TEMASEK JUNIOR COLLEGE

Temasek Junior College

2023 JC2 H2 Computing

Data Representation 1 – Number Representation

Syllabus Objectives

- Understand that values can be represented in different number bases: denary, binary and hexadecimal.
- Represent data in binary and hexadecimal forms.
- Write programs to perform the conversion of positive integers between different number bases: denary, binary and hexadecimal forms; and display results.

Reference Resources

- https://www.bbc.co.uk/bitesize/guides/zwsbwmn/revision/1
- https://www.bbc.co.uk/bitesize/guides/zp73wmn/revision/1
- https://www.binaryhexconverter.com/
- https://coderstoolbox.net/number/

1 Number Systems

Representing numbers in daily life using the digits 0 to 9 is not the only way to do so. There are multiple number base systems, each with its own set of digits for representing a number.

The **denary** or **decimal** (**base-10**) system is what we are most familiar with. In computer science, the **binary** (**base-2**) and **hexadecimal** (**hex** or **base-16**) systems are commonly used.

You can perform arithmetic calculations within each system and even convert numbers between systems. There are several methods, each with its advantages and disadvantages.

1.1 Denary (Base-10) Numbers

We first encounter the numbers used in everyday life when learning to count, in particular, the numbers 1, 2, 3, 4, 5, 6, 7, 8, 9 and 10. This gives us ten different symbols to represent individual digits, giving rise to a base-10 number system. Numbers in this system are **denary numbers** or **decimal numbers**.

Denary digits are written using one or more of the symbols 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. The value of a denary number is defined by the place values of its digits. An example is shown below for the denary number 346

Place Value	**$100 = 10^2$**	**$10 = 10^1$**	**$1 = 10^0$**
Digit	3	4	6
Product of Digit and Place Value	$3 \times 100 = 300$	$4 \times 10 = 40$	$6 \times 1 = 6$

Adding up the products at the bottom row, we obtain $300 + 40 + 6 = 346$.

You can see that starting from the right-hand end of the number (which holds the least significant digit), the place value increases by the power of the base number. In other words, each denary digit in a number has a place value, or **weight**, that is a power of 10.

This is a **positional weighted system** representation of numbers.

1.2 Binary (Base-2) Numbers

The binary number system is base-2. Each binary digit is written with either of the symbols 0 and 1.

As with a denary number, the value of a binary number is defined by place values. Each binary digit has a place value, or weight, that is a power of 2. An example is shown below for the binary number 101110.

Place Value	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Digit	1	0	1	1	1	0
Product of Digit and Place Value	1 × 32 = 32	0 × 16 = 0	1 × 8 = 8	1 × 4 = 4	1 × 2 = 2	0 × 1 = 0

Adding up the products at the bottom row, the binary number 101110 has a value which is equivalent to the denary number 32 + 0 + 8 + 4 + 2 + 0 = 46.

1.2.1 Bits, Bytes and Binary Codes

In computers, no attempt is made to represent ten different digits. Instead, computers are engineered with transistors, which are tiny switches that allow electricity to be switched **on** and **off** in a circuit. The **on** state is represented and recognised as **1**, while the **off** state as **0**.

These circuits are combined to represent any type of information (e.g. numbers, texts, images, sounds and program instructions) as combinations of 1s and 0s. Such combinations are known as **binary codes**. In other words, all software used by computer hardware handle information using binary codes. A binary code may represent a binary number, but this does not have to be the case always. Due to how computer systems handle information, you must be able to use the binary number system so as to understand computer systems,

A binary number is said to consist of **bits**. A **bit** is a digit in the binary number system written using either of the symbols 0 and 1. In a binary number, the leftmost digit is the **most significant bit** and the rightmost digit is the **least significant bit**.

Binary codes are most often based on the use of one or more groups of eight bits. A group of eight bits treated as a single unit is called a **byte**.

1 byte = 8 bits

The byte (bit) size will determine the maximum number of possible values that can be stored:

1 bit: 0 or 1 → 2^1 = 2 values
2 bits: 00, 01, 10 or 11 → 2^2 = 4 values
3 bits: 000, 001, 010, 011, 100, 101, 110, or 111 → 2^3 = 8 values
4 bits: … → 2^4 = 16 values
…
8 bits: … → 2^8 = 256 values
16 bits: … → 2^{16} = 65 536 values
32 bits: … → 2^{32} = 4 294 967 296 values
…

Question
What is the largest possible denary number that can be represented in 1 byte?

