1.1 Data representation

Binary systems

• recognise the use of binary numbers in computer systems
• convert denary numbers into binary and binary numbers into denary
• show understanding of the concept of a byte and how the byte is used to measure memory size
• use binary in computer registers for a given application (such as in robotics, digital instruments and counting systems)

1.1.2 Hexadecimal

• represent integers as hexadecimal numbers
• show understanding of the reasons for choosing hexadecimal to represent numbers
• convert positive hexadecimal integers to and from denary
• convert positive hexadecimal integers to and from binary
• represent numbers stored in registers and main memory as hexadecimal
• identify current uses of hexadecimal numbers in computing, such as defining colours in Hypertext Markup Language (HTML), Media Access Control (MAC) addresses, assembly languages and machine code, debugging

1.1.3 Data storage

• show understanding that sound (music), pictures, video, text and numbers are stored in different formats
• identify and describe methods of error detection and correction, such as parity checks, check digits, checksums and Automatic Repeat reQuests (ARQ)
• show understanding of the concept of Musical Instrument Digital Interface (MIDI) files, jpeg files, MP3 and MP4 files
• show understanding of the principles of data compression (lossless and lossy compression algorithms) applied to music/video, photos and text files

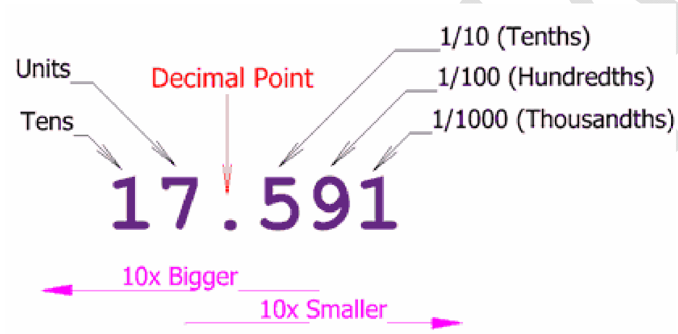
Number System

Decimals

To understand Binary and Hexadecimal numbers, it is best to know how Decimal Numbers work.

Every digit in a decimal number has a "position", and the **decimal point** helps us to know which position is which.

The position *just to the left* of the point is the "Units" position. Every position further to the left is 10 times bigger, and every position further to the right is 10 times smaller:



Now, this is **just a way of writing down a value**. Other ways include Roman Numerals, Binary, Hexadecimal, and more. You could even just draw dots on a sheet of paper!

The Decimal Number System is also called "Base 10″. Because it is based on the number 10.

And there are 10 symbols (0,1,2,3,4,5,6,7,8 and 9), but notice something interesting: **there is no symbol for "ten"**. "10" is actually two symbols put together, a "1" and a "0":

In decimal you count "0,1,2,3,4,5,6,7,8,9,…" but then you run out of symbols!

So you add **1 on the left** and then **start again at 0**: 10,11,12, …

Counting with Different Number Systems

But you don't **have to** use 10 as a "Base". You could use 2 ("Binary"), 16 ("Hexadecimal"), or any number you want to!

Example: In binary you count "0,1,…" but then you run out of symbols!

So you add **1 on the left** and then **start again at 0**: 10,11 …

So the general rule is:

Count up until just before the "Base", then start at 0 again, but first you add 1 to the number on your left.

Binary Numbers

Binary Numbers are just "Base 2″ instead of "Base 10″. So you start counting at 0, then 1, then you run out of digits … so you start back at 0 again, but increase the number on the left by 1.

Like this:

00**0**	
00**1**	
0**10**	there is no "2" in binary, so start back at 0 … … and add one to the number on the left
01**1**	
100	start back at 0 again, and add one to the number on the left… … but that number is already at 1 so it also goes back to 0 … … and 1 is added to the *next number* on the left
10**1**	
1**10**	etc…

Hexadecimal Numbers

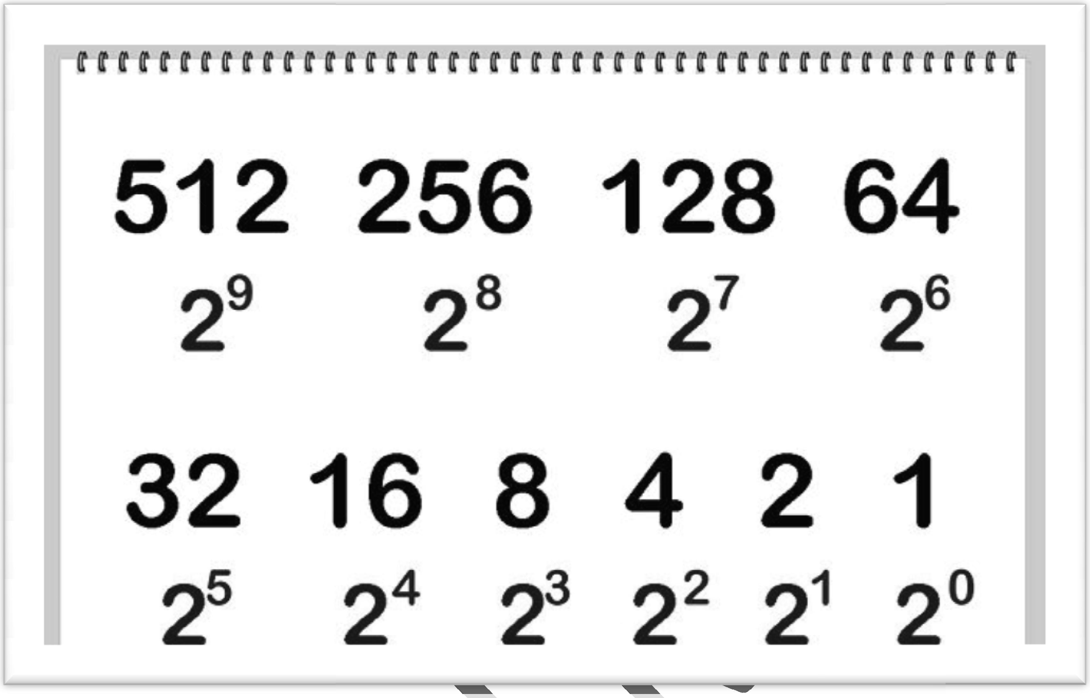
Hexadecimal numbers are interesting. There are 16 of them!

