CAMBRIDGE

Cambridge University Press
978-0-521-33992-6 - 25 Cambridge Computer Science Texts: The Principles of Computer Networking
D. Russell
Excerpt
More information

1 Data Transmission

This chapter examines the ways in which data is transmitted along communications channels or lines. The subject of data transmission is a large and complex area of electronic engineering, and a chapter of this size is intended only to give a flavour of some of the principles involved. The intention is first to describe how characters can be represented as bits, and then see how bits can be transmitted down communications channels. We shall touch lightly on Fourier Analysis and the fundamental results of Nyquist and Shannon. After a look at this theoretical basis, the chapter proceeds to examine some of the more common methods of transmitting data along communications lines. This leads to a discussion of the types of errors that we can expect to encounter. It will also be the grounding to which we shall refer when we come to discuss some of the various sorts of shared medium in Chapter 3.

1.1 Character Representation

The simplest form of communication between computers and human beings is by means of characters that are assembled into text of some sort. Other means of communication, such as graphs, sound, mice, pointing, joysticks, colour, and so on have increasingly been employed. However, in simple computer communications, text is still the most important currency.

It is normal to represent text inside a computer by choosing a set of symbols, the character set, and allocating each one to a unique number. These unique numbers are then stored in locations called bytes. A byte is a group of eight bits within a computer, and is thus capable of representing a number in the range 0 to 255 inclusive. Bytes are sometimes called "octets".

Fig 1.1 shows the ASCII character set. The letters "ASCII" stand for the American Standard Code for Information Interchange. (There are many other national and international codes that are either identical to ASCII, or are very close and differ only in the representation of a few graphic symbols. There are also others, such as ISO 8859-1 that are more extensive, and will probably replace ASCII. However, at the time of writing, ASCII is the most widely used character set of the many we might have chosen.) We can see from the table that there are 128 characters arranged in eight columns of 16. The columns are numbered 0 to 7 from left to right, and the rows 0 to 15 from top to bottom. To

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE

Cambridge University Press
978-0-521-33992-6 - 25 Cambridge Computer Science Texts: The Principles of Computer Networking
D. Russell
Excerpt
More information

find the binary representation of a particular character, we multiply the column number by 16 and add the row number. Thus, for example, the letter "K" is in the 11th row of the 4th column. $4 \times 16 + 11 = 75$ (decimal) = 1001011 (binary). It is simpler to use hexadecimal numbering. Then "K" is in the B^{th} row of the 4th column, and we may immediately write the hexadecimal number as #4B. (#nn is the notation used in this book to denote hexadecimal numbers, base sixteen)

		0	1	2	3	4	5	6	7
0	0	NUL	DLE		0	@	P	`	p
1	1	SOH	DC1	!	1	A	Q	a	q
2	2	STX	DC2	"	2	B	R	b	r
3	3	ETX	DC3	#	3	C	S	c	s
4	4	EOT	DC4	$	4	D	T	d	t
5	5	ENQ	NAK	%	5	E	U	e	u
6	6	ACK	SYN	&	6	F	V	f	v
7	7	BEL	ETB	'	7	G	W	g	w
8	8	BS	CAN	(	8	H	X	h	x
9	9	HT	EM	)	9	I	Y	i	y
10	A	LF	SUB	*	:	J	Z	j	z
11	B	VT	ESC	+	;	K	[	k	{
12	C	FF	FS	,	<	L	\	l	|
13	D	CR	GS	-	=	M	]	m	}
14	E	SO	RS	.	>	N	^	n	~
15	F	SI	US	/	?	O	_	o	DEL

Fig 1.1. The ASCII Character Set

The most important thing that we need to observe is the technique of representing a set of characters by a corresponding set of small binary patterns that fit into bytes. Thus, depending upon the circumstances, a given bit pattern in a byte may be interpreted either as a small integer, or as a particular graphic character. The text of this book has all at various times been represented as ASCII characters within the memory of several different computers (as well as various other representations).

There are 94 printable characters, plus the space character, in columns 2 to 7. These are the ones with the single character entry in the table. There are another 33 locations that do not represent printable characters, but have mysterious two- or three-character "names". These are the so-called *control characters*. The control characters are codes that are reserved for special functions. Thus, #0D represents the *Carriage Return*—CR function, #0A represents the *Line Feed*—LF

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE

Cambridge University Press
978-0-521-33992-6 - 25 Cambridge Computer Science Texts: The Principles of Computer Networking
D. Russell
Excerpt
More information

function, and so on. These types of functions are directly related to the basic operations that are performed on simple character display devices.

