

# Information representation

## Revision objectives

By the end of the chapter you should be able to:

- show understanding of the basis of different number systems; use the binary, denary and hexadecimal number systems; and convert a number from one number system to another
- express a positive or negative integer in two's complement form
- show understanding of, and be able to represent, character data in its internal binary form
- express a denary number in binary coded decimal (BCD) and vice versa and describe practical applications where BCD is used
- show understanding of how data for a bitmapped image is encoded
- use the terminology associated with bitmaps: pixel, file header, image resolution, screen resolution
- perform calculations estimating the file size for bitmapped images of different resolutions
- show understanding of how data for a vector graphic is represented and encoded
- use the terminology associated with vector graphics: drawing object, property and drawing list
- show understanding of how typical features found in bitmapped and vector graphics software are used in practice and are therefore appropriate for a given task
- show understanding of how sound is represented and encoded
- use the associated terminology: sampling, sampling rate, sampling resolution
- show understanding of how file sizes depend on sampling rate and sampling resolution
- show understanding of how typical features found in sound-editing software are used in practice
- show understanding of the characteristics of video streams: frame rate (frames/second); interlaced and progressive encoding; video interframe compression algorithms and spatial and temporal redundancy; multimedia container formats
- show understanding of how digital data can be compressed, using 'lossless' (including run-length encoding, RLE) or 'lossy' techniques.

## 1.01 Number representation

We present any denary number with some combination of the digits 0, 1, 2, 3, 4, ..., 8 and 9.

Any number system is founded on the concepts of:

- a base
- that digits in certain positions each have a place value
- the number of possible digits used is the base.

## Denary system

We were taught to use the **denary** (or **decimal**) numbering system – that is, using base 10 with possible digits 0, 1, 2, ..., 8 and 9.

### TERMS

**denary (decimal):** numbering system using base 10 with possible digits 0, 1, 2, ..., 8 and 9

**binary:** numbering system using base 2

## Binary system

The base 2 numbering (**binary**) system has possible digits 0 and 1.

This can be summarised as shown in Table 1.01.

System	Base	Possible digits	Place values				
denary	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9	etc.	$10^3$	$10^2$	$10^1$	Units
				8	7	2	6
binary	2	0, 1	etc.	$2^3$	$2^2$	$2^1$	Unit
				1	0	1	1

**Table 1.01** Denary and binary numbering systems

Intuitively we would read the denary number as ‘eight thousand, seven hundred and twenty six’.

Appreciate that it is based on the place-value concept that we have:

$$(8 \times 1000) + (7 \times 100) + (2 \times 10) + 6 = 8726$$

Applying the same method to the binary pattern 10111, computes the pattern as binary number:

$$(1 \times 16) + (0 \times 8) + (1 \times 4) + (1 \times 2) + 1 = 23$$

## Hexadecimal system

The base 16 numbering system can be summarised as shown in Table 1.02.

System	Base	Possible digits	Place values				
hexadecimal	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +0, +1, +2, +3, +4, +5 A, B, C, D, E, F	etc.	$16^3$	$16^2$	$16^1$	Units
					1	B	5

**Table 1.02** Hexadecimal numbering system

The Hexadecimal numbering system follows our three basic rules.

Since the ‘digits’ allowed in base 16 extend past 9 then we need a way to represent 10, 11, 12, 13, 14 and 15. The solution in hexadecimal is to use the characters A to F as shown.

**TIP**

If we did not do this, then the hexadecimal representation 13, could either be interpreted as 13 denary or  $(1 \times 16) + 3 = 19$  denary.

The number shown in Table 1.02 is:

$$(1 \times 256) + (B \times 16) + 5 = 256 + 176 + 5 = 437 \text{ denary}$$

## Conversion between different number presentations

We can now convert from binary to denary and vice versa and also from hexadecimal to denary and vice versa. What about conversion between binary and hexadecimal?

One approach would be to convert into denary first – but there is a more direct way.