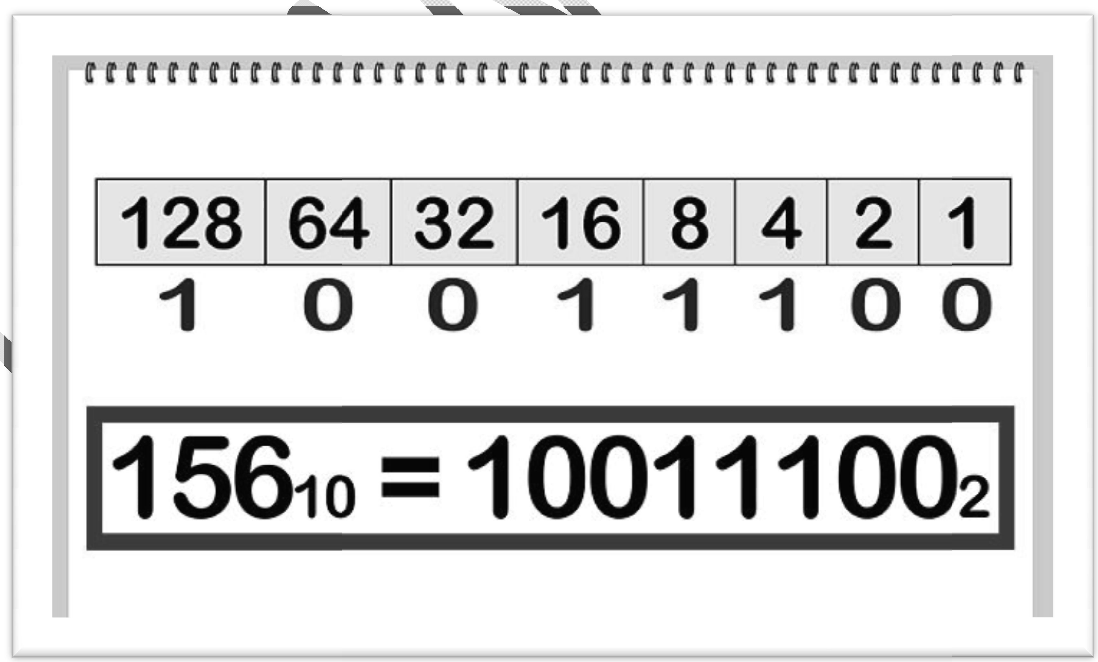
They look the same as the decimal numbers up to 9, but then there are the letters ("A',"B","C","D","E","F") in place of the decimal numbers 10 to 15.

So a single Hexadecimal digit can show 16 different values instead of the normal 10 like this:

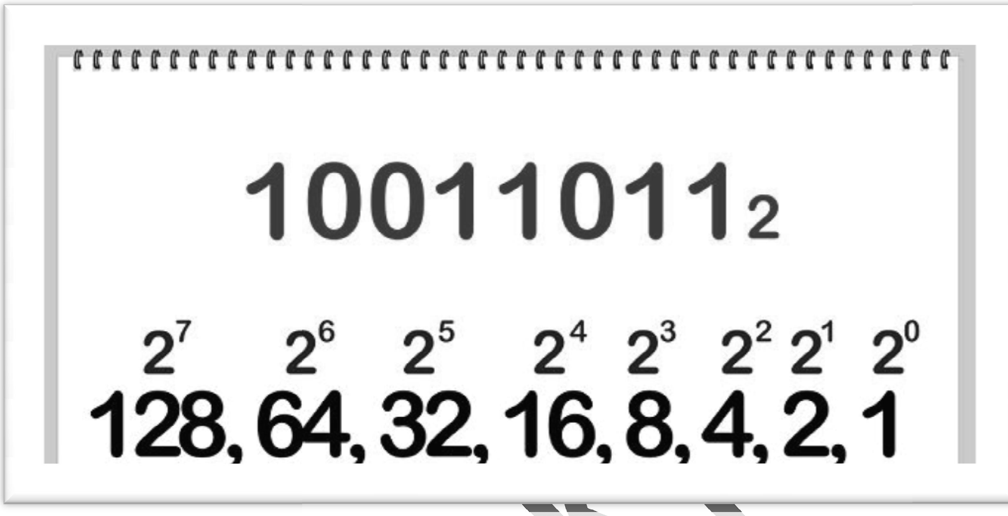
Decimal:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hexadecimal:	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