For example, this text was originally typed at a simple video display terminal. On this terminal, the CR function returns the cursor to the left hand side of the screen, and the LF function moves the cursor down one line. These types of terminal and the simple means of communication that they employ are directly descended from the telex and telegraph machines that were developed over many years for sending character information over great distances. Computer communications has adopted and adapted this technology for its own uses.

In these early applications simple character machines communicated directly with each other over long distances. A typical teleprinter for example, consisted of a keyboard, a printing mechanism for printing on rolls of paper, and frequently a paper tape reader and punch for reading or punching holes in paper tape. In the early days of computing, programs were often prepared on rolls of paper tape on such machines, and then fed into the computer as ASCII encoded characters read directly off the paper tapes. Knowing this historical background makes it easier to appreciate some of the otherwise surprising aspects of character codes such as ASCII.

Codes that betray this history are such things as "ENQ"—enquire of the machine its ***answerback code***. This code was a unique code on such machines as the Teletype† which was set as radial legs on a bakelite wheel, and was sent in response to the ENQ as a way of automatic self-identification. DC1, DC2, DC3, DC4 were device control codes that were sent to a device with a paper tape reader or punch, and caused this device to switch on or off.

Of course, there are many machines without paper tape equipment and thus the meanings of some of the control codes change with time. For example, DC1 and DC3 can be used to control other ancillary devices, such as cassette tapes. Alternatively, they now almost universally have the meaning not of controlling the function of an ancillary device, but of controlling the flow of the actual data. Thus, it is now the common convention that DC3 means ***"Stop the flow of data to me for a while"***, and DC1 means ***"OK you can start sending again now"***. Common alternative names for these control characters are "X-off" and "X-on" respectively.

The punched paper tape background neatly explains the strange character "DEL" at position #7F. The paper tapes were often prepared manually, ready for being sent automatically some time later. Human typing is notoriously error prone, and it essential that some form of

† Teletype is a trademark of the Teletype corporation

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE

Cambridge University Press
978-0-521-33992-6 - 25 Cambridge Computer Science Texts: The Principles of Computer Networking
D. Russell
Excerpt
More information

error correction be available. The DEL character is just that. When preparing paper tape, the operator could correct mistakes by winding back the paper tape and pressing the DEL key. This punches out all the holes, and effectively erases or DELetes the character. Thus it is important to have DEL in this place. Similarly, pieces of paper tape need a piece of blank tape at the front and the end, and it is frequently useful to have blank sections in the middle. Such blank tape shows up a binary zeros, or #00—NUL. When dealing with character data it is normal to remove or ignore NUL characters. Another example of the evolution of meaning is that the DEL character now often means not that a character has been deleted, but that the last character typed should be deleted (i.e. that there has just been a typing error).

There is another vitally important feature of the ASCII character set, and that is that it uses only half the available values #00 to #7F. Characters are usually represented in an 8-bit byte, giving 256 possible values. The other 128 values from #80 to #FF are not allocated. Fundamentally ASCII is a seven bit code. The eighth bit, #80 is used as a *parity bit.* The function of this bit is for error checking, and we shall examine parity in some considerable detail in the next chapter. However, we cannot avoid mentioning it briefly now. The idea of parity is to make the number of bits in an eight-bit byte even. Thus, our character "K" has the 7-bit binary pattern 1001011. This has 4 one-bits, and thus the number is already even, so the parity bit is 0, and the full eight bit pattern is 01001011. On the other hand, "L" has the pattern 1001100. This has three one bits, and so the parity bit must be 1 to make even parity: 11001100.

What is the use of this? Well, the assumption is that if transmission errors occur, they will affect individual isolated bits. If an eight-bit byte is received, and an error has corrupted one bit of the data, then the parity will be wrong, and the character will be rejected as in error. These days, parity is seen as a weak error detection mechanism. This is mainly because errors tend to occur in groups or bursts, and parity is only about 50% successful in detecting such errors. We merely note now that ASCII characters are still normally transmitted with even parity, and leave further discussion of error checking until the next chapter. Note that we have said that ASCII characters are *transmitted* with parity. Normally the seven bit ASCII is stored within computers with the parity bit set to zero. The parity is generated as the character is transmitted, and checked as the character is received.

We are now in a position to see how a string of text might be stored within a machine. For example the eight characters "ABC:123." would be stored in eight bytes with a hexadecimal representation of #4142433A3132332E.

