for initial or additional processing, as and when required.
3. **Processing: P**erforming arithmetic operations (add, subtract, divide, etc.), or logical operations (comparisons like equal to, less than, grater than, etc.) on data, to convert them into useful information.
4. **Outputting:** The process of producing useful information or results for the user, such as a printed report or visual display.
5. **Controlling:** Directing the manner and sequence in which all of the above operations are performed.

A block diagram of the basic computer organization is as shown below figure. In this figure, the solid lines indicate the flow of instruction and data, and the dotted lines represent the control exercised by the control unit.



Central Processing Unit (CPU)
Basic Organization of a Computer System

It displays the five major building blocks (functional units) of a digital computer system. These five units correspond to the basic operations, performed by all computer systems.

INPUT UNIT:

Data and instructions must enter the computer system, before any computation can be performed on the supplied data. This task is performed by the input unit, which links the external environment with the computer system. Data and instructions enter input units in forms, which depend upon the particular device used. Input interfaces are designed to match the unique physical or electrical characteristics of input devices, to the requirements of the computer system.

The following functions are performed by an input unit:

1. It accept (or reads) the instructions and data from the outside world.
2. It converts these instruction and data in computer acceptable form.
3. It supplies the converted instructions and data to the computer system for further processing.

Some examples of input devices are Keyboard, mouse, Optical character Recognition (OCR), Bar Code Reader, Digitizer etc.

OUTPUT UNIT:

The job an output unit is just the reverse of that of an input unit. It supplies the information obtained from data processing, to the outside world. Hence it links the computer with the external environment. As computer work with binary code, the results produced are also in the binary form. Hence before supplying results to the outside world, they must be converted to human acceptable form. This task is accomplished by units called output interfaces. Output interfaces are designed to match the unique physical or electrical characteristics of output devices to the requirements of the external environment.

The following function is performed by an output unit:

1. It accepts the results produced by the computer, which are in coded from, and hence, cannot be easily understood by us.
2. It converts these coded results to human acceptable (readable) form.
3. It supplies the converted results to the outside world.

Some examples of output devices are Monitors, Printers, Plotters etc.

STORAGE UNIT:

The data and instructions, which are entered into the computer system through input units, have to be stored inside the computer, before the actual processing starts. Similarly, the results produced by the computer after processing, must also be kept somewhere inside the computer system, before being passed on to the output units.

The specific function of the storage unit is to hold (store):

1. The data and instructions required for processing (received from input devices).
2. Intermediate results of processing.
3. Final results of processing, before these results are released to an output device.

The storage unit of all computers is comprised of the following two types of storage:

1. Primary storage:

The primary storage, also known as main memory, is used to hold pieces of program instructions and data, processing, of the job(s), which the computer system is currently working on. These pieces of information are represented electronically in the main memory chip's circuitry, and while it remains in the main memory, the central processing unit can access it directly at a very fast speed. However, the primary storage can hold information only while the computer system is on. As soon as the computer system is switched off or reset, the information held in the primary storage disappears. Moreover, the primary storage normally has limited storage capacity, because it is very expensive. The primary storage of modern computer system is made up of semiconductor devices. Examples are RAM, Masked ROM, PROM, EPROM, EEPROM

Primary storage of a computer system has the following limitations:

Limited capacity:
The storage capacity of the primary storage of computer is not sufficient to store the large volume of data, which needs to be handled by most data processing centers.

Volatile:
The primary storage is volatile, and the data stored in it is lost, when the electric power is turned off or interrupted.

2. Secondary storage:

The secondary storage, also known as auxiliary storage, is used to take care of the limitations of the primary storage. That is, it is used to supplement the limited storage capacity and the volatile characteristic of primary storage. This is because secondary storage is much cheaper than primary storage, and it can retain information even when the computer system is switched off or reset. The secondary storage is normally used to hold the program instructions, data, and information of those jobs, on which the computer system is not working on currently, but needs to hold them for processing later. The most commonly used secondary storage medium is the magnetic disk.

ARITHMETIC LOGIC UNIT:

The arithmetic logic unit (ALU) of a computer system is the place, where the actual execution of the instructions takes place, during the processing operation. To be more precise, calculations are performed, and all comparisons (decisions) are made in the ALU. The data and instructions, stored in the primary storage before processing, are transferred as and when needed to the ALU, where processing takes place. No processing is done in the primary storage unit. Intermediate results generated in the ALU are temporarily transferred back to the primary storage, until needed later. Hence, data may move from primary storage to ALU, and back again to storage, many times, before the processing is over.

The type and number of arithmetic and logic operations, which a computer can perform, is determined by the engineering design of the ALU. Almost all ALUs are designed to perform the four basic arithmetic operations (add, subtract, multiply and divide), and logic operations or comparisons, such as less than, equal to, and greater than.

CONTROL UNIT

How does the input device know that it is time for to feed data into the storage unit? How does the ALU know, what should be done with the data once they are rec-eived? Moreover, how is it that only the final results are sent to the output device, and not the intermediate result? All this is possible due to the control unit acts as a central nervous system, for the other components of the computer system.