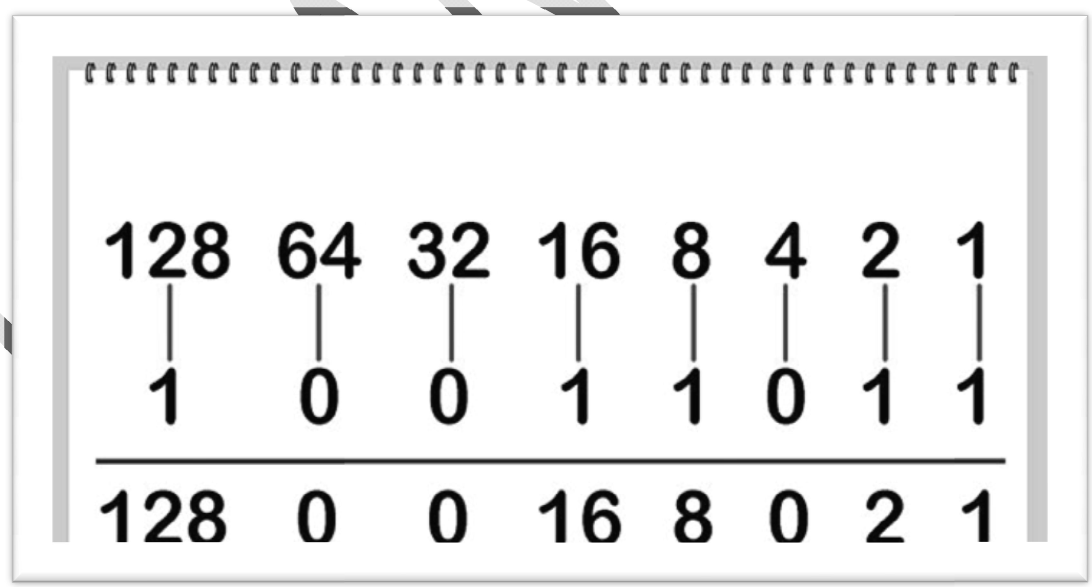
How to Convert from Decimal to Binary

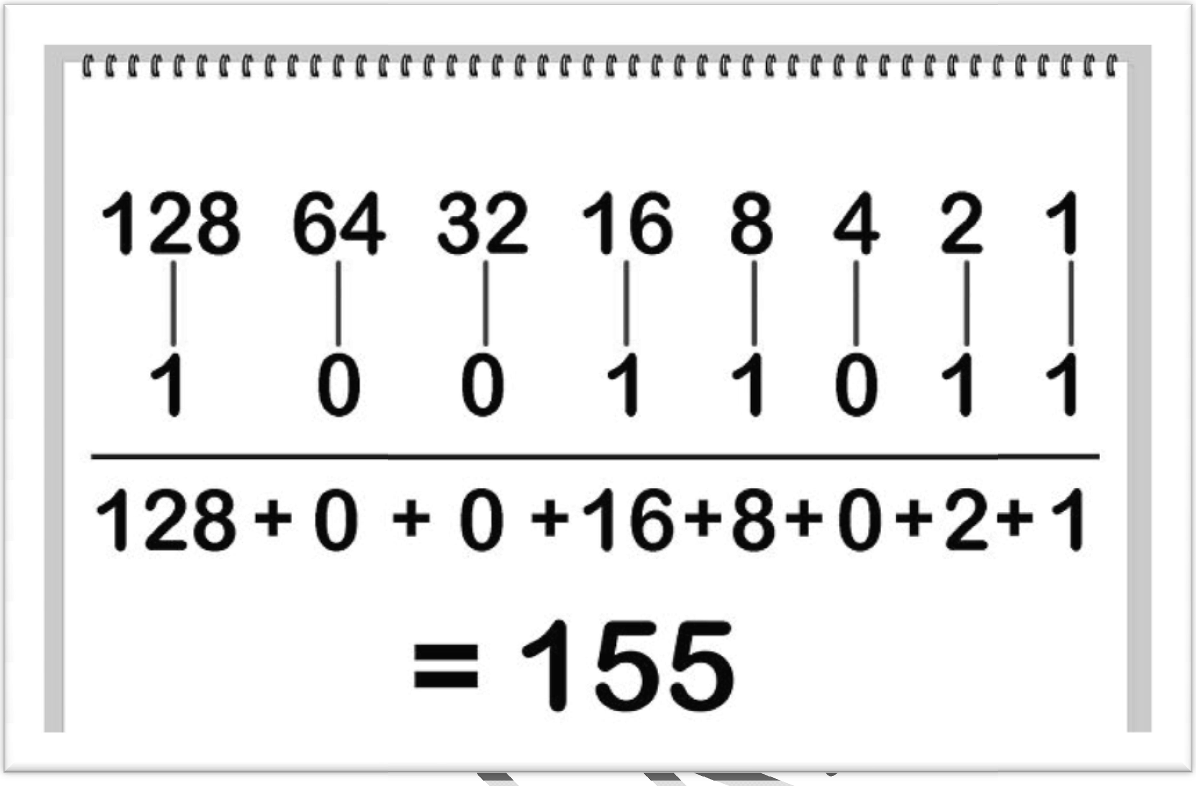


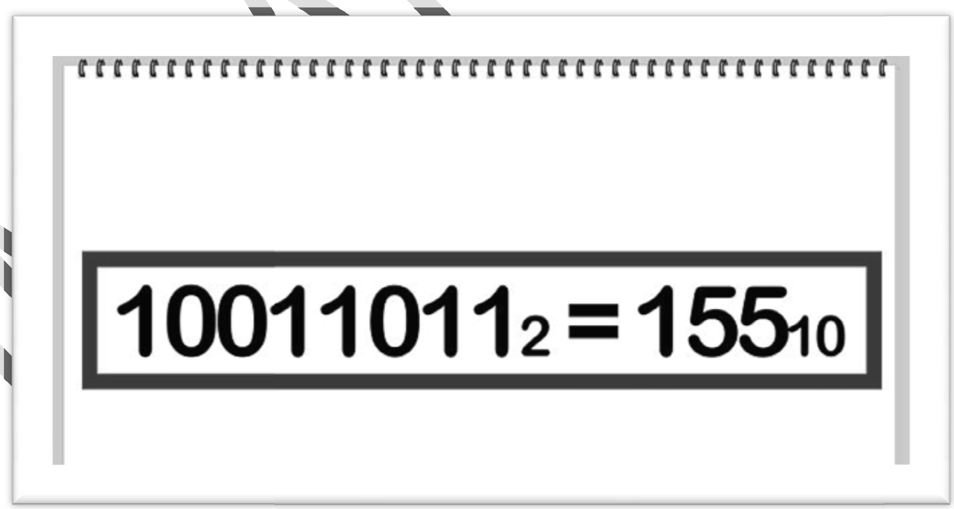


How to Convert from Binary to Decimal

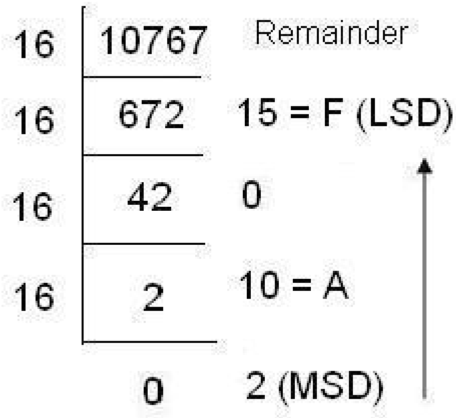




