Contents

The Binary Numbering System

- The Binary Numbering System is the most fundamental numbering system in all digital and computer based systems

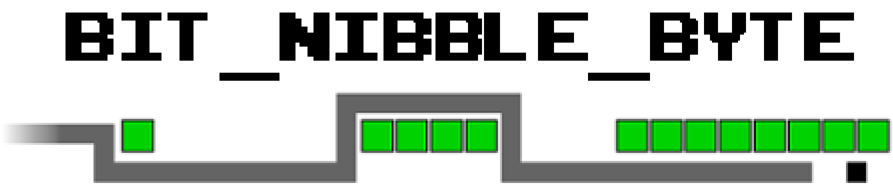
- Electronic computers use **electricity** to represent real information.

- **Digital logic** and **computer systems** use just two values or states to represent a condition, a logic level "1" or a logic level "0"

10 Min video *Why do computers use binary, anyway*? https://goo.gl/KZwkss

Computers store millions of pieces of binary information and therefore a way of grouping them together is needed

A single binary digit is called a bit which stands for **bi**nary digi**t**



Four bits (half a byte) together is called a nibble.

Eight bits are called one byte.

1024 bytes are called one kilobyte (kb)

1024 kilobytes are called one megabyte (mb)

1024 megabytes are called one gigabyte (gb)

1024 gigabytes are called one terabyte (tb)

Number of Bytes	Common Name
1,024 (2^{10})	kilobyte (kb)
1,048,576 (2^{20})	Megabyte (Mb)
1,073,741,824 (2^{30})	Gigabyte (Gb)
a very long number! (2^{40})	Terabyte (Tb)

As microprocessor systems became increasingly larger, the individual binary digits (bits) are grouped together into 8's to form a single BYTE with most computer hardware such as hard drives and memory modules commonly indicate their size Gigabytes or Terabytes

In the binary numbering system, a binary number such as 101100101_2 is a string of "1's" and "0's" with each digit along the string from right to left having a value twice that of the previous digit.

	Column 8	Column 7	Column 6	Column 5	Column 4	Column 3	Column 2	Column 1
$Base^{exp}$	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Weight	**128**	**64**	**32**	**16**	**8**	**4**	**2**	**1**

Where:

Base^{exp} is the representation of the binary digit which will be either "0" or "1"

Weight is the decimal weighting for the corresponding binary digit 1

How many possible values can the numbers represent?

One bit can store two possible values (0 or 1)

How many possible ways can **two bits** be arranged into?

00	both zero
01	a zero and a one
10	a one and a zero
11	both one

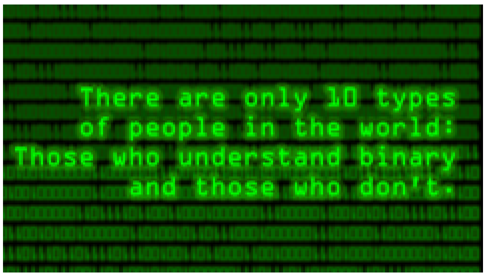
Therefore, **two bits** give four different values

i.e. 1 bit can store 2^1 (=2) possible values
2 bits can store 2^2 (=4) possible values
3 bits can store 2^3 (=8) possible values
4 bits can store 2^4 (=16) possible values
:
:
16 bits can store 2^{16} (=65,536) possible values
24 bits can store 2^{24} (=16,777,216) possible value and so on…

100 BC – Binary System – to 1689

- A **binary code** is a way of representing text or computer processor instructions by the use of the binary number system's two-binary digits **0** and **1**

- A bit string is assigned to each particular symbol or instruction.



- A binary string of eight binary digits (bits) can represent any of 256 possible values
- WHY…? HINT _ _ _ _ _ _ _ _

0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111

There are 16 different values that can be stored in one nibble

Can you see a pattern?

Recall: 4 bits can represent $2^4 = 16$ possible values

0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111

In the right column, the zero and one swap every time

0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111

In the second column, the zero and one are on for two values, then off for two values.