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE

Cambridge University Press
978-0-521-33992-6 - 25 Cambridge Computer Science Texts: The Principles of Computer Networking
D. Russell
Excerpt
More information

In this introduction we have only looked at one way in which character data is stored inside a computer. In practice, there are several other ways of representing characters. We shall return to look at them in Chapter 8. In the meantime we should note that because there are other much more complex ways of representing much more complex abstract types than character data, then we must be prepared to handle arbitrary bit patterns. While this may seem a little obvious, several of the early communications systems were only able to handle printable characters (columns 2 to 7 in the ASCII table). This led to all sorts of trouble as new applications made increasing demands on the simple technology.

1.2 Principles of Transmission

Information is transmitted through a medium by varying some physical property of the medium in one place, and measuring the resulting changes in physical property in another place. For example, the voltage across one end of a pair of wires can be varied, and the resulting voltage or current measured across the other end. If the medium is a radio channel, then an alternating current can be caused to flow in the transmitting antenna, and the resulting electromagnetic wave is detected by using a receiving antenna and attendant receiver. The radio signal can be just switched on and off, and the transmission could then be much like a Morse-code transmission. Alternatively, the amplitude of the signal could be varied (amplitude modulation as commonly employed in broadcast medium-wave radio), or its frequency (frequency modulation as used on VHF/FM broadcast radio), or the phase of the transmitted signal can be varied (as in some of the more sophisticated data transmission modems). Again, light may be transmitted down a filament of glass by repeated internal reflections. Light is just a rather high-frequency electromagnetic wave, like radio waves, and thus all the same modulation techniques can in principle be applied. However, the only practical form at present is the switching on and off of the beam of light. There are many other forms of transmission and modulation and we only offer these as illustrative examples.

Assume, then, that we are using a pair of wires to transmit the data. Let us assume that we are using some physical property that is a single valued function of time, $s(t)$. Let us suppose we wish to transmit the single character "\". In ASCII this is represented by the hexadecimal value #5C. This is the bit pattern 01011100. If we choose to represent a binary "1" by, say, $-5V$ (volts), and a binary "0" by $+5V$, then the graph of the voltage with time might look something like Fig 1.2. (The reader is cautioned that the question of data representation is bedevilled by varying conventions concerning whether a positive or a negative voltage represents a "1"-bit.) If we choose to transmit, say, 1000

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE

Cambridge University Press
978-0-521-33992-6 - 25 Cambridge Computer Science Texts: The Principles of Computer Networking
D. Russell
Excerpt
More information

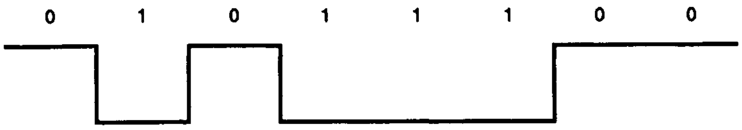

Fig 1.2. Signal Trace For The Character "\"

characters per second, then each bit would take $^1/_8{}^{th} mS$, or $125\mu S$.

Depending on the medium used, there is a whole wealth of detailed theoretical and practical knowledge that explains and predicts the precise way in which an input signal is transmitted along the medium to the far end. However, there is a large range of situations for which the following analysis applies (and, correspondingly, a small range of situations for which it does not).

At first sight, we might appear to need to study the various ways in which each individual waveform is transmitted. For example, a regular square wave will be transmitted and distorted in a different way from a regular saw tooth wave, which is in turn different to the distortion suffered by a regular sinusoidal wave. Indeed, each wave is distorted differently. We appear to be faced with an infinite problem—a separate study for each and every possible waveform.

1.3 Fourier Analysis

Fortunately, there is a simplification that can be made for linear channels. A linear channel is one for which the output signal is proportional to the input signal, or *linear* in the input signal. Thus, although a linear channel may distort an input signal, a saw tooth for instance, if it is fed with a signal twice as large, then the resulting signal will be exactly the same distorted shape, but twice as large. This is closely true for methods that are used for the transmission of analogue signals. Of course, this is no accident. The linear property is so desirable that it is a necessary part of analogue channels.