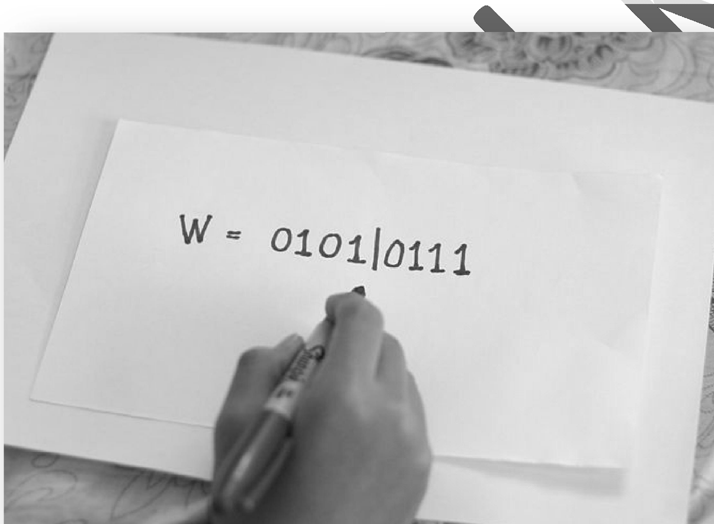

128 64 32 16 8 4 2 1
1 0 0 1 1 0 1 1
128 + 0 + 0 + 16 + 8 + 0 + 2 + 1
= 155

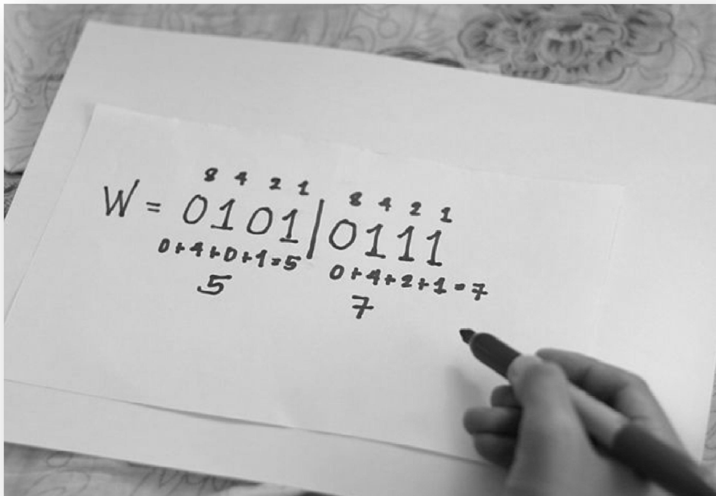

$10011011_2 = 155_{10}$

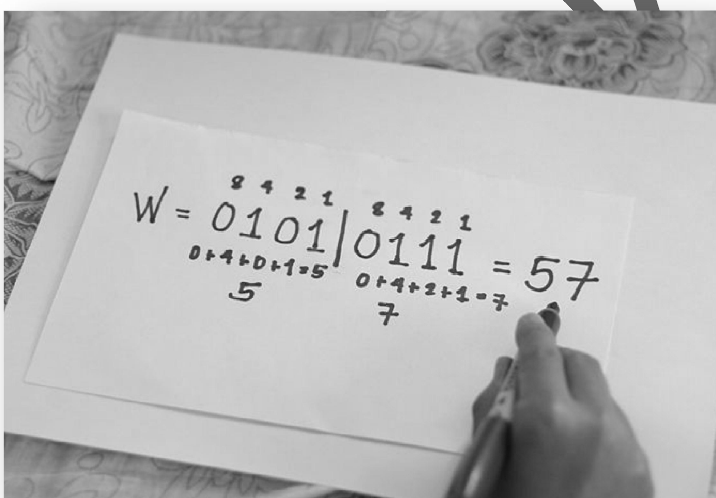
How to Convert from Decimal to Hexadecimal



