

AQA Computer Science A-Level
4.5.2 Number bases
Advanced Notes



Specification:

3.5.2.1 Number base:

Be familiar with the concept of a number base, in particular:

- decimal (base 10)
- binary (base 2)
- hexadecimal (base 16)

Convert between decimal, binary and hexadecimal number bases.

Be familiar with, and able to use, hexadecimal as a shorthand for binary and to understand why it is used in this way.



Number bases

The same number can be represented in a variety of **different ways**. Humans use **base 10**, sometimes called **denary** or **decimal**, but we could use **binary** or **hexadecimal**.

Decimal (base 10)

Decimal is the number base that humans use to count, perhaps because we have **ten** fingers. Decimal uses the ten digits **0 through to 9** to represent numbers.

Decimal numbers can be denoted with a **subscript 10**, like so:

$$27_{10}$$

Binary (base 2)

Binary uses **only two characters** for each digit, either **a 1 or a 0**. These two values can easily be represented by computers with **high or low current**.

Binary numbers can be denoted with a **subscript 2**, like so:

$$10110010_2$$

Synoptic Link

Methods exist for representing **negative** and **non-integer** numbers with binary.

These methods are covered in **binary number system**.



Hexadecimal (base 16)

In contrast to decimal, **hexadecimal** uses the digits **0 through to 9** followed by the uppercase characters **A to F** to represent the **decimal** numbers 0 to 15.

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

This means that hexadecimal can make use of **16 different characters** for each digit.

Hexadecimal numbers can be denoted with a **subscript 16**, like so:

$$7E_{16}$$

Of all the number bases covered by this course, hexadecimal is the **most compact**. This means that it can represent **the same number** as binary or decimal while **using far fewer digits**.

For example, the number **733452** uses **six** digits in decimal, a whopping **twenty** in binary (10110011000100001100) but just **five** in hexadecimal (B310C).

As you'll see later in these notes, it's **easy** to convert between hexadecimal and binary. This, combined with its **compact** nature, makes hexadecimal useful as a **shorthand representation for binary**.



Conversions

Converting from binary to decimal

You can convert between binary and decimal by using **place value headers**. Starting with **one** and increasing in **powers of two**, placing larger values **to the left** of smaller values. For example, the binary number 10110010_2 could have place value headers added as follows:

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

The binary number could then be converted to decimal by **adding together** all of the place values with a **binary one** below them.

$$128 + 32 + 16 + 2 = 178$$

So the binary number 10110010_2 is equivalent to the decimal number 178_{10} .

Converting from decimal to binary

When converting from decimal to binary, you use the same place value headers. Starting from the left hand side, you place a one if the value is less than or equal to your number, and a zero otherwise.

Once you've placed a one, you must subtract the value of that position from your number and continue as before.

Let's say we're converting the number **53** to binary. First, write out your place value headers in powers of two. Keep on going until you've written a value which is larger than your number. For 53, we're going to go up to 64.

64	32	16	8	4	2	1
----	----	----	---	---	---	---



Now, starting from the left, compare the place value to your number. 64 is greater than 53 so we place a 0 under 64.

64	32	16	8	4	2	1
0						

Moving to the right, we see that 32 is lower than 53, so we place a 1 under 32.

64	32	16	8	4	2	1
0	1					

Because we've placed a one, we have to subtract 32 from 53 to find what's left to be represented. In this case, $53 - 32 = 21$.

We move to the right again and find 16, which is lower than 21, so we place a 1 under 16.

64	32	16	8	4	2	1
0	1	1				

Again, because we've placed a one, we have to calculate a new value. $21 - 16 = 5$.

Moving right, we find 8. This is larger than 5 so we place a 0.

64	32	16	8	4	2	1
0	1	1	0			

After moving right again, we find 4. As 4 is lower than 5, we place a 1.

64	32	16	8	4	2	1
0	1	1	0	1		



Having placed a 1, we must again calculate a new value. $5 - 4 = 1$.

Moving right to find 2, we place a 0 as 2 is greater than 1.

64	32	16	8	4	2	1
0	1	1	0	1	0	

Moving right for the last time, we have 1. $1 = 1$ so we place a 1.

64	32	16	8	4	2	1
0	1	1	0	1	0	1

Now that we've placed a 0 or a 1 under each place value, we have our answer. Although it's acceptable to [remove any leading 0s](#), it may be preferable to add 0s to the start of your answer to make it a [whole number of bytes](#) (a multiple of 8 bits).

Note

If you're using signed binary, you'll need to take care with any leading 0s.