CENTRAL PROCESSING UNIT

The control unit and the arithmetic logic unit of a computer system are jointly known as the Central Processing Unit (CPU). The CPU is the brain of the computer system. In a human body, all major decisions are taken by the brain and the other parts of the body function as directed by the brain. Similarly, in a computer system, all major calculation and comparisons are made inside the CPU, and the CPU is responsible for activating and controlling the operation of other units of the computer system.

THE EVOLUTION OF COMPUTERS

The Mark I Computer (1937-44): Also known as Automatic Sequence Controlled Calculator, this was the first fully automatic calculating machine designed by Howard A. Aiken of Harvard University, in collaboration with IBM (International Business Machines) Corporation. Its design was based on the techniques already developed for punched card machinery. It was an electro-mechanical device, since both mechanical and electronic components were used in its design.

Although this machine proved to be extremely reliable, it was very complex in design and huge in size.

The Atanasoff-Berry Computer (1939-42): This electronic machine was developed by Dr. John Atanasoff to solve certain mathematical equations. It was called the Atanasoff-Berry Computer, or ABC, after its inventor's name and his assistant, Clifford Berry. It used 45 vacuum tubes for internal logic and capacitors for storage.

The ENIAC (1943-46): The Electronic Numerical Integrator and Calculator (ENIAC) was the first all electronic computer. It was constructed at the Moore School of Engineering of the University of Pennsylvania, U.S.A. by a design team led by Professors J. Presper Eckedrt and John Mauchly. ENIAC was developed because of military need, and was used for many years to solve ballistic problems.

The EDVAC (1946-52): A major drawback of ENIAC was that its programs were wired on boards, which made it difficult to change the programs. This problem was later overcome by the "stored program" concept introduced by Dr. John Von Neumann. The basic idea behind this concept is that a sequence of instructions, as well as data, can be stored in the memory of the computer, for automatically directing the flow of operations. The Electronic Discrete Variable Automatic Computer (EDVAC) was designed on stored program concept.

The EDSAC (1947-49): Almost simultaneously with EDVAC of U.S.A., the Britishers developed the Electronic Delay Storage Automatic Calculator (EDSAC). The machine executed its first program in May 1949. In this machine, addition operation was accomplished in 1500 microseconds, and multiplication operation in 4000 microseconds.

The UNIVAC I (1951): The Universal Automatic Computer (UNIVAC) was the first digital computer, which was not "one of a king". Many UNIVAC machines were produced, the first of which was installed in the Census Bureau in 1951 and was used continuously for 10 years. The first business use of a computer, a UNIVAC I, was by General Electric Corporation in 1954.

Applications of the computer systems:

- **Businesses:** Businessmen make bar graphs and pie charts from tedious figures to convey information with far more impact than numbers alone can covey. Furthermore, computers help businesses to predict their future sales, profits, costs etc. making companies more accurate in their accounts. Computers may also play a vital role in aiding thousands of organizations to make judgmental and hard-provoking decisions concerning financial problems and prospective trends.
- **Buildings:** Architects use computer animated graphics to experiment with possible exteriors and to give clients a visual walk-through of their proposed buildings.
- **Education:** Most good schools in the world have computers available for use in the classroom. It has been proved that learning with computers has been more successful and this is why numerous forms of new teaching methods have been introduced. This enhances the knowledge of the student at a much faster pace than the old traditional methods.
- **Retailing:** Products from meats to magazines are packed with zebra-striped bar codes that can be read by the computer scanners at supermarket checkout stands to determine prices and help manage inventory. Thus, a detailed receipt of the groceries can be made, which is useful for both the customer and the retail store, especially for the stock control system.
- **Energy:** Energy companies use computers to locate oil, coal, natural gas and uranium. With the use of these technological machines, these companies can figure out the site of a natural resource, its concentration and other related figures.
- **Transportation:** Computers are used in cars to monitor fluid levels, temperatures and electrical systems. Computers are also used to help run rapid transit systems, load containerships and track railroads cars across the country.
- **Money:** Computers speed up record keeping and allow banks to offer same-day services and even do-it yourself banking over the phone and internet. Computers have helped fuel the cashless economy, enabling the widespread use of credit cards, debit cards and instantaneous credit checks by banks and retailers.
- **Agriculture:** Farmers use small computers to help with billing, crop information, and cost per acre, feed combinations, and market price checks. Cattle ranchers can also use computers for information about livestock breeding and performance.
- **Health and Medicine:** Computers are helping immensely to monitor thee extremely ill in the intensive care unit and provide cross-sectional views of the body. Doctors use computers to assist them in diagnosing certain diseases of the sort. This type of computer is called the Expert System, which is basically a collection of accumulated expertise in a specific area of field.
- **Manufacturing Industries:** Computers have made their way towards jobs that were unpleasant or too dangerous for humans to do, such as working hundreds of feet below the earth or opening a package that might contain an explosive device. In other industries, computers are used to control the production of resources very precisely. All robots and machinery are now controlled by various computers, making the production process faster and cheaper. All the stages of manufacturing, from designing to production, can be done with the use of computer technology with greater diversity.
- **Scientific Research:** This is very important for mankind and with the development of computers; scientific research has propelled towards the better a great deal. Because of

high-speed characteristics of computer systems, researchers can simulate environments, emulate physical characteristics and allow scientists to proof of their theories in a cost-effective manner.

