Introduction

A computer has many electronic components that work as switches. These components have two logics as input and output: ON and OFF. A similar logic is used to represent data in binary form. ON is represented as 1 and OFF is represented as 0.

Place value of denary system

The denary system has a base value of 10. It counts in multiples of 10. The number 7324 has 7 thousands, 3 hundreds, 2 tens and 4 ones. The following figure illustrates the place values in a denary system.

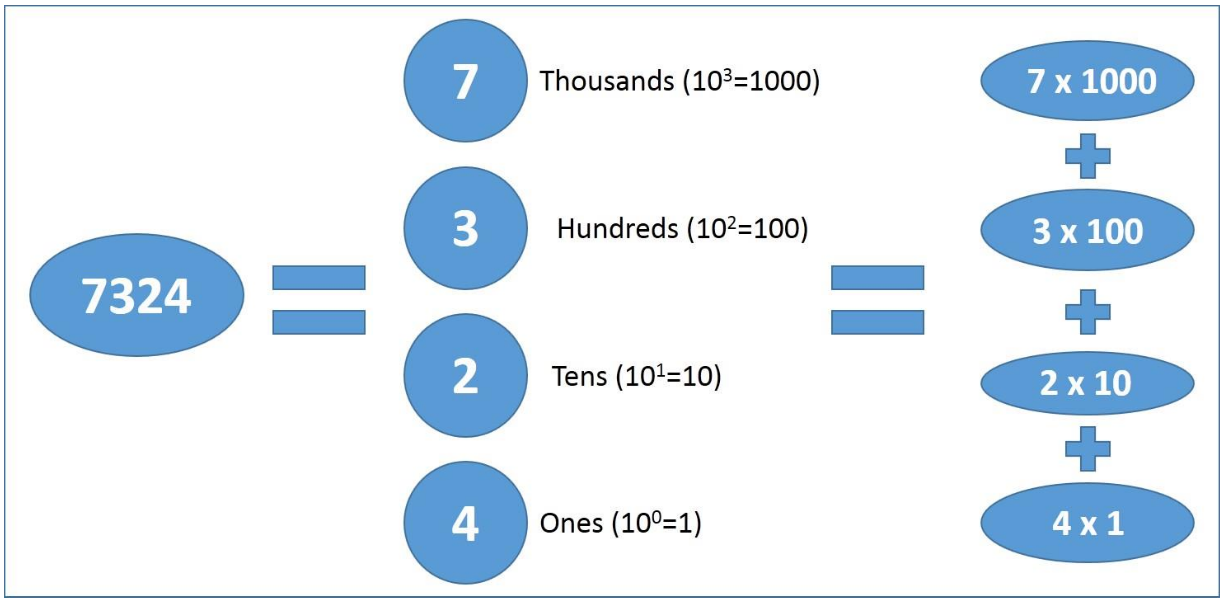

Figure 1: Place values in a denary system

Place value of binary system

The binary numbers can also be represented using place values. The place values have a base 2. The first 8 digits are represented in the table below.