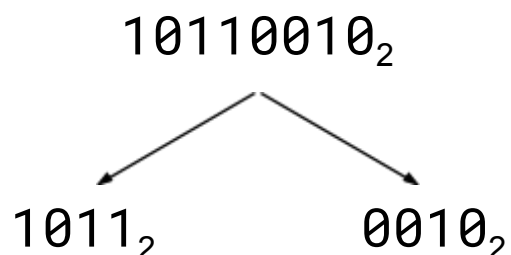
$$53_{10} = 0110101_2 = 110101_2 = 00110101_2$$



Converting from binary to hexadecimal

In order to convert from binary to hexadecimal, the binary number must first be split into nybbles. A nybble is four binary bits, or half a byte.

For example, the binary number 10110010_2 would be split into two nybbles:



Each binary nybble is then converted to decimal as in the previous example:

8	4	2	1	8	4	2	1
1	0	1	1	0	0	1	0
$8 + 2 + 1 = 11$				$2 = 2$			

Once each nybble has been converted to decimal, the decimal value can be converted to its hexadecimal equivalent like so:

$$11_{10} = B_{16} \qquad 2_{10} = 2_{16}$$

Finally, the hexadecimal digits are concatenated to form a hexadecimal number:

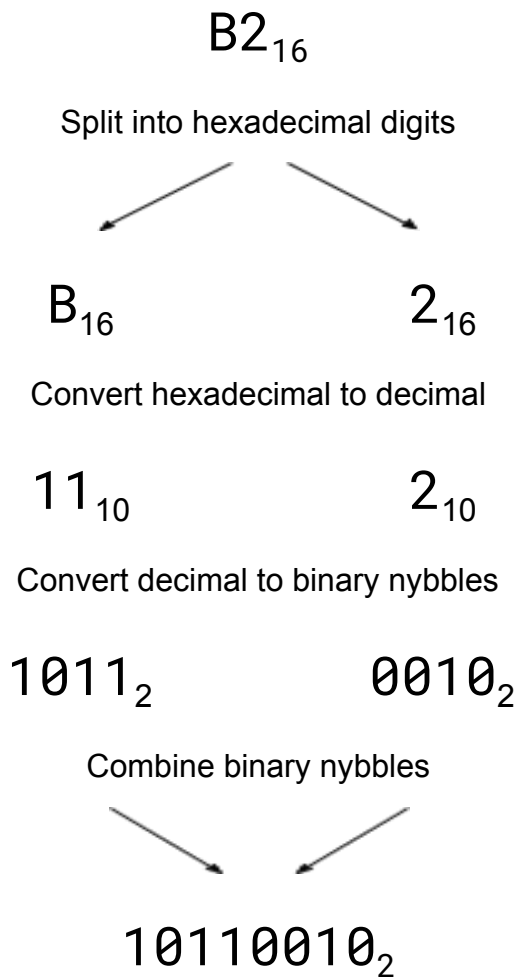
$$10110010_2 = B2_{16}$$



Converting from hexadecimal to decimal (by converting from hexadecimal to binary)

When converting from hexadecimal to decimal, it's usually easiest to go [via binary](#).

First convert your hexadecimal number to [binary](#). Do this by converting each hexadecimal digit to a [decimal digit](#) and then to a [binary nybble](#) before [combining the nybbles](#) to form a single binary number.



Note

When converting from hexadecimal to binary, use this method but stop here.

128	64	32	16	8	4	2	1
1	0	1	1	0	0	1	0

$$128 + 32 + 16 + 2 = 178$$

$$B2_{16} = 178_{10}$$



Converting from decimal to hexadecimal

Converting a number from decimal to hexadecimal is simply the **reverse of converting from hexadecimal to decimal**.

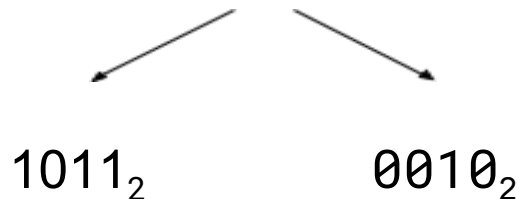
First convert your **decimal number to binary**, then **split it into decimal nybbles**. Each nybble can then be **converted to decimal** before **converting decimal to hexadecimal**.

Let's convert 178_{10} to hexadecimal.

First, **convert to binary** using the method described under "Converting from decimal to binary". If required, add leading 0s to your answer so that you have a **whole number of nybbles**.

$$10110010_2$$

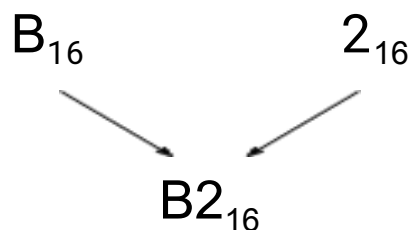
Now **split** your binary number **into nybbles**.



Then **convert** each nybble **to decimal**.

$$11_{10} \qquad 2_{10}$$

Lastly, **convert decimal to hexadecimal** and **combine** to form a single number.



$$178_{10} = B2_{16}$$