**Example:**

Convert 0011110101010100 into hexadecimal

Divide the binary into groups of four binary digits:

0011 1101 0101 0100

Write the denary for each group

0011 1101 0101 0100  
 3 13 5 4

We can then convert each denary number to its hexadecimal equivalent:

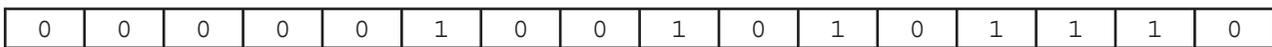
3 D 5 4 = 3D54 hex

The method can be used in reverse to convert from hexadecimal to binary.

Example: Convert 4AE hex to a binary number stored as two bytes.

Hexadecimal:	4	A	E
Denary:	4	10	14
Binary:	0100	1010	1110

'Stored as two bytes' means this number will be stored as 16-bit binary pattern as shown in Figure 1.01.



**Figure 1.01** A binary number stored as two bytes

Note the need to pack out the leftmost group of four bits with zero bits.

**Numbers in the computer**

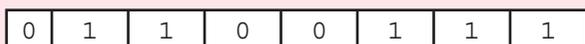
All data in the computer must be represented in binary form.

Consider a single byte used to represent a positive integer.

- the most significant bit position has place value of 128
- the least significant position has place value of a 'unit', that is 0 or 1.

**Progress check 1.01**

**1** What positive integer is this?



**2** A positive integer is represented using a single byte. What is the denary value?

- a** 0100 0001   **b** 1010 1010   **c** 1111 1111

**3** What is the eight-bit binary representation for these integers?

- a** 3                      **b** 89                      **c** 257

**4** Convert these hexadecimal numbers to denary:

- a** 1A                      **b** 10B

**5** Convert these hexadecimal numbers to 12-bit binary representations:

- a** 7D                      **b** 196                      **c** AEC

**Two's complement representation**

We need to be able to represent both positive and negative integers.

One (simple) method would be to use the most significant bit to act as a 'sign' bit (1 for a negative integer and 0 for a positive integer). This method is called 'sign and magnitude' but is not in our 9608 syllabus.

We shall use a representation – two's complement – which has a negative place value for the most significant

bit. For a two's complement presentation using a single byte the place values are as shown in Figure 1.02.

-128	64	32	16	8	4	2	u

Figure 1.02 Two's complement place values

**Example:**

Convert the following denary numbers to an eight-bit two's complement binary number.

1  $56 = 32 + 16 + 8$

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

2  $-125 = -128 + 3 = -128 + (2 + 1)$

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

3  $-17 = -128 + 111 = -128 + (64 + 32 + 8 + 4 + 2 + 1)$

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

**TIP** Note the method for a negative number. If its negative, we must have the '1' lot of '-128' – we then need to work out what positive number to add to it.

**Representing characters**

All data – including characters – must be represented in main memory, saved in the backing store and processed by a program as a number value. A coding system such as ASCII or Unicode will be used.

**LOOK FORWARD »**  
 ASCII and Unicode are discussed in Chapter 10, section 10.01.

**Binary-Coded Decimal (BCD)**

Binary-coded decimal is a binary representation which can be used for a positive denary integer.

Each digit of the denary number is represented in sequence with a group of four binary digits.

Example: Represent the denary integer 859 in BCD.

8	5	9
1000	0101	1001

So, 859 denary is 100001011001 as a BCD representation.

Early computers stored date and time values in the BIOS of the operating system using BCD representation. Some later games consoles including Atari and Sony PlayStation did likewise. However in 2010, the PlayStation software interpreted the final two digits of the date '10' (stored in BCD) as the hexadecimal number 16. The resulting date of 2016 made the console inoperable!

**1.02 Images**

**Bitmapped image**

A bitmap graphic is a rectangular grid built up from a number of **pixels**. A pixel is the smallest addressable picture element which can be represented. The term bitmap comes from the concept that the bit patterns which make up the file are 'mapped' to an area in the main memory. Each pixel will be a particular colour. Each pixel's colour will be represented as a binary pattern. The contents of the bitmap file will be this sequence of binary colour codes.