	Column 8	Column 7	Column 6	Column 5	Column 4	Column 3	Column 2	Column 1
$Base^{exp}$	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Weight	128	64	32	16	8	4	2	1

0000	Zero	
0001	1	(1 x 1)
0010	2	(1 x 2 + 0 x 1)
0011	3	(1 x 2 + 1 x 1)
0100	4	(1 x 4 + 0 x 2 + 0 x 1)
0101	5	(1 x 4 + 0 x 2 + 1 x 1)
0110	6	(1 x 4 + 1 x 2 + 0 x 1)
0111	7	(1 x 4 + 1 x 2 + 1 x 1)
1000	8	(1 x 8 + 0 x 4 + 0 x 2 + 0 x 1)
1001	9	(1 x 8 + 0 x 4 + 0 x 2 + 1 x 1)
1010	10	(1 x 8 + 0 x 4 + 1 x 2 + 0 x 1)
1011	11	(1 x 8 + 0 x 4 + 1 x 2 + 1 x 1)
1100	12	(1 x 8 + 1 x 4 + 0 x 2 + 0 x 1)
1101	13	(1 x 8 + 1 x 4 + 0 x 2 + 1 x 1)
1110	14	(1 x 8 + 1 x 4 + 1 x 2 + 0 x 1)
1111	15	(1 x 8 + 1 x 4 + 1 x 2 + 1 x 1)

0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111

In the third column, the ones and zeros are on for four values, then off for four values.

0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111

In the fourth column, the ones are on for eight values and off for eight values.

	Column 8	Column 7	Column 6	Column 5	Column 4	Column 3	Column 2	Column 1
$Base^{exp}$	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Weight	**128**	**64**	**32**	**16**	**8**	**4**	**2**	**1**

Ever seen a common form of the binary code?

- Almost everyone has seen or heard of it
- Common in public buildings/recreation areas such as playgrounds
- Its characters are an 8-bit binary number: the bit for a single position is either 0 for flat and 1 for raised.
- WHAT IS IT???

Braille characters live in a 4×2 matrix.

This means there are eight positions where the surface is either flat or raised.

You can naturally denote a Braille character by an 8-bit binary number: the bit for a single position is either 0 for flat and 1 for raised.

Converting Binary ($base_2$) to Decimal ($base_{10}$) and Vice Versa

Division Method of Conversion

Decimal ($base_{10}$) to Binary ($base_2$) Conversion

If you want to convert the decimal number 1584_{10} to $base_2$ ***division by 2*** can be used, with each calculation remainder being the binary representation working from the lsb to the msb (from right to left during the calculation)

					REMAINDER
1584	/	2	=	792	**0**
792	/	2	=	396	**0**
396	/	2	=	198	**0**
198	/	2	=	99	**0**
99	/	2	=	49	**1**
49	/	2	=	24	**1**
24	/	2	=	12	**0**
12	/	2	=	6	**0**
6	/	2	=	3	**0**
3	/	2	=	1	**1**
1	/	2	=	0	**1**

Now, taking the top remainder and using it as the rightmost bit = the LSB, we can complete the binary equivalent: 11000110000_2

SOLUTION: **$1,584_{10} = 11000110000_2$**

An alternative to the Division Method of Conversion above, the Sum of Weights Method of Conversion can be used to convert between decimal and binary number bases.

Convert Binary to Decimal using the Sum of Weights Method:

Example 1: 11011010

Example 2: 10011101

	Column 8	Column 7	Column 6	Column 5	Column 4	Column 3	Column 2	Column 1
Base^{exp}	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Weight	**128**	**64**	**32**	**16**	**8**	**4**	**2**	**1**
Example1	1	1	0	1	1	0	1	0
Example2	1	0	0	1	1	1	0	1

Example1:

11011010 = (1 * 128) + (1 * 64) + (0 * 32) + (1 * 16) + (1 * 8) + (0 * 4) + (1 * 2) + (0 * 1) =

= 128 + 64 + 16 + 8 + 2 = **218**

Example2:

10011101 = (1 * 128) + (0 * 64) + (0 * 32) + (1 * 16) + (1 * 8) + (1 * 4) + (0 * 2) + (1 * 1) =

= 128 + 16 + 8 + 4 + 1 = **157**

Decimal Digit Value	256	128	64	32	16	8	4	2	1
Binary Digit Value	1	0	1	1	0	0	1	0	1

101100101_2

= 256+0+64+32+0+0+4+0+1

= 357_{10}

Convert Decimal to Binary using Repeated Subtraction Method:

EXERCISE: What is 357_{10} represented as an 8-bit binary number? i.e. _ _ _ _ _ _ _ _
ANSWER:

1. Start with 357 and, looking at this table, which number can be used to subtract from 357?
2. In this case, it's $2^8 = 256$
3. Subtract this from 357 leaving a remainder of 101_{10}

Repeat this until you reach the end:

1. Start with 101 and, looking at this table, which number can be used to subtract from 101?
2. In this case, it's $2^6 = 64$
3. Subtract this from 101 leaving a remainder of 37
4. REPEAT..