- **Communication with the World:** The computes are most popular for their uses to connect with others on the World Wide Web. Therefore, communication between two or more parties is possible which is relatively cheap considering the old fashioned methods. Emailing, teleconferencing and the use of voice messages are very fast, effective and surprisingly cheaper as well. When connected to the Internet, people can gain various amounts of knowledge, and know about world events as they occur. Purchasing on the Internet is also becoming very popular, and has numerous advantages over the traditional shopping methods.
- **Real Time systems:** Many computers provide an environment, which is completely based on real time. This means processing of one entity is done so quickly and effectively, that another entity is not effected. For example Airline systems and Banking systems will come under this category. These systems are immensely huge because they interact with all other airlines or baking systems in the world. A computer system, therefore, becomes more than just necessary in daily uses.

There are so many applications of computers, that it is impractical to mention all of them. This is the Computer Age and these machines are beginning to affect out lives in many ways.

Number System

(1) Non-Positional number system

Number systems are of two types – non positional and positional. In early days, human beings counted on fingers, when counting beyond ten fingers, they used stones, pebbles, or stick to indicate values. This method of counting uses an additive approach or non positional number system. Each symbol represents the same value regardless of its position in a number, and to find the value of a number, one has to count the number of symbols present in the number. Since it is very difficult to perform arithmetic with such a number system, positional number system were developed.

(2) Positional Number System.

There are only few symbols called digits. These symbols represent different values, depending on the position they occupy in a number. The value of each digit in such a number is determined by this consideration.

1. The digit itself.
2. The position of the digit in the number, and
3. The base of the number system (where base is defined as the total number of digits available in the number system)

In our day to day life, we use decimal number system. in this system, base is equal to 10 because there are altogether ten symbols or digits (0,1,2,3,4,5,6,7,8 and 9) . you know that in decimal number system, successive position to the left of the decimal point represent units, tens, hundreds, thousands, etc.

Hence we can represent any number by using the available digits and arranging them in various positions. The principal that apply to any other positional number system. It is important to keep track of only the base of the number system in which we are working.

The value of the base in all positional number system suggests the following characteristics.

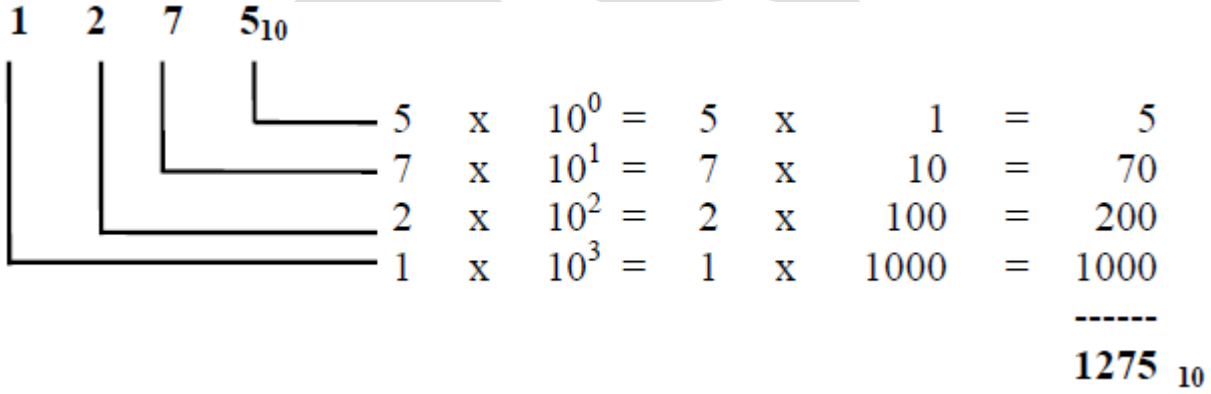
1. The value of the base determines the total number of different symbols or digits available in the number system. The first of these choices is always zero.
2. The maximum value of a single digit is always equal to one less than the value of base.

The study of number systems is useful to the student of computing due to the fact that number systems other than the familiar **decimal (base 10) number system** are used in the computer field.

Digital computers internally use the **binary (base 2) number system** to represent data and perform arithmetic calculations. The binary number system is very efficient for computers, but not for humans. Representing even relatively small numbers with the binary system requires working with long strings of ones and zeroes.

The **hexadecimal (base 16) number system** (often called "hex" for short) provides us with a shorthand method of working with binary numbers. One digit in hex corresponds to four binary digits (bits), so the internal representation of one byte can be represented either by eight binary digits or two hexadecimal digits. Less commonly used is the **octal (base 8)** number system, where one digit in octal corresponds to three binary digits (bits).