1.3 Hexadecimal (Base-16) Numbers

Hexadecimal (base-16) numbers are often used in computer science. Each hexadecimal digit is represented by one of the following symbols: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. The symbols A through F represent the denary values 10 to 15. Hence, the 16 digits of the hexadecimal number system represent the denary values 0 through 15.

Denary Value	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hexadecimal Digit	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

As with a denary number and a binary number, the value of a hexadecimal number is defined by place values. Each hexadecimal digit has a placed value or weight that is a power of 16. An example is shown below for the hexadecimal number 2A6.

Place Value	**$16^2 = 256$**	**$16^1 = 16$**	**$16^0 = 1$**
Digit	2	A	6
Product of Digit and Place Value	$2 \times 256 = 512$	$A = 10$ $10 \times 16 = 160$	$6 \times 1 = 6$

Adding up the products at the bottom row, the hexadecimal number 2A6 has a value which is equivalent to the denary number 512 + 160 + 6 = 678.

1.3.1 Hexadecimal as a "Shorthand" for Binary

While computers always work with binary numbers, hexadecimal numbers are often used by humans as a shorthand for binary numbers in computer science. To understand why hexadecimal numbers are used, we need first to define the **nibble** as a group of four bits.

1 nibble = 4 bits

A nibble can be represented by one hexadecimal digit. This means that each byte of binary code consisting of eight bits can be written as two hexadecimal digits. Two examples are shown below, together with their denary forms.

Denary	**Binary Code**	**Hexadecimal**
10	00001010	0A
255	11111111	FF

It is sufficient to know at this juncture that the hexadecimal number is indeed shorter than its binary code equivalent. We will develop greater understanding via an in-depth study of the conversion of numbers between different forms in **Section 2**.

Quick Practice

Based on what have been covered thus far, try rehearsing the process of converting

(a) the binary codes 00001010 and 11111111

(b) the hexadecimal forms 0A and FF

as given in the aforementioned examples, back to their denary forms?

Observe that converting the binary code 00001010 to its denary form is the same as converting the binary number 1010 to its denary form. When you are doing the conversion manually, the four leading zeros in the binary code 00001010 do not interfere with the conversion process and hence need not be included. Nevertheless, in the case of a binary code, all positions in the byte i.e. the eight bits, must be either a 0 or a 1.

The same principle applies for the use of hexadecimal numbers when they are used as a shorthand for binary codes i.e. the leading zero(es) will not interfere with the conversion process . In the above example, the hexadecimal digit A already represents the denary value of 10. However, the hexadecimal digit A represents only one nibble i.e. four bits. Hence a leading hexadecimal digit 0 is included to complete the byte i.e. the eight bits.

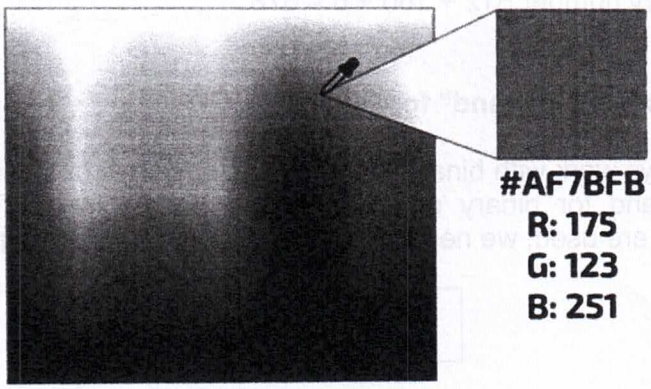
1.3.2 Common uses of Hexadecimal Numbers

Hexadecimal numbers are often used instead of binary numbers because

- they are easier to read and interpret.
- they use fewer digits to represent the same value.
- compared to binary numbers, it is less likely that a digit will be written down incorrectly.

Below are some areas of computing where you might come across the use of hexadecimals.

(A) Representation of colours



Many programming languages and software applications allow programmers, designers and digital artists to enter their choice of colour as a hexadecimal value. This is because, compared to binary values, hexadecimal values are much easier to work with.

Most electronic screens use RGB to display colour. Each colour combines 8 bits for a shade of red (R), 8 bits for a shade of green (G) and 8 bits for a shade of blue (B). Therefore, to represent any RGB colour, 24 bits are needed.

To remember a combination of 1s and 0s that make up 24 bits is definitely much more difficult than to remember a hexadecimal value of just 6 hexadecimal digits (recall that 1 hexadecimal digit represents 1 nibble i.e. 4 bits). For example, the colour orange can be represented as FFA500 in hexadecimal instead of 111111111010010100000000 in binary.