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1

In th

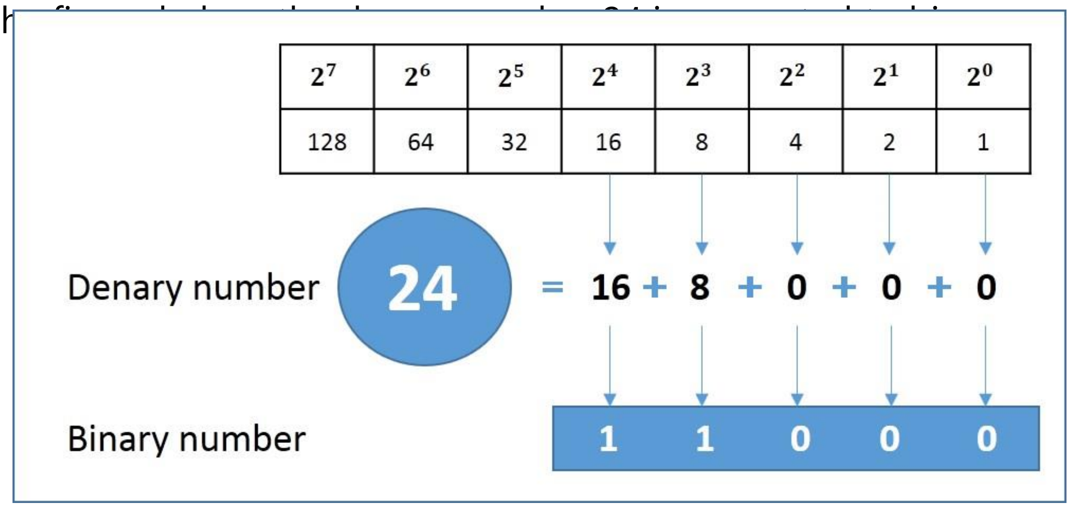

Figure 2: Binary equivalent of 24

In th

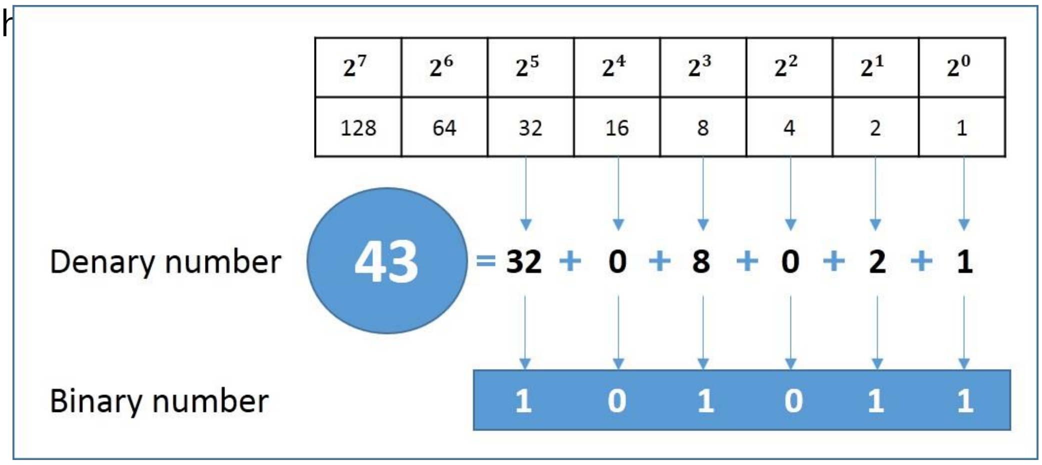

Figure 3: Binary equivalent of 43

Size of computer memory

A binary digit is referred to as a bit. A nibble consists of 4 bits. A byte consists of 8 bits. A byte is the smallest unit of memory of the computer system. The memory sizes available with computers are in multiples of 8 such as 16-bit systems, 32-bit systems, etc.

Name of the memory size	Number of bits	Equivalent denary value
1 kilobyte (1 kB)	2^{10}	1024 bytes
1 megabyte (1 MB)	2^{20}	1024^2 bytes
1 gigabyte (1 GB)	2^{30}	1024^3 bytes
1 terabyte (1 TB)	2^{40}	1024^4 bytes
1 petabyte (1 PB)	2^{50}	1024^5 bytes

Converting denary to binary

A denary number is converted to binary by dividing it by 2 and calculating the remainders as shown in the following example. Let us convert 91 to a binary number.

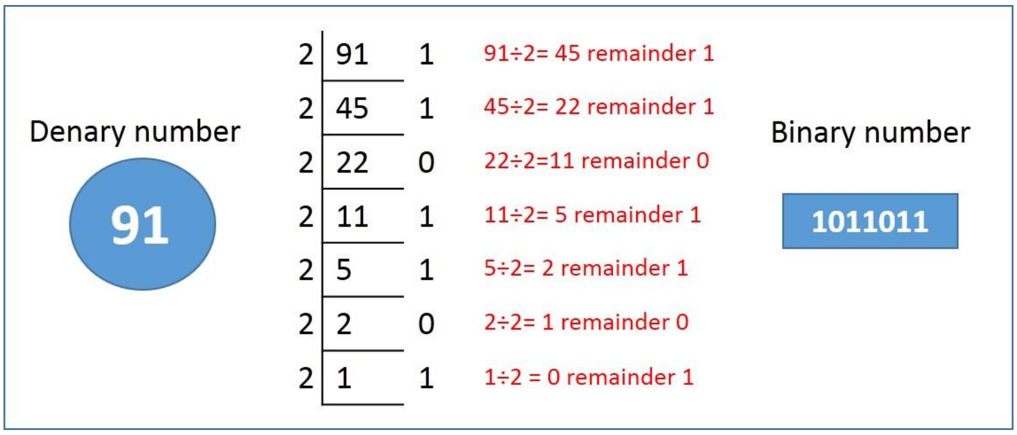

Figure 3: Converting denary number 91 to binary

The binary equivalent of the denary number 91 is obtained by arranging the remainders in reverse order.