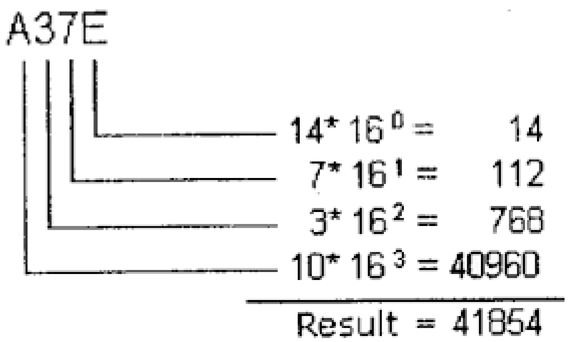
How to Convert from Binary to Hexadecimal



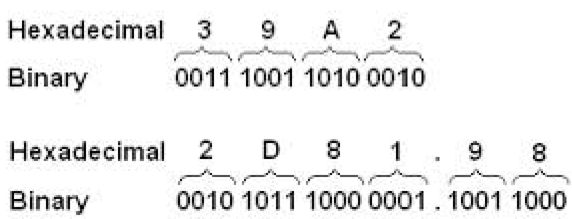

8 4 2 1
W = 0101 | 0111
8 4 2 1
0+4+0+1=5
5
0+4+2+1=7
7


8 4 2 1
W = 0101 | 0111 = 57
8 4 2 1
0+4+0+1=5
5
0+4+2+1=7
7

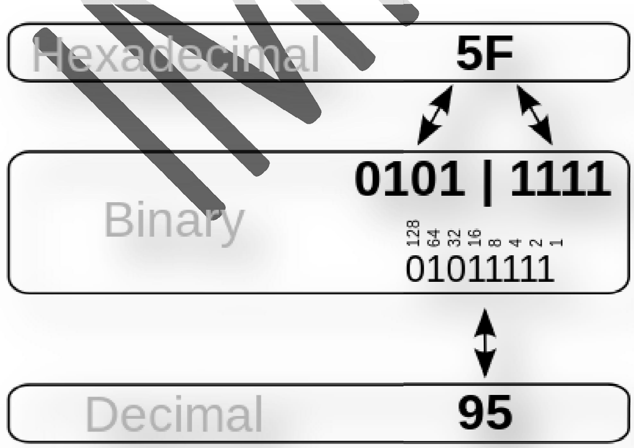
How to Convert from Hexadecimal to Decimal



How to Convert from Hexadecimal to Binary



How to Convert from Hexadecimal to Binary to Decimal



PARITY BITS

In communications, parity checking refers to the use of *parity bits* to check that data has been transmitted accurately. The parity bit is added to every data unit (typically seven or eight bits) that are transmitted. The parity bit for each unit is set so that all bytes have either an odd number or an even number of set bits.

Assume, for example, that two devices are communicating with even parity(the most common form of parity checking). As the transmitting device sends data, it counts the number of set bits in each group of seven bits. If the number of set bits is even, it sets the parity bit to 0; if the number of set bits is odd, it sets the parity bit to 1. In this way, every byte has an even number of set bits. On the receiving side, the device checks each byte to make sure that it has an even number of set bits. If it finds an odd number of set bits, the receiver knows there was an error during transmission.

The sender and receiver must both agree to use parity checking and to agree on whether parity is to be odd or even. If the two sides are not configured with the same *parity sense*, communication will be impossible.

Parity checking is the most basic form of error detection in communications. Although it detects many errors, it is not foolproof, because it cannot detect situations in which an even number of bits in the same data unit are changed due to electrical noise. There are many other more sophisticated protocols for ensuring transmission accuracy, such as MNP and CCITT V.42.

Parity checking is used not only in communications but also to test memory storage devices. Many PCs, for example, perform a parity check on memory every time a byte of data is read.

Bytes of data transferred using a serial cable are checked for errors at the receiving end using an even parity check.

Can these bytes of data pass the even parity check?

(a) 01010101

.... [1]

(b) 11001000

.... [1]

(c) How can any errors be corrected?

....

....

.... [2]

Question: An odd parity system receives the following messages:

(a) 110011

(b) 110101110100

(c) 1100010101010

Determine which groups, if any, are in error.

Automatic Repeat reQuest (ARQ)

Automatic Repeat reQuest (**ARQ**), also known as **Automatic Repeat Query**, is an error-control method for data transmission that uses acknowledgements (messages sent by the receiver indicating that it has correctly received a data frame or packet) and timeouts (specified periods of time allowed to elapse before an acknowledgment is to be received) to achieve reliable data transmission over an unreliable service. If the sender does not receive an acknowledgment before the timeout, it usually re-transmits the frame/packet until the sender receives an acknowledgment or exceeds a predefined number of re-transmissions.

File Format	Type of File	Filename Extension
Sun Systems sound	audio	.au
Windows sound	audio	.wav
Audio interchange	audio	.aiff, .aifc
RealAudio	audio stream	.ra, .ram
CompuServe GIF	graphics	.gif
JPEG	graphics	.jpg, .jpeg
Windows Bitmap	graphics	.bmp
QuickTime	video	.mov, .moov, .qt
Video for Windows	video	.avi
MPEG video	video	.mpg, .mpeg

Lossless and lossy compression

Lossless and lossy compression are terms that describe whether or not, in the compression of a file, all original data can be recovered when the file is uncompressed. With lossless compression, every single bit of data that was originally in the file remains after the file is uncompressed. All of the information is completely restored. This is generally the technique of choice for text or spreadsheet files, where losing words or financial data could pose a problem. The Graphics Interchange File (GIF) is an image format used on the Web that provides lossless compression.