In the event that a computer user (programmer, operator, end user, etc.) needs to examine a display of the internal representation of computer data (such a display is called a "dump"), viewing the data in a "shorthand" representation (such as hex or octal) is less tedious than viewing the data in binary representation. The binary, hexadecimal , and octal number systems will be looked at in the following pages.
The decimal number system that we are all familiar with is a positional number system. The actual number of symbols used in a positional number system depends on its base (also called the radix). The highest numerical symbol always has a value of one less than the base. The decimal number system has a base of 10, so the numeral with the highest value is 9; the octal number system has a base of 8, so the numeral with the highest value is 7, the binary number system has a base of 2, so the numeral with the higest value is 1, etc.

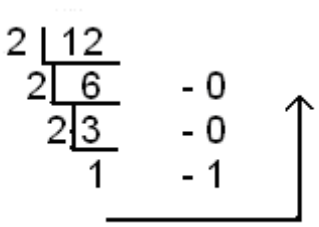


Any number can be represented by arranging symbols in specific positions. You know that in the decimal number system, the successive positions to the left of the decimal point represent units (ones), tens, hundreds, thousands, etc. Put another way, each position represents a specific power of base 10. For example, the decimal number 1,275 (written 1,27510)* can be expanded as above.

Conversions of Decimal to Binary-

The method that is used for converting of decimals into binary is known as the remainder method. We use the following steps in getting the binary number-
(a) Divide the decimal number by 2.
(b) Write the remainder (which is either 0 or 1) at the right most position.
(c) Repeat the process of dividing by 2 until the quotient is 0 and keep writing the remainder after each step of division.
(d) Write the remainders in reverse order.

Example- Convert $(12)_{10}$ into binary number system.



Conversions of Decimal Fractions to Binary Fractions-

For converting decimal fractions into binary fractions, we use multiplication. Instead of looking for a remainder
we look for an integer. The following steps are used in getting the binary fractions-

(a) Multiply the decimal fraction by 2.

(b) If a non-zero integer is generated, record the non-zero integer otherwise record 0.
(c) Remove the non-zero integer and repeat the above steps till the fraction value becomes 0.
(d) Write down the number according to the occurrence.

Example- Find the binary equivalent of $(0.75)_{10}$.

	Number (to be recorded)
$0.75 \times 2 = 1.50$	1
$0.50 \times 2 = 1.00$	1

Thus $(0.75)_{10} = (0.11)_2$.
Moreover, we can write $(45.75)_{10} = (101101.11)_2$.

Remark- If the conversion is not ended and still continuing; we write the approximation in 16 bits.

The Binary Number System

The same principles of positional number systems we applied to the decimal number system can be applied to the binary number system. However, the base of the binary number system is two, so each position of the binary number represents a successive power of two. From right to left, the successive positions of the binary number are weighted 1, 2, 4, 8, 16, 32, 64, etc. A list of the first several powers of 2 follows:

$2^0 = 1$	$2^1 = 2$	$2^2 = 4$	$2^3 = 8$	$2^4 = 16$	$2^5 = 32$
$2^6 = 64$	$2^7 = 128$	$2^8 = 256$	$2^9 = 512$	$2^{10} = 1024$	$2^{11} = 2048$

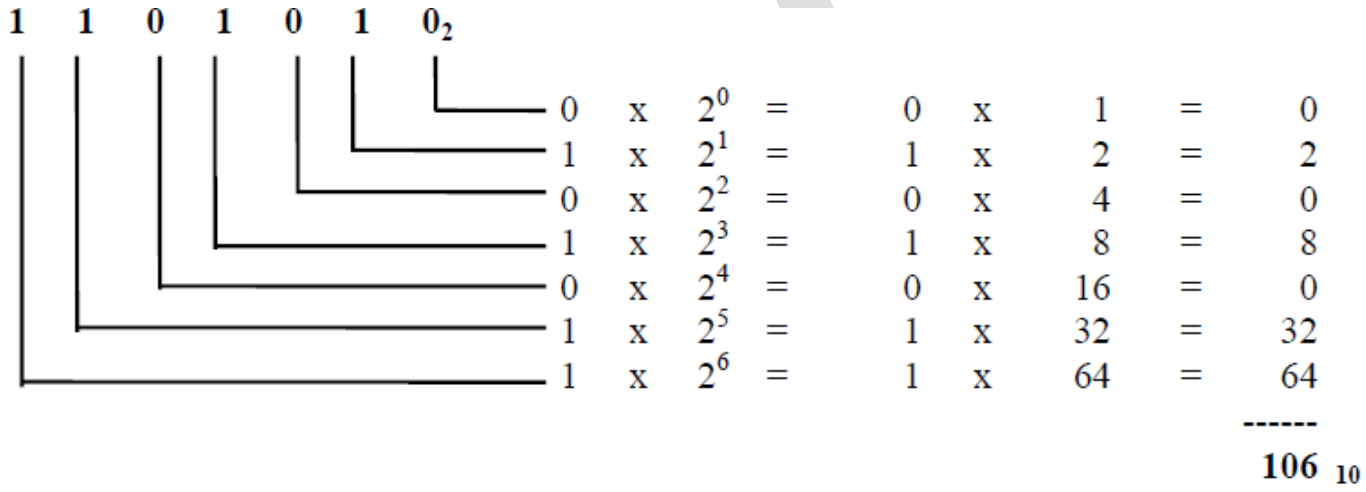
For reference, the following table shows the decimal numbers 0 through 31 with their binary equivalents:

Decimal	Binary	Decimal	Binary
0	**0**	**16**	**10000**
1	**1**	**17**	**10001**
2	**10**	**18**	**10010**
3	**11**	**19**	**10011**

4	**100**	**20**	**10100**
5	**101**	**21**	**10101**
6	**110**	**22**	**10110**
7	**111**	**23**	**10111**
8	**1000**	**24**	**11000**
9	**1001**	**25**	**11001**
10	**1010**	**26**	**11010**
11	**1011**	**27**	**11011**
12	**1100**	**28**	**11100**
13	**1101**	**29**	**11101**
14	**1110**	**30**	**11110**
15	**1111**	**31**	**11111**

Converting a Binary Number to a Decimal Number

To determine the value of a binary number (1101010, for example), we can expand the number using the positional weights as follows:



Conversions of Binary Fractions to Decimal Fractions

The conversions of binary fractions to the decimal fractions is similar to conversion of binary numbers to decimal numbers. Here, instead of a decimal point we have a binary point. The exponential expressions (or weight of the bits) of each fractional placeholder is 2-1, 2-2……

$$(1011.101)_2 = X_{10}$$
$$= 1x2^3 + 0x2^2 + 1x2^1 + 1x2^0 + 1x2^{-1} + 0x2^{-2} + 1x2^{-3}$$
$$= 8 + 0 + 2 + 1 + 0.5 + 0 + 0.125$$
$$= (11.625)_{10}$$

The Octal Number System

The same principles of positional number systems we applied to the decimal and binary number systems can be applied to the octal number system. However, the base of the octal number system is eight, so each position of the octal number represents a successive power of eight. From right to left, the successive positions of the octal number are weighted 1, 8, 64, 512, etc. A list of the first several powers of 8 follows:

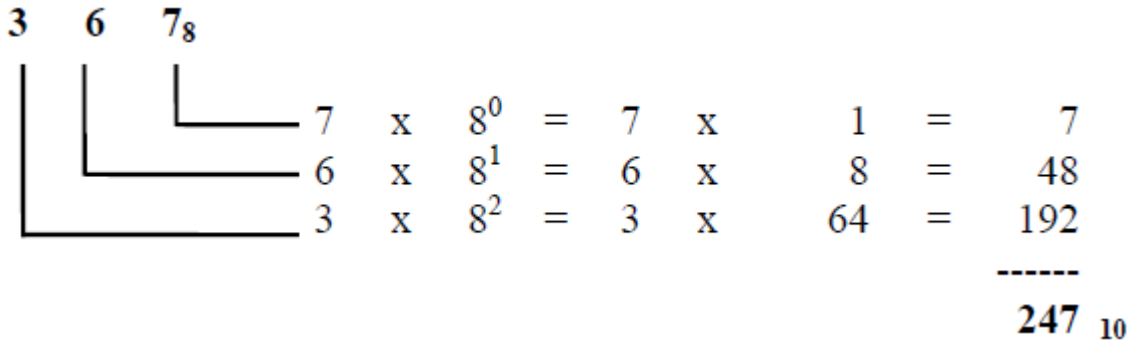
$8^0 = 1$ $8^1 = 8$ $8^2 = 64$ $8^3 = 512$ $8^4 = 4096$ $8^5 = 32768$

For reference, the following table shows the decimal numbers 0 through 31 with their octal equivalents:

Decimal	Octal	Decimal	Octal
0	0	16	20
1	1	17	21
2	2	18	22
3	3	19	23
4	4	20	24
5	5	21	25
6	6	22	26
7	7	23	27
8	10	24	30
9	11	25	31
10	12	26	32
11	13	27	33
12	14	28	34
13	15	29	35
14	16	30	36
15	17	31	37

Converting an Octal Number to a Decimal Number

To determine the value of an octal number (3678, for example), we can expand the number using the positional weights as follows:



Conversion an octal fraction number to a decimal number

$$(22.34)_8 = X_{10}$$

$$22.34 = 2 \times 8^1 + 2 \times 8^0 + 3 \times 8^{-1} + 4 \times 8^{-2}$$
$$= 16 + 2 + 3 \times 1/8 + 4 \times 1/64$$
$$= (18.4375)$$

$$(22.34)_8 = (18.4375)_{10}$$

Converting a Decimal Number to an Octal Number

To convert a decimal number to its octal equivalent, the remainder method (the same method used in converting a decimal number to its binary equivalent) can be used. To review, the remainder method involves the following four steps:

(1) Divide the decimal number by the base (in the case of octal, divide by 8).
(2) Indicate the remainder to the right.
(3) Continue dividing into each quotient (and indicating the remainder) until the divide operation produces a zero quotient.