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	1	0	1	1	0	1	1

The answer can be checked by:

$(0 \times 128) + (1 \times 64) + (0 \times 32) + (1 \times 16) + (1 \times 8) + (0 \times 4) + (1 \times 2) + (1 \times 1)$ =91

Binary combinations

Audio files use bit depth to denote the number of bits of information in each sample. A one-bit system has one-place value and can have 2 possible combinations: 0 or 1.
Similarly, a 2-bit system has two-place values and has 4 possible combinations, as shown in the table below:

Place value 2	Place value 1	Binary number	Denary number
0	0	00	0
0	1	01	1
1	0	10	2
1	1	11	3

Similarly, a 3-bit system has three-place values and has 8 possible combinations, as shown in the table below:

Place value 3	Place value 2	Place value 1	Binary number	Denary number
0	0	0	000	0
0	0	1	001	1
0	1	0	010	2
0	1	1	011	3
1	0	0	100	4
1	0	1	101	5
1	1	0	110	6
1	1	1	111	7

From the above examples, it can be noted that an n-bit system has 2^n possible combinations.

Representing numbers

Programmers use many arithmetic operations in a program. The numbers are either represented as integers or floating point numbers. Integers are whole numbers and floating point numbers are used to represent numbers with decimal points. A 16-bit system can represent integers up to $2^{16}-1=65535$. 8-bit, 16-bit, 32-bit and 64-bit are the most common bit lengths.

Adding binary numbers

Binary numbers are added in a column method as the denary numbers are added. Let us consider the following example of adding 0101 and 1011.

	Place value 4	Place value 2	Place value 3	Place value 1
Carry			1	
Number 1	0	1	0	1
Number 2	1	0	0	1
Sum	1	1	1	0

When the binary digits 1 and 0 are added, the sum is 1. When both the digits are 0, the sum is 0. When both the digits are 1, the sum is 10 and 1 is carried over. 0101 and 1001 represent the denary numbers 5 and 9. The sum of 5 and 9 is 14. Convert the sum obtained to denary number.

$(8\times1)+(4\times1)+(2\times1)+(0\times1)=14$

Overflow Error

A CPU with an 8-bit register has a capacity of up to 11111111 in binary. If an extra bit is added, it is said to be an overflow error. Consider addition of two binary numbers 11101101 and 10000100 as shown below:

		2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Carry	1				1	1			
Number 1		1	1	1	0	1	1	0	1
Number 2		1	0	0	0	0	1	0	0
Sum	1	0	1	1	1	0	0	0	1

The sum of these two numbers is bigger than 8 bits (an extra bit than the register can hold). The computer thinks that 11101101+10000100=01110001 as it does not have space to store the extra bit. The number of bits a register can hold is called word size. Exceeding the capacity of word size in a register results in an overflow error.

Binary shifts

Let us consider the denary number 6. Its binary equivalent is 0110.

Denary number	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
6	0	0	0	0	0	1	1	0

The binary equivalent of 12 (6 × 2) is:

Denary number	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
12	0	0	0	0	1	1	0	0

It can be noted that the binary equivalent of 6 shifted to the left is the binary equivalent of 12. Now, the binary equivalent of 24 is:

Denary number	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
24	0	0	0	1	1	0	0	0

Again, it can be noted that the binary equivalent of 12 shifted to the left is the binary equivalent of 24. It can be summarised that when a denary number is multiplied by 2, its binary equivalent shifts left by 1 place. Also, multiplication by 4 results in the shift of the binary equivalent by 2 places and multiplication by 8 results in a shift of the binary equivalent by 3 places and so on.

Representing negative numbers

Signed integers can be either positive or negative. An extra bit is used to represent the sign of a number in binary representation. In case of signed numbers, the leftmost bit is used to represent sign and is called the sign bit. 0 represents a positive number and 1 represents a negative number. The rest of the bits represent the magnitude of the number. Consider the 8-bit binary number 10001101.

	Sign bit	Magnitude						
Binary	1	0	0	0	1	1	0	1
Denary	-13							

The smallest number that can be represented using 8 bits is 11111111 (-127) and the largest number is 01111111 (+127). Similarly, the signed number can be represented in 32-bits, 64-bits and so on.

Finding two's complement is an alternate method to represent negative numbers. This method is used by most computers to perform mathematical operations.

Let us consider an example of representing -5. The binary value of 5 is 101. The leftmost bit is added to represent the positive sign. +5 is 0101. Each bit is inverted and, hence, the 0101 becomes 1010. 1 is added to this number. 1010 + 1=1011. A few examples of representing binary numbers using two's complement are given in the table below:

Sign bit (-2^3)	(2^3)	(2^2)	(2^1)	Denary number
1	0	1	1	-5
0	1	1	1	7
1	0	0	1	-7
1	1	0	1	-3

Sum of numbers: Using two's complement

Let us consider adding -4 and 3 using two's complement.

	Sign bit (-2^3)	(2^3)	(2^2)	(2^1)
Carry				
Number 1 = -4	1	1	0	0
Number 2 = 3	0	0	1	1
Sum	1	1	1	1

Converting the sum 1111 into denary number,
-8+4+2+1=-1