When a channel is linear, it is possible to analyse the signal propagation in terms of a *Fourier series*. Under certain constraints Fourier showed that any signal can be represented as the sum of an infinite series of sine waves. There are many standard texts that cover this way of representing signals. When a signal is represented as a Fourier series, there is a series of signals starting from low frequencies and increasing in frequency. In general, the slowly varying parts of a signal are represented by the low frequencies, and the quickly varying parts of a signal are represented by the high frequencies. Beyond a certain point the *amplitude* or size of the higher and higher frequency components starts to decrease. This is when the representation of the

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE

Cambridge University Press
978-0-521-33992-6 - 25 Cambridge Computer Science Texts: The Principles of Computer Networking
D. Russell
Excerpt
More information

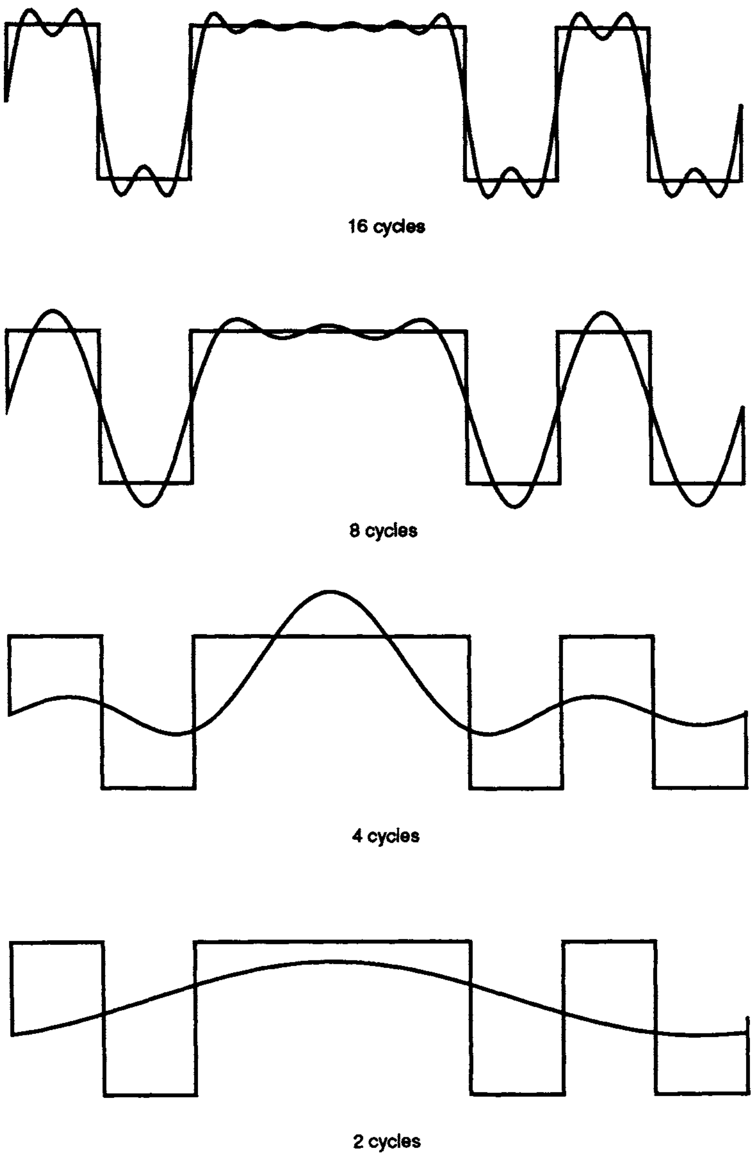

Fig 1.3. Fourier Approximation to a Bit Pattern

signal is essentially complete. The series is said to be *converging*.

For example, in Fig 1.3 we see the effect of including more and more frequencies in approximating more and more closely to a simple bit

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE

Cambridge University Press
978-0-521-33992-6 - 25 Cambridge Computer Science Texts: The Principles of Computer Networking
D. Russell
Excerpt
More information

pattern 01000101. The increasing frequencies are shown in terms of the number of cycles that the signal makes across the whole pattern of bits. One cycle corresponds to the base frequency, two cycles to a frequency of twice the basic frequency, and so on.

When we stop at the frequency corresponding to two cycles then the sum merely shows where the block of three consecutive zeros occurs. For four cycles we can just about make out all the bits, though perhaps with some difficulty. For eight cycles the Fourier sum shows the bits quite clearly, but the square edges are not very sharp. With 16 cycles the edges are becoming fairly sharp, and this sharpness increases as more and more frequencies are added.

With Fourier analysis, we can analyse our communications channel in terms of these single sinusoidal signals. A typical frequency response diagram is shown in Fig 1.4.