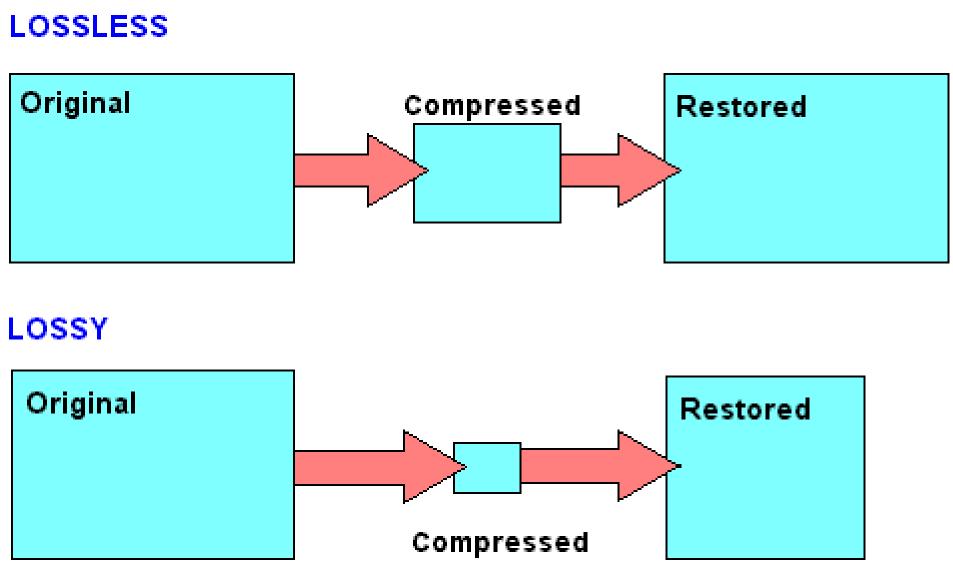
On the other hand, lossy compression reduces a file by permanently eliminating certain information, especially redundant information. When the file is uncompressed, only a part of the original information is still there (although the user may not notice it). Lossy compression is generally used for video and sound, where a certain amount of information loss will not be detected by most users. The JPEG image file, commonly used for photographs and other complex still images on the Web, is an image that has lossy compression. Using JPEG compression, the creator can decide how much loss to introduce and make a trade-off between file size and image quality.

Lossy compression is most commonly used to compress multimedia data (audio, video, and still images), especially in applications such as streaming media and internet telephony. By contrast, lossless compression is typically required for text and data files, such as bank records and text articles. In many cases it is advantageous to make a master lossless file that can then be used to produce compressed files for different purposes; for example, a multi-megabyte file can be used at full size to produce a full-page advertisement in a glossy magazine, and a 10 kilobyte lossy copy can be made for a small image on a web page.

Lossy file compression results in lost data and quality from the original version. Lossy compression is typically associated with image files, such as JPEGs, but can also be used for audio files, like MP3s or AAC files. The "lossyness" of an image file may show up as jagged edges or pixelated areas. In audio files, the lossyness may produce a watery sound or reduce the dynamic range of the audio.

Because lossy compression removes data from the original file, the resulting file often takes up much less disk space than the original. For example, a JPEG image may reduce an image's file size by more than 80%, with little noticeable effect. Similarly, a compressed MP3 file may be one tenth the size of the original audio file and may sound almost identical.

The keyword here is "almost." JPEG and MP3 compression both remove data from the original file, which may be noticeable upon close examination. Both of these compression algorithms allow for various "quality settings," which determine how compressed the file will be. The quality setting involves a trade-off between quality and file size. A file that uses greater compression will take up less space, but may not look or sound as good as a less compressed file. Some image and audio formats allow lossless compression, which does not reduce the file's quality at all.

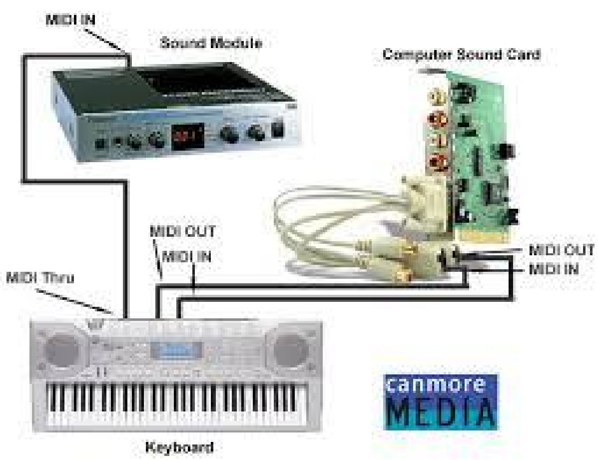


Domain Name System (DNS)

The **Domain Name System (DNS)** is a hierarchical distributed naming system for computers, services, or any resource connected to the Internet or a private network. It associates various information with domain names assigned to each of the participating entities. Most prominently, it translates domain names, which can be easily memorized by humans, to the numerical IP addresses needed for the purpose of computer services and devices worldwide. The Domain Name System is an essential component of the functionality of most Internet services because it is the Internet's primary directory service.

Musical Instrument Digital Interface

MIDI (/ˈmɪdi/; short for Musical Instrument Digital Interface) is a technical standard that describes a protocol, digital interface and connectors and allows a wide variety of electronic musical instruments, computers and other related devices to connect and communicate with one another.



MAY/JUNE SESSION 2002

Q1) Describe two ways that a scanner could be used to transfer paper-based documents to files stored on

May/June 2002

Q1) Errors can occur in data when it is being transmitted. A zero can be changed to a one and aone can be changed to a zero.

A method of checking the data is to write 1 in bit position 7 if there is an odd number of 1's in the other 7 bits, otherwise write 0 in bit position 7. For example the following bit pattern has a 1 in bit position 7 as the other bits contain an odd number of one's.