(B) Media access control (MAC) addresses

A media access control (MAC) address is a number that relates to a network interface controller. MAC addresses are usually displayed as a set of hexadecimal digits separated by colons. An example of a MAC address is b4:f7:a1:8c:8e:d3.

The colons are not stored in the memory but are added when the address is displayed to make the MAC address easier for the user to read and to communicate.

Questions

How many binary digits are required to represent the above MAC address?
Hence what is the length in bits of the above MAC address?

More about MAC addresses shall be discussed when you learn about networks.

(C) Memory dumps

When an error has occurred during the execution of a program, a memory dump may be used to output the current state of the computer's working memory as a hexadecimal representation. The memory dump may be useful in helping a user to debug the error.

```
0000000 0000 0001 0001 1010 0010 0001 0004 0128
0000010 0000 0016 0000 0028 0000 0010 0000 0020
0000020 0000 0001 0004 0000 0000 0000 0000 0000
0000030 0000 0000 0000 0010 0000 0000 0000 0204
0000040 0004 8384 0084 c7c8 00c8 4748 0048 e8e9
0000050 00e9 6a69 0069 a8a9 00a9 2828 0028 fdfc
0000060 00fc 1819 0019 9898 0098 d9d8 00d8 5857
0000070 0057 7b7a 007a bab9 00b9 3a3c 003c 8888
0000080 8888 8888 8888 8888 288e be88 8888 8888
0000090 3b83 5788 8888 8888 7667 778e 8828 8888
00000a0 d61f 7abd 8818 8888 467c 585f 8814 8188
00000b0 8b06 e8f7 88aa 8388 8b3b 88f3 88bd e988
00000c0 8a18 880c e841 c988 b328 6871 688e 958b
00000d0 a948 5862 5884 7e81 3788 1ab4 5a84 3eec
00000e0 3d86 dcb8 5cbb 8888 8888 8888 8888 8888
00000f0 8888 8888 8888 8888 8888 8888 8888 0000
0000100 0000 0000 0000 0000 0000 0000 0000 0000
*
0000130 0000 0000 0000 0000 0000 0000 0000
000013e
```

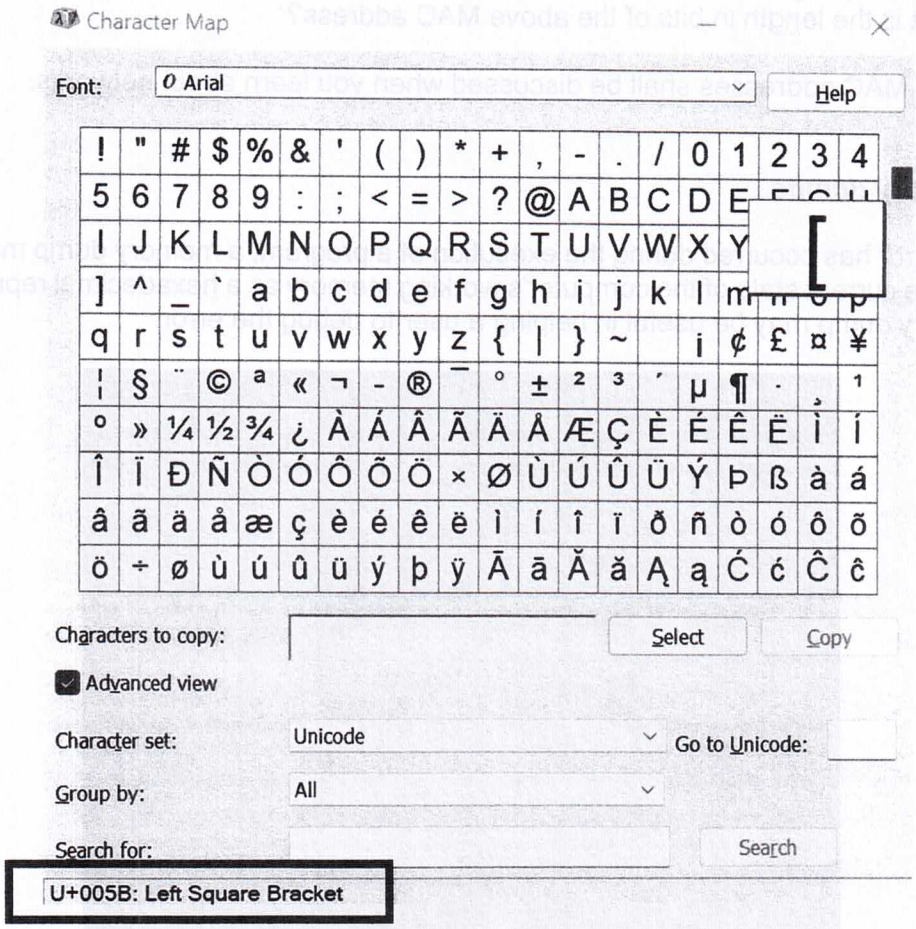
By representing the memory dump using hexadecimal numbers instead of binary numbers, the output of the memory dump can be reduced by 75%. Why?