**TERMS** **pixel:** the smallest addressable picture element which can be represented.

There are several types of encoding and file formats for bitmap images:

- Monochrome: black and white pixels only
- 16 colour: 16 available colours for the pixels
- 256 colour: 256 possible colours
- 24-bit colour (or true colour) where millions of different colours are possible.

The encoding for each type can be worked out as shown in Table 1.03.

Bitmap encoding	Pixel representation	Explanation
Monochrome	1 bit	Only two colours needed (Black and white). One byte can store eight pixels.
16 colour	4 bits	Each byte can store two pixels.
256 colour	8 bits (1 byte)	Each byte stores one pixel.
24-bit colour	24 bits (3 bytes)	The number of different colours possible is $2^{24}$ (16, 777, 216).

**Table 1.03** Encodings for bitmap images

### TIP

These calculations are an application of the study of number systems in Chapter 1, section 1.01.

In addition to the pixel data, the bitmap file will have other data stored in a file header. The header data will give the size of the bitmap (width and height measured in pixels) and the type of bitmap (encoding)

Bitmaps have the drawback that they have a large file size. If an attempt is made to over-enlarge the bitmap with -editing software the individual pixels may become visible. This is called the staircase effect. Figure 1.03 shows an image of a mouse on the left and the same image after it has been enlarged – the individual pixels can clearly be seen.



**Figure 1.03** A bitmap and its enlarged version

The clarity with which a bitmap image is viewed on a monitor screen will depend on two factors:

- resolution of the image: the number of pixels per centimetre. A small image size made up from a large number of pixels will produce a sharper display.
- screen resolution: the number of pixels which can be viewed horizontally and vertically on the screen. A typical PC screen resolution is 1680 pixels × 1080 pixels. This is a key factor to consider when purchasing a monitor – what is the highest possible screen resolution?

## Vector graphics

A vector graphic is made up from a number of drawing objects. A vector graphic program such as Microsoft Visio or Corel Draw comes with a vast number of different objects organised into groups or 'shape libraries'.

Objects are organised into groups of shapes – the creator has selected a straight line from the 'Connectors' group and an LCD monitor from the 'Computer' group.

Objects have properties. These properties determine the size and appearance of each object. If an object is re-sized its properties are simply recalculated.

An example could be a network topology diagram where a library of networking shapes exists containing objects for a computer, file server, printer, etc. The user could quickly construct a network topology diagram.

The advantage of vector graphics is that changing the size of any object will not affect the quantity of the drawing's appearance. That is, the objects are scalable.

## Applications of bitmapped and vector graphics

Bitmapped graphics are used to:

- capture scanned images from a paper document.
- scan a photograph.

Vector graphics are used for:

- general line-drawing diagrams
- diagrams for specialist applications, such as flowcharting, object-oriented class diagrams, network topologies and any application where there is a specialist shapes library available.

A diagram using vector graphics software could be intended for inclusion in a word processor document. When completed it must be saved in one of the universally recognised file formats.

## 1.03 Sound

Sound is a key requirement for most software. Sound will be used for:

- sounding context-sensitive warning messages to the user
- the playback of music files, video and bit-streamed media content
- specialist applications, such as the reading of a text document to a visual impaired user.

A sound signal is an analogue signal. To be saved as data on the computer, the sound signal must be converted from an analogue to a digital signal. This will be done by some form of analogue-to-digital converter (ADC).

The sound will be sampled at a set time interval and these sample values form the binary values which form the sound file. The issues which affect the sound quality and the file size are:

- How many bits are used to encode each sampled value (the **sampling resolution**)
- How often the samples are taken, that is, how many values per second (the **sampling rate**)

The graph in Figure 1.04 illustrates the sampling rate. Samples are being taken every one millisecond; that is, 1000 samples will be taken every second.

This example used only eight bits to store each sample. Figure 1.05 shows the sampled data values stored in main memory from address 300 onwards.