Bit position → Data →

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

(a) Complete the bit pattern below by writing the bit which should be in position 7.

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

In order to give a better chance of errors being discovered, a block of data can havevertical as well as horizontal checks. Complete the table by inserting the bits that shouldbe on the bottom row.

Bit position

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

(c) Suggest a reason why a bit could have changed when the data were transmitted.

....

....

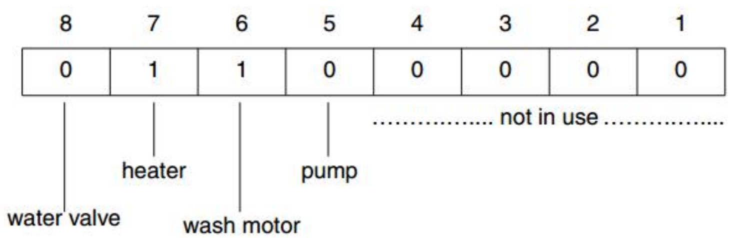
.... [1]

May/June 2005

Q2) A microprocessor controls the washing cycle of an automatic washing machine and givesoutput to the following devices:

• water valve
• heater
• wash motor
• pump

Control bits are sent to turn parts of the system on or off, i.e. 1 is on and 0 is off.



(a) State what is happening when the above bit pattern is set.

.... [1]

(b) Write down the bit pattern that would be set if the water has reached the correct level, the temperature is the required temperature, the clothes have been washed and thepump is now pumping the water out of the machine.

8	7	6	5	4	3	2	1

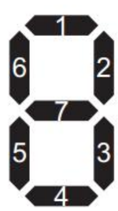
[1]

(c) State one other process that the microprocessor could control.

.... [1]

May/June 2007

Q3) A 7-segment display is used to indicate which floor a lift is on. Each segment is numberedas shown:



A byte is used to hold the data needed to light the correct segments. Bit 0 is always zero.
For example, 3 is represented by

and by

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

Bit Number

(a) If the lift is to stop at more than one floor, the data is held in successive bytes. Forexample:

FIRST BYTE:	0	0	0	0	1	1	1	0
SECOND BYTE:	1	1	0	1	1	0	1	0

Which floor numbers are stored in each byte?

First byte floor number.. [1]

Second byte floor number.. [1]

(b) What bit pattern is used to indicate Floor 2?

[1]

(c) The lift is travelling down to stop at Floors 5, 3 and 1. When it stops at Floor 5, apassenger gets in and presses the button for Floor 2.

How does the system ensure that the lift stops at Floors 3, 2 and 1 in that order?

....

....

....

.... [3]

May/June 2012

Q4) A vending machine has the choices shown below.

10	tea	11	with milk	12	with sugar	13	with milk and sugar
20	coffee	21	with milk	22	with sugar	23	with milk and sugar
30	hot chocolate	31	extra milk	32	extra sugar	33	with extra milk and extra sugar
40	cold water	41	hot water	42	fizzy water		
50	coke	51	orange	52	lemon		
60	chicken soup	61	tomato soup				

A customer uses a keypad to make their choice. Each number entered is represented in a6-bit binary register.
For example, key press 33 (hot chocolate with extra milk and extra sugar) is representedby

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

(i) If a customer chooses coffee with milk and sugar what is the key press?

[1]

(ii) How is it represented in the 6-bit register?

32	16	8	4	2	1

[2]

(b) If the 6-bit register shows.

1	0	1	0	0	1

What drink has the customer chosen?

....[1]

(c) A customer using the vending machine gets an error message after keying in their selection.

What could have caused this error message?

.... ………………..[1]

(d) It was decided to split the register so that each digit was represented by its own 3-bit register:

For example, 4 2 would now be represented as:

1	0	0
4	2	1

0	1	0
4	2	1

(i) What drink has been chosen if the 3-bit registers contain:

1	1	0
4	2	1

0	0	0
4	2	1

.... ………………..[1]

(ii) How would the *lemon* option be shown on **both** types of register?

4	2	1

4	2	1

32	16	8	4	2	1

[2]

(iii) What is the advantage of using two 3-bit registers rather than one 6-bit register?

....

.... [1]

Q5) (a) John has bought a 4 Gbyte MP3 player.

(You may assume: 1 byte = 8 bits, 1 Mbyte = 1024 kbytes and 1Gbyte = 1024 Mbytes)

(i) We can assume that each song lasts 3 minutes and is recorded at 128 kbps(kilobits per second).

How much memory is required per song?

....

(ii) Using your answer in (i), how many songs can be stored on John's MP3 player?

....

....

(b) John also bought a device for recording television programmes. It allows him to recorda programme at the same time as he is watching an earlier recording.

Describe how such a system would work.

....

....

May/June 2013

Q6) Some decorative lights are made up from a cluster of red, blue, green, yellow and white LEDs.

Each colour is represented by a binary code:

32	16	8	4	2	1	
1	0	0	0	0	0	red
0	1	0	0	0	0	blue
0	0	1	0	0	0	green
0	0	0	1	0	0	yellow
0	0	0	0	1	0	white
0	0	0	0	0	1	black (all lights off)

A 6-bit register, R1, stores the 1-values to represent a sequence of colours. Thus, if R1 contains:

0	1	0	1	0	1

this means the blue, yellow and black colour sequence is stored and displayed in that order.

The length of time each light is on is set by a binary value in another register, R2:

Thus

0	1	0

means each colour is on for 2 seconds.

(a) The two registers contain the following values.

What is the sequence of coloured lights and the timing for each colour?

R1

0	1	1	0	1	0

R2

1	1	1

Sequence of colours...

....

Timing.. ....