(D) Character Codes

Another use of hexadecimal numbers is in the use of character codes.

Try entering the code "\u005B" in the Python IDLE. You would get the left square bracket ([ ) as the output. u005B is the Unicode for the left square bracket ([ ) and 005B is its hexadecimal representation.

The same Unicode can be found in the character maps of your computers, such as the one below.



2 Conversion between Number Systems

In the previous section, you would have caught a glimpse of how to convert a binary number and a hexadecimal number to the denary form. In this section, you shall study in detail the various conversion strategies.

To distinguish between number bases, we shall use a subscript to denote the number system for which the representation is in.

For example, the value of the binary (base-2) number 100 shall be written as 100_2.

This is different from the denary (base-10) number 100, which shall written as 100_{10}.

The hexadecimal (base-16) number 100 shall be written in a similar manner as 100_{16}.

2.1 Converting a Binary Number to Denary Form

Method 1: Place Values Table

Each digit of a binary number has a specific place value or weight that is a power of 2.

When converting a 4-bit binary number, the following table can be used:

	Leftmost digit (Most significant bit)			Rightmost digit (Least significant bit)
Place Value	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit				

In the case of a 8-bit binary number, the following table can be used:

	Leftmost Digit (Most significant bit)							Rightmost digit (Least significant bit)
Place Value	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit								

It follows that for a *n*-bit binary number, the table can be adjusted such that the leftmost digit will be at the place value of 2^{n-1}.

To convert a binary number into the denary form, you add up the place values only for the binary digits that are set to 1.

Example: Convert the binary number 11000110_2 into denary form.

	Leftmost Digit (Most significant bit)							Rightmost digit (Least significant bit)
Place Value	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit	1	1	0	0	0	1	1	0

Hence 11000110_2 $= 128 + 64 + 4 + 2$
$= 198_{10}$

Method 2: Multiplication by 2

Step 1	Start from the most significant bit (leftmost digit). Multiply by two and add the result to the next digit.
Step 2	Multiply by 2 again and add the result to the following digit.
Step 3	Repeat Step 2 until the result of multiplication is added to the last digit.

Example: Convert the binary number 11001_2 into denary form.

Digit	Addition	Multiplication	Result
1	-	1 × 2	2
1	1 + 2 = 3	3 × 2	6
0	0 + 6 = 6	6 × 2	12
0	0 + 12 = 12	12 × 2	24
1	1 + 24 = 25	-	-

Hence $11001_2 = 25_{10}$

Practice

Convert the following binary numbers to the denary form using both of the methods described above.

(a) 11001100_2

(b) 1011001001_2

2.2 Converting a Denary Number to Binary Form

Method 1: Division by 2

Step 1	Divide the denary number by two and write down the quotient and remainder separately.
Step 2	Divide the quotient obtained by two again and write down the new quotient and remainder separately.
Step 3	Repeat Step 2 until the quotient is 0.
Step 4	Write down all the remainders in reverse sequence, starting from the last calculated remainder to the first calculated remainder. The result obtained is the required binary form.

Example: Convert the denary number 135_{10} into binary form.

Division	Quotient	Remainder
135 / 2	67	1
67 / 2	33	1
33 / 2	16	1
16 / 2	8	0
8 / 2	4	0
4 / 2	2	0
2 / 2	1	0
1 / 2	0	1

Right most digit (Least significant bit)

Leftmost digit (Most significant bit)

Reading from the bottom row to the top row, the resulting binary form will be 10000111_2.

Hence $135_{10} = 10000111_2$.

Method 2: Place Values Table

Step 1	Draw a table with the place values of each digit.
Step 2	Find the maximum place value that is lesser than or equal to the given denary number. Set the digit that corresponds to that place value to 1.
Step 3	Subtract the place value from the denary number.
Step 4	Take the result of the subtraction and repeat Steps 2 and 3 until the result is 0.
Step 5	Set the remaining digits to 0.
Step 6	Read the resulting binary number from left to right.