The base 8 number is the numeric remainder reading from the last division to the first (if you start at the bottom, the answer will read from top to bottom)

Example 1: Convert the decimal number 465_{10} to its octal equivalent:

	0	**7**	(3)	Divide 8 into 7. The quotient is 0 with a remainder of 7, as indicated. Since the quotient is 0, stop here.
	8 ⌐ 7			
	7	**2**	(2)	Divide 8 into 58 (the quotient from the previous division). The quotient is 7 with a remainder of 2, indicated on the right.
	8 ⌐ 58			
	58	**1**	(1)	Divide 8 into 465. The quotient is 58 with a remainder of 1; indicate the 1 on the right.
START HERE ⇒	8 ⌐ 465			

Conversions of Decimal Fractions to Octal Fractions

We follow the same steps of conversions of decimal fractions to binary fractions. Here we multiply the fraction by 8 instead of 2.

$(19.11)_{10} = X_8$

$8 \lfloor 19$
$2 - 3$

$0.11 \times 8 = 0.88 \Rightarrow 0$
$0.88 \times 8 = 7.04 \Rightarrow 7$
$0.04 \times 8 = 0.32 \Rightarrow 0$
$0.32 \times 8 = 2.56 \Rightarrow 2$
$0.56 \times 8 = 4.48 \Rightarrow 4$

$(19.11)_{10} = (23.07024)_8$

The Hexadecimal Number System

The **hexadecimal (base 16) number system** is a positional number system as are the decimal number system and the binary number system. Recall that in any positional number system, regardless of the base, the highest numerical symbol always has a value of one less than the base. Furthermore, one and only one symbol must ever be used to represent a value in any position of the number.

For number systems with a base of 10 or less, a combination of Arabic numerals can be used to represent any value in that number system. The decimal number system uses the Arabic numerals 0 through 9; the binary number system uses the Arabic numerals 0 and 1; the octal number system uses

the Arabic numerals 0 through 7; and any other number system with a base less than 10 would use the Arabic numerals from 0 to one less than the base of that number system.

However, if the base of the number system is greater than 10, more than 10 symbols are needed to represent all of the possible positional values in that number system. The hexadecimal number system uses not only the Arabic numerals 0 through 9, but also uses the letters A, B, C, D, E, and F to represent the equivalent of $_{1010}$ through $_{1510,}$ respectively.

For reference, the following table shows the decimal numbers 0 through 31 with their hexadecimal equivalents:

Decimal	Hexadecima	Decimal	Hexadecimal
0	0	16	10
1	1	17	11
2	2	18	12
3	3	19	13
4	4	20	14
5	5	21	15
6	6	22	16
7	7	23	17
8	8	24	18
9	9	25	19
10	A	26	1A
11	B	27	1B
12	C	28	1C
13	D	29	1D
14	E	30	1E
15	F	31	1F

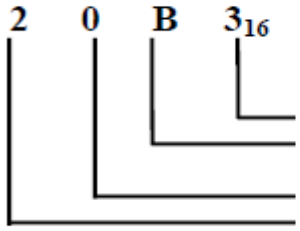
The same principles of positional number systems we applied to the decimal, binary, and octal number systems can be applied to the hexadecimal number system. However, the base of the hexadecimal number system is 16, so each position of the hexadecimal number represents a successive power of 16. From right to left, the successive positions of the hexadecimal number are weighted 1, 16, 256, 4096, 65536, etc.:

$16^0 = 1$ $16^1 = 16$ $16^2 = 256$ $16^3 = 4096$ $16^4 = 65536$

Converting a Hexadecimal Number to a Decimal Number

We can use the same method that we used to convert binary numbers and octal numbers to decimal numbers to convert a hexadecimal number to a decimal number, keeping in mind that we are now dealing with base 16. From right to left, we multiply each digit of the hexadecimal number by the value of 16 raised to successive powers, starting with the zero power, then sum the results of the multiplications. Remember that if one of the digits of the hexadecimal number happens to be a letter A through F, then the corresponding value of 10 through 15 must be used in the multiplication.

Example 1: Convert the hexadecimal number 20B316 to its decimal equivalent.