.... [2]

(b) What will the two registers contain if the coloured light sequence is red, green and black and the timing is 5 seconds?

R1

R2

(c) What is the problem with trying to display green, blue, red in that order?

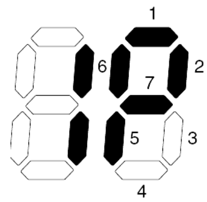
....

....

.... [2]

October/November 2003

Q7) Two 7 segment displays are used on a car dashboard to give information to the driver. Each segment is numbered as shown.



For example, the information 1P shown above is represented by:

	7	6	5	4	3	2	1	0
(1)	0	0	0	0	1	1	0	0

and by:

(2)	1	1	1	0	0	1	1	0

Bit 0 is always zero

(a) What is being displayed to the driver if bytes (1) and (2) are showing?

(1)	1	1	0	0	1	1	0	0	....
(2)	1	1	1	0	0	0	1	0	....

[2]

(b) What bit patterns must be used to show the information 0L?

(0)								
(L)								

(c) Most of the other information on the dashboard is in analogue form.

(i) State one advantage of displaying information in analogue form.

....

.... [1]

(ii) State one disadvantage of displaying information in analogue form.

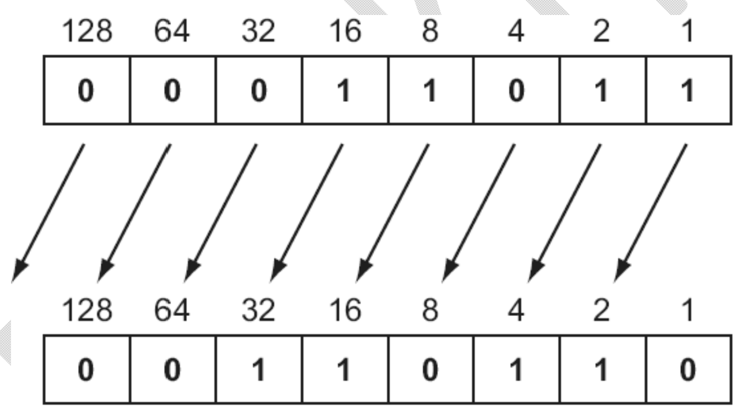
....

.... [1]

October/November 2013

Q8) A denary number can be represented as an 8-bit binary number. For example:

27 would be represented as:



All the bits in the binary number have now been shifted (moved) one place to the left.

What denary number does this now represent?

.... [1]

(b) What effect did the shift have on the original denary number?

.... [1]

(c) If the above binary number was shifted another one place to the left, what denary

number would it be equivalent to?

.... [1]

(c) (i) Represent the denary number 46 as an 8-bit binary number.

128	64	32	16	8	4	2	1

(ii) Shift this 8-bit binary number 2 places to the left.

What is the denary equivalent?

.... [1]

(iii) What problem would arise if you tried to shift this 8-bit binary number 3 places to the left?

.... [1]

....
..

NUMBER SYSTEM WORKSHEETS

Q1)

9 (a)

0							

[1]

(b)

1	0	0				1	1

one mark one mark [2]

(c) Electrical interference [1]

Q2)

(a) heater on and motor on/hot wash [1]
(b) 8 7 6 5 4 3 2 1
0 0 0 1 0 0 0 0[1]
(c) Any one from:
release door – via door switch
releasing powder at set intervals/fabric conditioner
drying/spinning
give error messages/beeps
stored programs for different washes e.g. cottons/woollens [1]

Q3) (a) 7
5 [2]
(b) 10110110 [1]
(c) Any three points from:
Notes lift is going down
Notes required floor is less than present floor
Sorts remaining numbers into descending order of floors [3]

Q4)

(a) key press:

2	3

represented by:

0	1	0	1	1	1

(b) drink chosen: hot water/41

(c) Any **one** from:
- incorrect number typed/keyed in
- not one of the accepted codes used
- code not recognised
- machine malfunction (e.g. no cups)

(d) **(i)** – chicken soup/60

(ii)

5

1	0	1

2

0	1	0

1	1	0	1	0	0

(iii) – gives an additional row of options
– now have 0 to 77 instead of only 0 to 63

Q5)
(a)
- 3 minutes = 180 seconds
- each song = 180 * 128 = 23 040 kbits
- number of bytes = 23 040/8 = 2880 kbyte
- = 2.8(125) Mbyte [2]

(ii)
- 4 Gbyte = 4 * 1024 = 4 096 Mbyte
- therefore, number of songs = 4 096/2.8125 = 1456 songs [2]

(b) Any three points from:

– uses hard disk/disk pack (2 to 5 disks)
– each disk surface has a R/W head
– use of read and write buffers
– R/W operation is faster than general data transfer rate …
– … therefore simultaneous read/write operations can occur
– description of how a DVD-RAM works
– concentric tracks allow R/W at the same time
– fast R/W operation [3]

Q6) (a) sequence of colours is:

- blue
- green
- white

Time delay is: 7 seconds [2]

(b)

1	0	1	0	0	1

1	0	1

<- - - - - - - 1 mark - - - - - - -> <- - 1 mark - -> [2]

(c) – sequence always starts from left to right

– so sequence would still be red, blue then green [2]

Q7)

(a)

- 4
- F [2]

(b)

(1) 01111110

(2) 01110000 [2]

(c)

(i) any one from:

drivers used to analogue instruments

readings are steadier

more accurate (because of infinite number of positions)

easier to see “trends” in read outs/easier to understand [1]

(ii) any one from:

not as easy to read as digital

needs to be interpreted by user

mechanical device more likely to break down/fail [1]