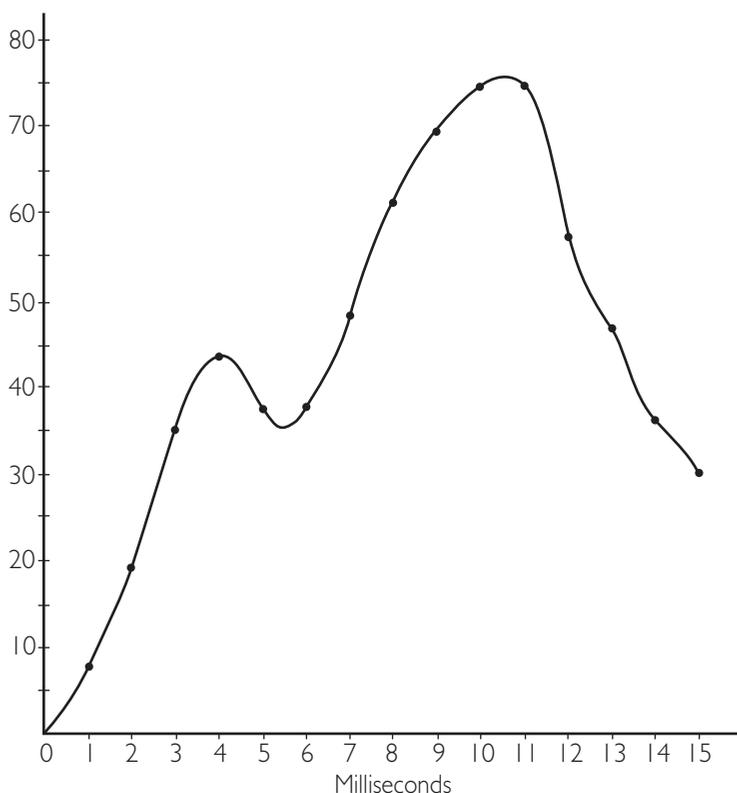


Figure 1.04 A graph of sound samples

300	301	302	303	304	305	306	307	308	309
8	20	35	44	38	38	48	61	69	75
					310	311	312	313	314
					75	57	45	36	29

Figure 1.05 Samples stored in memory

It should be apparent from Figure 1.05 that:

- If samples are taken more frequently, the quality of the sound wave will increase.
- If a larger number of bits is used to encode each sample, the sound resolution will increase.

Sound editing software is used for the recording of:

- Spoken word using a microphone
- The 'digitising' of an analogue sound source. An example could be the connection of a record turntable to the computer. The sound from a vinyl record is then recorded using the sound recording software

Editing features of the software would include:

- Cutting and pasting of sections of the recording
- Filtering out certain sounds. For example the 'clicks' on a scratched vinyl record
- Recording as a single (mono) channel or two channels (stereo)
- Normalising the recording level
- Export of the sound recording to a variety of file formats. For example MP3.

## 1.04 Video

Video is in widespread use on computers for recreational and educational use:

- YouTube is one of the most popular websites where users can post their own video content.
- Videos are an excellent medium for an explanation of the working of a piece of equipment or to provide a learning tutorial.

A video is a sequence of still photographic images which are displayed in sequence. The frequency with which they are displayed gives the appearance of continuous motion, and what is contained on individual frames is not apparent.

The frequency with which the frames are displayed is called the **frame rate**. A continuous effect to the human eye is achieved with a frame rate of 25 frames per second or higher.

### TERMS

**Frame rate:** the frequency with which video frames are displayed

## Progressive encoding

A system which stores the data for an entire frame and displays all the frame data at the same time is called 'progressive encoding'. This means the frame rate will be the number of pictures displayed per second. Traditional film uses progressive encoding.

## Interlaced encoding

The problem is that some devices, such as a television, are not designed to display all the frame data at the same time. The data from a single frame is encoded as two separate fields; one containing the data for the even numbered rows and the second frame has the data for the odd numbered rows. The term interlaced comes from the concept that the image is rendered by switching between the even field and the odd field. It follows that the rate of picture display is twice the frame rate.