3	x	16^0	=	3	x	1	=	3
11	x	16^1	=	11	x	16	=	176
0	x	16^2	=	0	x	256	=	0
2	x	16^3	=	2	x	4096	=	8192

								8371 $_{10}$

Converting a fraction hexadecimal number to a decimal number

$$(81.21)_{16} = X_{10}$$
$$= 8 \times 16^1 + 1 \times 16^0 + 2 \times 16^{-1} + 1 \times 16^{-2}$$
$$= 8 \times 16 + 1 \times 1 + 2/16 + 1/16^2$$
$$= (129.1289)_{10}$$

$$(81.21)_{16} = (129.1289)_{10}$$

Converting a Decimal Number to a Hexadecimal Number

To convert a decimal number to its hexadecimal equivalent, the remainder method (the same method used in converting a decimal number to its binary equivalent) can be used. To review, the remainder method involves the following four steps:

(1) Divide the decimal number by the base (in the case of hexadecimal, divide by 16).
(2) Indicate the remainder to the right. If the remainder is between 10 and 15, indicate the corresponding hex digit A through F.
(3) Continue dividing into each quotient (and indicating the remainder) until the divide operation produces a zero quotient.
(4) The base 16 number is the numeric remainder reading from the last division to the first (if you start at the bottom, the answer will read from top to bottom).

Example 1: Convert 9263_{10} to its hexadecimal equivalent:

0				
16 | 2	**2**	(4)	Divide 16 into 2. The quotient is 0 with a remainder of 2, as indicated. Since the quotient is 0, stop here.
2				
16 | 36	**4**	(3)	Divide 16 into 36. The quotient is 2 with a remainder of 4, indicated on the right.
36				
16 | 578	**2**	(2)	Divide 16 into 578 (the quotient from the previous division). The quotient is 36 with a remainder of 2, indicated on the right.
578				
START HERE ⇒ 16 | 9263	**F**	(1)	Divide 16 into 9263. The quotient is 578 with a remainder of 15; so indicate the hex equivalent, "F", on the right.

The answer, reading the remainders from top to bottom, is **242F**, so **$9263_{10} = 242F_{16}$**.

Converting Decimal fraction number to Hexadecimal Number

$(2479.859))_{10} = X_{16}$

16 | 2479
16 | 154 - 15(F)
9 - 10(A)

16 x 0.859 = 13.744 => 13 (D)
16 x 0.744 = 11.904 => 11 (B)
16 x 0.904 = 14.464 => 14 (E)
16 x 0.464 = 7.424 => 7
16 x 0.424 = 6.784 => 6

$(2479.859)_{10} = (9AF.DBE76)_{16}$

Converting Binary-to-Hexadecimal or Hexadecimal-to-Binary

Converting a binary number to its hexadecimal equivalent or vice-versa is a simple matter. Four binary digits are equivalent to one hexadecimal digit, as shown in the table below:

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

To convert from binary to hexadecimal, divide the binary number into groups of 4 digits **starting on the right of the binary number.** If the leftmost group has less than 4 bits, put in the necessary number of leading zeroes on the left. For each group of four bits, write the corresponding single hex digit.

Example 1: $1101001101110111_2 = ?_{16}$

Answer:

Bin:	1101	0011	0111	0111
Hex:	D	3	7	7

Example 2: $101101111_2 = ?_{16}$

Answer:

Bin:	0001	0110	1111
Hex:	1	6	F

To convert from hexadecimal to binary, write the corresponding group of four binary digits for each hex digit.

Example 1: $1BE9_{16} = ?_2$

Answer:

Hex:	1	B	E	9
Bin:	0001	1011	1110	1001

Example 2: $B0A_{16} = ?_2$

Answer:

Hex:	B	0	A
Bin:	1011	0000	1010

Converting fraction Binary-to-Hexadecimal and Hexadecimal to Binary

$(100101110\ .\ 11011)_2 = X_{16}$

0001 0010 1110 . 1101 1000
1 2 E . D 8

$(100101110\ .\ 11011)_2 = (12E\ .\ D8)_{16}$

$(5D.\ 2A)_{16} = X_2$

5 D . 2 A
0101 1101 . 0010 1010

$(5D.\ 2A)_{16} = (01011101.00101010)_2$

Converting Binary-to-Octal or Octal-to-Binary

Converting a binary number to its octal equivalent or vice-versa is a simple matter. Three binary digits are equivalent to one octal digit, as shown in the table below:

Binary	Octal
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

To convert from binary to octal, divide the binary number into groups of 3 digits **starting on the right of the binary number.** If the leftmost group has less than 3 bits, put in the necessary number of leading zeroes on the left. For each group of three bits, write the corresponding single octal digit.

Example 1: $1101\ 001101110111_2 = ?_8$
Answer:

Bin:	**001**	**101**	**001**	**101**	**110**	**111**
Oct:	**1**	**5**	**1**	**5**	**6**	**7**

Example 2: $101101111_2 = ?_8$
Answer:

Bin:	**101**	**101**	**111**
Oct:	**5**	**5**	**7**

To convert from octal to binary, write the corresponding group of three binary digits for each octal digit.