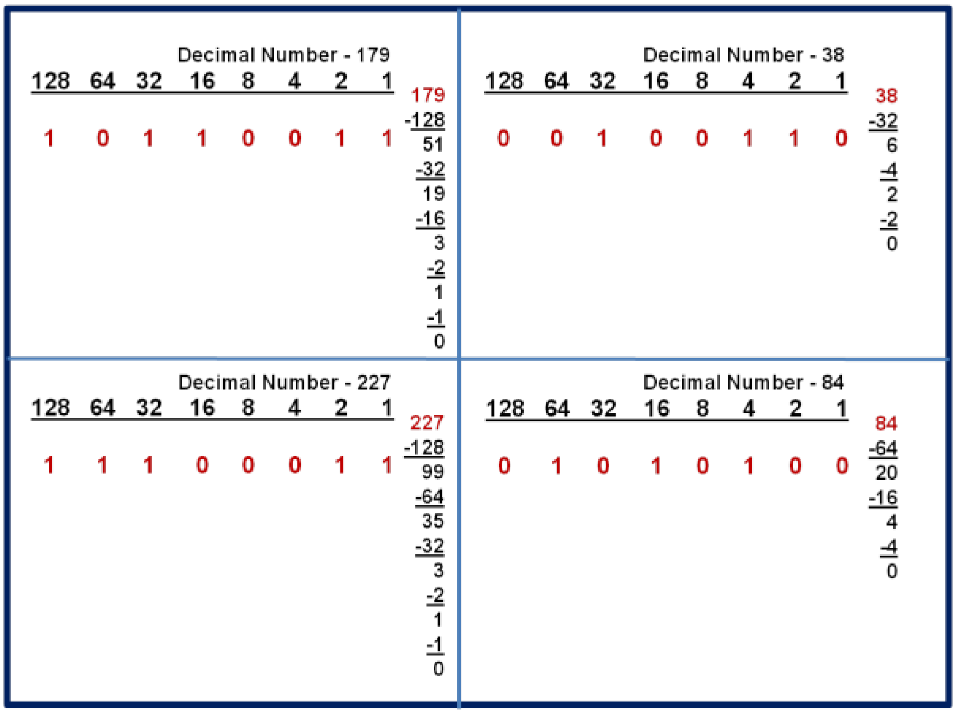
	Column 8	Column 7	Column 6	Column 5	Column 4	Column 3	Column 2	Column 1
Base^{exp}	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Weight	**128**	**64**	**32**	**16**	**8**	**4**	**2**	**1**

=	256	(357-256=101_{10} remaining to be represented as binary)
=	64	(101-64=37_{10} remainder)
=	32	(37-32=5_{10} remainder)
=	4	(5-4=1_{10} remainder)
=	1	end

Therefore, for every weighted positional number that you've used in the calculation above, this will represent a binary 1 value in your solution, therefore:

Weight	**256**	**128**	**64**	**32**	**16**	**8**	**4**	**2**	**1**
Our Solution gave a **1** in the following positions	**256**		**64**	**32**			**4**		**1**
Binary Digit Value	**1**	**0**	**1**	**1**	**0**	**0**	**1**	**0**	**1**

Some more examples:



Decimal number	Binary number code 8 4 2 1	Binary to decimal conversion
0	0000	= 0 + 0 = 0
1	0001	= 0 + 1 = 1
2	0010	= 2 + 0 = 2
3	0011	= 2 + 1 = 3
4	0100	= 4 + 0 = 4
5	0101	= 4 + 1 = 5
6	0110	= 4 + 2 = 6
7	0111	= 4 + 2 + 1 = 7
8	1000	= 8 + 0 = 8
9	1001	= 8 + 1 = 9

6_{10} as a 4-digit binary representation $=0110_2$ because, from the left to right

$2^3+2^2+2^1+2^0$

0110 $=0+2^2+2^1+0$

$=0+4+2+0$

$=6$

Binary to Decimal Summary

- A "BIT" is the abbreviated term derived from BInary digiT
- A Binary system has only two states, Logic "0" and Logic "1" giving a base of 2
- A Decimal system uses 10 different digits, 0 to 9 giving it a base of 10

- A Binary number is a weighted number who's weighted value increases from right to left
- The weight of a binary digit doubles from right to left
- A decimal number can be converted to a binary number by using the sum-of-weights method or the repeated division-by-2 method
- When we convert numbers from binary to decimal, or decimal to binary, subscripts are used to avoid errors (10_{10} is decimal and 10_2 is binary)

The one main disadvantage of binary numbers is that the binary string equivalent of a large decimal base-10 number can be quite long

Number Bases Reference Table

BASE $_{10}$	BASE $_{2}$	BASE $_{16}$	BASE $_{8}$
DECIMAL	**BINARY**	**HEXADECIMAL**	**OCTAL**
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	8	10
9	1001	9	11
10	1010	A	12
11	1011	B	13
12	1100	C	14
13	1101	D	15
14	1110	E	16
15	1111	F	17