With increasing demand for the display of video content through DVD players, set-top boxes and other home electronic devices there is still a need for interlaced encoded video format files.

The picture frames that make up interlaced fields have a correct order relative to each other:

- The spatial order shows which should be the odd or even field.
- The temporal order refers to a field or frame and which field represents an earlier moment in time.

If either one or both of these orders is incorrect the result of the playback will appear as jerky motion or blurred edges to content.

## 1.05 Compression techniques

Both sound and video files tend to have large file size. Techniques used which encode the data in a way which results in less bytes for the file are highly desirable.

Compression is the technique of reducing the size of a file without a significant loss in the later quality in the use of the file.

## Image compression techniques

### Run-length encoding (RLE)

Consider a bitmapped file of a photograph where over half of the pixels are the same pixel value, representing the blue sky. An alternative to saving (say) the 300 consecutive pixel values on a row could be to save a single copy of



## File formats

Over the years standards in the computing industry have emerged for image files (we have already mentioned .BMP, .PNG, .GIF and JPEG) and sound data. Popular sound formats include .WAV, .MPEG and .MP3.

Video – which is a combination of moving pictures and sound – requires its own industry standards. The detail about encoding methods used for this is outside the scope of our syllabus. The key issue is that there is correct synchronisation between the picture display and the accompanying sound commentary.

The current popular multimedia container formats include:

- AVI (standard Microsoft Windows container)
- MOV (standard QuickTime container)

- MP4 (standard container for MPEG-4 multimedia)
- Matroska (not standard for any codec or system, but it is an open standard).

The differences between container formats arise from issues such as:

- popularity: is the container format widely supported? This is the reason that the AVI format is still the most popular format.
- overheads: This refers to the difference in file size between two files with the same content in a different container. For a two-hour film, an AVI file may be up to 10 MB larger than a file in Matroska format.
- support for advanced codec functionality: Older formats, such as AVI, do not support new codec features, such as streaming media.

## Summary

- Numbers can be written using a binary, denary or hexadecimal base.
- Two's complement is a representation which allows both positive and negative integers to be represented.
- Binary-coded decimal (BCD) is a coding system used for positive integers.
- Images can be encoded as a bitmap, made up of a rectangular grid of pixels. The file header will contain data about the image: its height, width and the type of bitmap. Bitmap resolutions are monochrome, 16 colour, 256 colour and true colour. From the resolution and the dimensions, the file size can be calculated.
- Vector graphics are constructed using drawing objects selected from shape libraries provided by the software. Each object has a set of properties which are stored as part of the vector file.
- Sound is encoded as samples taken from the analogue source with a set sampling rate. The number of bits used to encode each sample (the sampling resolution) determines the sound quality.
- Video is made up of a sequence of image frames with an accompanying sound track. The encoding can be interlaced or progressive. Various multimedia formats are used commercially. These formats may use compression techniques to address spatial and temporal redundancy.
- Compression techniques use either 'lossy' or 'lossless'. One lossless technique is run-length encoding (RLE).

## Exam-style questions

- 1 Binary representation is used for many different data values.  
 Consider the binary pattern 1010 0110  
 What is its value if it represents:
- a an 8-bit two's complement integer? [1]
  - b an 8-bit sign and magnitude integer? [1]
  - c a hexadecimal number? [1]

*Cambridge International AS and A Level Computing 9691 Paper 33, Q2 a Nov 2012*

- 2 a i Convert the hexadecimal number 7A to denary. [1]  
 ii Convert the binary number 0101 1100 to hexadecimal. [1]  
 iii Why do computer scientists often write binary numbers in hexadecimal? [1]
- b The diagram shows a program loaded into main memory starting at memory address 7A Hex.

Main memory (contents shown in Hex.)	
Address	Hex.
7A	2150
7B	A351
7C	A552
7D	FFFF
⋮	
90	003C

How many bits are used for each main memory location? [1]

*Cambridge International AS and A Level 9691 Paper 31, Q3 b & c (i) Nov 2013*