Example 1: $1764_8 = ?_2$
Answer:

Oct:	**1**	**7**	**6**	**4**
Bin:	**001**	**111**	**110**	**100**

Example 2: $731_8 = ?_2$
Answer:

Oct:	**7**	**3**	**1**
Bin:	**111**	**011**	**001**

Converting fraction of a Binary to Octal and Octal to Binary

$(1101101.11101)_2 = X_8$

001 101 101.111 010
1 5 5 . 7 2

$(1101101.11101)_2 = (155.72)_8$

$(57.127)_8 = X_2$

5 7 . 1 2 7
101 111 . 001 010 111

$(57.127)_8 = (101111001010111)_2$

Conversions of Hexadecimal to Octal

We convert each hexadecimal digit in binary. Combine all the binary numbers. Again group them into 3-bit form. Convert the 3-bit block in octal.

Example- Convert (2D)16 into octal.

Hexadecimal Number	2	D(=13)
Binary Number	0010	1101

Combining the binary number, we get 00101101=101101
Grouping the binary number in 3-bit

Binary Number	101	101
Octal Number	5	5

Thus (2D)16 = (55)8.

Conversions of Hexadecimal Fractions to Octal Fractions

We follow the same steps of hexadecimal to octal conversion.

Hexadecimal Number	2	D(=13)	C(=12)
Binary Number	0010	1101	1100

Combining the binary number, we get 00101101.1100=101101.11
Grouping the binary number in 3-bit

Binary Number	101	101	110
Octal Number	5	5	6

Thus (2D.C)16 = (55.6)8.

Conversions of Octal to Hexadecimal The conversion involves the following steps-
(a) Convert each octal digit to 3 –bit binary form.
(b) Combine all the 3-bit binary numbers.
(c) Group them in 4-bit binary form by starting from MSB to LSB.
(d) Convert these 4-bit blocks into their hexadecimal symbols.

Example- Convert $(55)_8$ into hexadecimal.

Octal Number	5	5
Binary Number	101	101

Combining the 3-bit binary block, we have 101101.
Grouping them in 4 bit binary form-

Binary Number	0010	1101
Hexadecimal Symbol	2	D

Thus $(55)_8 = (2D)_{16}$.

Converting fraction of a Octal –to –Hexadecimal and Hexadecimal –to-Octal Number

Example- Convert $(55.6)_8$ into hexadecimal.

Octal Number	5	5	6
Binary Number	101	101	110

Combining the 3-bit binary block, we have 101101.110.
Grouping them in 4 bit binary form-

Binary Number	0010	1101	1100
Hexadecimal Symbol	2	D	C

Thus $(55)_8 = (2D.C)_{16}$.

Binary Addition

Adding two binary numbers together is easy, keeping in mind the following four addition rules.

(1)	**0**	+	**0**	=	**0**
(2)	**0**	+	**1**	=	**1**
(3)	**1**	+	**0**	=	**1**
(4)	**1**	+	**1**	=	**10**

Note in the last example that it was necessary to "carry the 1". After the first two binary counting numbers, 0 and 1, all of the binary digits are used up. In the decimal system, we used up all the digits after the tenth counting number, 9. The same method is used in both systems to come up with the next number: place a zero in the "ones" position and start over again with one in the next position on the left. In the decimal system, this gives ten, or 10. In binary, it gives 102, which is read "one-zero, base two."

```
  1 1 0      1 1      1 0 0      1 1      1 0 1 0      1 1 1 1 1
+ 0 0 1    + 1 0    + 1 0 1    + 0 1    + 0 1 1 1    + 0 1 0 1 1
---------  -------  ---------  -------  -----------  -------------
  1 1 1    1 0 1    1 0 0 1    1 0 0    1 0 0 0 1    1 0 1 0 1 0
```

Binary Subtraction

We will use the complement method to perform subtraction in binary and in the sections on octal and hexadecimal that follow. As mentioned in the previous section, the use of complemented binary numbers makes it possible for the computer to add or subtract numbers using only circuitry for addition - the computer performs the subtraction of A - B by adding A + (two's complement of B) and then dropping the carried 1.

The steps for subtracting two binary numbers are as follows:

(1) Compute the one's complement of the subtrahend by subtracting each digit of the subtrahend by 1. A shortcut for doing this is to simply reverse each digit of the subtrahend - the 1's become 0's and the 0's become 1's.

(2) Add 1 to the one's complement of the subtrahend to get the two's complement of the subtrahend.

Example 1: Compute 11010101_2 - 1001011_2

(1) Compute the one's complement of 1001011_2 by subtracting each digit from 1 (note that a leading zero was added to the 7-digit subtrahend to make it the same size as the 8-digit minuend):

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

(Note that the one's complement of the subtrahend causes each of the original digits to be reversed.)

(2) Add 1 to the one's complement of the subtrahend, giving the two's complement of the subtrahend:

1	0	1	1	0	1	0	0
						+	1
1	0	1	1	0	1	0	1

(3) Add the two's complement of the subtrahend to the minuend and drop the high-order 1, giving the difference:

	1	1	1		1		1	
	1	1	0	1	0	1	0	1
+	1	0	1	1	0	1	0	1
~~1~~	1	0	0	0	1	0	1	0

So 11010101_2 - 1001011_2 = 10001010_2.

The answer can be checked by making sure that 1001011_2 + 10001010_2 = 11010101_2 .