Example: Convert the denary number 135_{10} into binary form.

Step 1

Place Value	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit								

Step 2: *The maximum place value that is lesser than or equal to 135 is 128.*

Place Value	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit	1							

Step 3: *135 – 128 = 7*

Step 4: *Repeating Step 2, the maximum place value lesser than or equal to 7 is 4.*

Place Value	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit	1					1		

Repeating Step 3, 7 – 4 = 3

Repeating Step 2, the maximum place value lesser than or equal to 3 is 2.

Place Value	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit	1					1	1	

Repeating Step 3, 3 – 2 = 1

Repeating Step 2, the maximum place value lesser than or equal to 1 is 1.

Place Value	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit	1					1	1	1

Repeating Step 3, 1 – 1 = 0

Step 5

Place Value	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit	1	0	0	0	0	1	1	1

Step 6: Hence $135_{10} = 10000111_2$.

Practice

Convert the following denary numbers to the binary form using both the methods described above.

(a) 199_{10}
(b) 5_{10}
(c) 76_{10}

A byte of data allows for 8 bits. What happens when more than 8 bits are required to represent a denary number?

Practice

Convert the following denary numbers to the binary form using both the methods described above.

(a) 256_{10}
(b) 550_{10}

From the above practice, you would have realised that in order to represent a denary number which requires more than eight bits i.e. greater than 255_{10}, you will need to use an extra byte.

This is because processors in computers work with binary values in multiples of 8 bits (recall 1 byte = 8 bits). It follows that additional bytes will be required where appropriate as the value increases.

2.3 Converting a Binary Number to Hexadecimal Form

Recall that 1 nibble can be represented by a hexadecimal digit. In addition, 1 nibble is equivalent to 4 bits. Hence, every four binary digits may correspond to one hexadecimal digit.

As such, to convert a binary number to hexadecimal form, the following steps can be taken.

Step 1	Working from the rightmost binary digit (least significant bit), split the binary number into nibbles (sets of four bits). Include leading zeros if necessary.
Step 2	Calculate the value of each nibble in denary form using the place values for each set of four binary digits.
Step 3	Convert the denary value into the corresponding hexadecimal digit.
Step 4	Read the resultant hexadecimal number from left to right.

Example: Convert the binary number 10100111_2 into hexadecimal form.

Step 1: *Split the number to 2 separate nibbles.*

Place Value	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$		$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit	1	0	1	0		0	1	1	1
	1st nibble					2nd nibble			

Step 2: *Calculate the denary value of each nibble.*

1st nibble = 8 + 2
= 10_{10}

2nd nibble = 4 + 2 + 1
= 7_{10}

Step 3: *Convert the denary values into hexadecimal form.*

Denary Value	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hexadecimal Digit	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Place Value	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$		$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit	1	0	1	0		0	1	1	1
	1st nibble					2nd nibble			
	$10_{10} = A_{16}$					$7_{10} = 7_{16}$			

Step 4: Hence $10100111_2 = A7_{16}$.

Practice

Convert the following binary numbers to hexadecimal form.

(a) 11100011_2

(b) 1001101_2

2.4 Converting a Hexadecimal Number to Binary Form

To convert a hexadecimal digit to the binary form, the following steps can be taken.

Step 1	Take each hexadecimal digit separately to find its corresponding denary form.
Step 2	Convert each denary value to a four-digit binary number using appropriate place values for each of the digits. As each value has to be expressed using four digits, leading zeros should be added where appropriate.
Step 3	Read the binary number from left to right.

Example: Convert the hexadecimal number $B03_{16}$ to its binary form.

Step 1: *Find the equivalent denary number of each hexadecimal digit*

Hexadecimal digit	B	0	3
Denary digit	11	0	3

$B_{16} = 11_{10}$ $0_{16} = 0_{10}$ $3_{16} = 3_{10}$

Step 2: *Convert each denary number to a 4-bit binary number using one of the two prescribed methods in* ***Section 2.2****. The method of division by 2 is shown below.*

Division	**Quotient**	**Remainder**	
11 / 2	5	1	**Right most digit (Least significant bit)**
5 / 2	2	1	
2 / 2	1	0	
1 / 2	0	1	**Leftmost digit (Most significant bit)**

Hence $11_{10} = 1011_2$

Intuitively, you should be able to tell that $0_{10} = 0000$

Division	**Quotient**	**Remainder**	
3 / 2	1	1	**Right most digit (Least significant bit)**
1 / 2	0	1	**Leftmost digit (Most significant bit)**

Hence $3_{16} = 0011$

Step 3: Hence $B03_{16} = 101100000011_2$

Hexadecimal digit	B	0	3
Denary digit	11	0	3
4-bit Binary Form	1011	0000	0011

Practice

Convert the following hexadecimal numbers to binary form.

(a) $6B_{16}$

(b) $F3_{16}$

2.5 Converting a Denary Number to Hexadecimal Form

Method 1: Division by 16

Step 1	Divide the denary number by 16 and write down the quotient and remainder separately. If the remainder is from 10 to 15, convert them to the corresponding hexadecimal digit form.
Step 2	Divide the quotient obtained and write down the new quotient and remainder separately. If the remainder is from 10 to 15, convert them to the corresponding hexadecimal digit form.
Step 3	Repeat Step 2 until the quotient is 0.
Step 4	Write down all the remainders starting from the last calculated remainder to the first calculated remainder. The result obtained is the required binary form.

Example: Convert the denary number 125_{10} to its hexadecimal form.

Division	Quotient	Remainder
125 / 16	7	$13 = D_{16}$
7 / 16	0	7

Reading from the bottom row to the top row, the resulting hexadecimal form will be $7D_{16}$.

Hence $125_{10} = 7D_{16}$.

Method 2: Using the binary form as intermediate

Step 1	Convert the denary number to binary form.
Step 2	Convert the binary form to hexadecimal form.

Example: Convert the denary number 103_{10} to its hexadecimal form.

Step 1: *Convert the denary number to a binary number using one of the two prescribed methods in* ***Section 2.2****. The method of using place values table is shown below.*

64 < 103

Place Value	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit		1						

103 – 64 = 39; 32 < 39

Place Value	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit		1	1					

39 – 32 = 7; 4 < 7

Place Value	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit		1	1			1		

7 – 4 = 3; 2 < 3

Place Value	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit		1	1			1	1	

3 – 2 = 1; 1 = 1

Place Value	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit		1	1			1	1	1

1 – 1 = 0

Place Value	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit	0	1	1	0	0	1	1	1

Hence $103_{10} = 01100111_2$.

Step 2: *Perform the steps for converting binary to hexadecimal number.*

Place Value	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$		$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit	0	1	1	0		0	1	1	1
Denary Value	4 + 2 = 6					4 + 2 + 1 = 7			
Hexadecimal Value	6					7			

Hence $103_{10} = 67_{16}$

Practice

Convert the following denary numbers to hexadecimal form using both the methods prescribed.

(a) 100_{10}

(b) 256_{10}

2.6 Converting a Hexadecimal Number to Denary Form

There are two methods to convert a hexadecimal number to its denary form.

Method 1: Place Values Table

Each digit of a hexadecimal number has a specific place value or weight that is a power of 16.

Example: Convert the hexadecimal number $F8_{16}$ to its denary form.

Place Value	$16^1 = 16$	$16^0 = 1$
Hexadecimal Digit	F = 15	8

Hence $F8_{16} = (15 \times 16) + (8 \times 1) = 248_{10}$.

Method 2: Using the binary form as intermediate

Step 1	Convert the hexadecimal number to binary form.
Step 2	Convert the binary form to denary form.

Example: Convert the hexadecimal number $B7_{16}$ to its denary form.

Step 1: Convert the hexadecimal number to binary form.

Hexadecimal digit	B	7
Denary digit	11	7

Division	**Quotient**	**Remainder**	
11 / 2	5	1	**Right most digit (Least significant bit)**
5 / 2	2	1	
2 / 2	1	0	
1 / 2	0	1	**Leftmost digit (Most significant bit)**

Hence $11_{10} = 1011_2$

Division	**Quotient**	**Remainder**	
7 / 2	3	1	**Right most digit (Least significant bit)**
3 / 2	1	1	
1 / 2	0	1	**Leftmost digit (Most significant bit)**

Hence $7_{10} = 0111_2$ (note that a leading 0 is added).

Hexadecimal digit	B	7
Denary digit	11	7
Binary digit	1101	0111

Step 2: Convert the binary form to denary form.

Place Value	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
Binary Digit	1	0	1	1	0	1	1	1

Hence $B7_{16} = 128 + 32 + 16 + 4 + 2 + 1 = 183_{10}$

Practice

Convert the following hexadecimal numbers to denary form using both the methods prescribed.

(a) $E13_{16}$

(b) $D15_{16}$