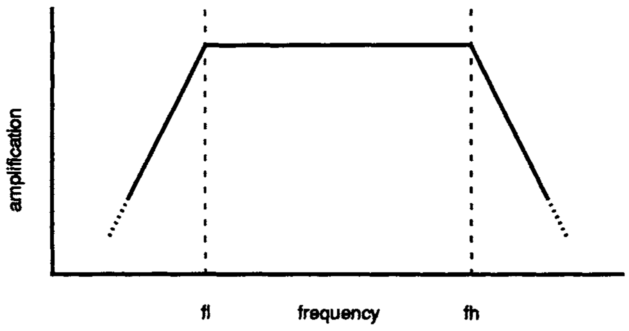

Fig 1.4. Frequency Response

Fig 1.4 shows a flat response over a large mid range which tails off for high and low frequencies beyond the central band. Such a graph is exhibited by audio channels such as telephone lines and "Hi-Fi" amplifiers. On telephone channels the flat response is roughly between 300 and 3100 Hz. This provides the familiar channel with some detectable distortion but with perfectly acceptable performance for human conversation. On the other hand, a "Hi-Fi" audio amplifier will have a much broader range covering as much as possible of the range of the human ear—$20 Hz$ up to $20,000 Hz$ is not unusual. In this way, an attempt is made to reduce the distortion of the reproduced signal to below that detectable by most listeners.

Square waves, such as our bit patterns, always have an infinite if converging series. Thus, some of the frequencies in the signal will be higher than fh for any real channel. In Fig 1.3 we showed the effect of truncating the series at certain values. As the number of terms included

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE

Cambridge University Press
978-0-521-33992-6 - 25 Cambridge Computer Science Texts: The Principles of Computer Networking
D. Russell
Excerpt
More information

becomes greater, the goodness of the fit improves. In practice, of course, the series will not be truncated, but the higher terms lying above fh will be progressively attenuated.

Conversely, the attenuation of low frequencies, below fl, will cause a reduction in the faithfulness in which slowly varying features are represented by the Fourier series. Just as wavelengths of the same order (and shorter) as a bit duration are required to represent single bits, so also wavelengths as long as (or longer than) a bit duration are required to represent consecutive runs of bits of the same value.

1.4 Noise

Apparently we could compensate for a cutoff frequency fh that is too low by inserting an electronic amplifier that amplified differentially above fh in an attempt to compensate for the fall off in the basic channel. However there are fundamental limits to the extent in which we can compensate by using this trick. The limit arises because whenever a signal is present, there is always a certain amount of noise.

Noise comes in many forms. Of course, there is the obvious interference caused by switches, motor cars, fluorescent lights and so on. This tends to be very erratic and bursty in nature. Such interference can often be prevented by shielding. Much more fundamental is the noise in the channel caused by the motion of the individual electrons. This is caused by the fundamental physics of the electronic circuits. All electronic circuits generate some of this noise, and any amplifier will amplify this noise along with the signal that is of interest.

Thus when a signal is injected into one end of a channel, a mixture of signal and noise comes out of the other end. Should the signal coming out of the channel be too low, then we can amplify it, but we will also amplify the noise. If the signal is weaker than the noise, then it will be lost in the noise as surely as the noise of a cough in the cheers of a football crowd. This ever-present background noise is of fundamental importance.

1.5 Limits of Data Transmission

There are two fundamental laws that describe the amount of data that can be transmitted on a communications line, and how this is limited both by frequency response and noise. Nyquist derived the result that limits the signalling rate of a perfectly noiseless channel, and Shannon extended this result to account for the effect of noise.

Nyquist looked at a perfectly noiseless channel, but one that was limited in bandwidth by a filter that let through frequencies up to a sharp cutoff (F Hz). If the received signal is then carefully examined, it is only necessary to sample the signal $2F$ times per second. More frequent

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE

Cambridge University Press
978-0-521-33992-6 - 25 Cambridge Computer Science Texts: The Principles of Computer Networking
D. Russell
Excerpt
More information

sampling would not add any more information. Of course, remember we are talking of perfect theoretical samples—the experimental error of an actual real measurement is ignored. The more frequent samples are pointless because they could only add information about signals with a frequency higher than F, and these, by definition, cannot get through the channel because of the perfectly sharp cutoff filter.

If the signal can have a number B different levels, then Nyquist's theorem states that the maximum data rate r is given by

$$r = 2F \log_2 B \quad \text{bits per second}$$

For example, if we have a perfect channel that can transmit signals of 0, 1, 2, 3 volts, and is limited to 1 kHz, then the maximum data rate is $2 \times 1000 \times \log_2 4 = 4,000$ bits per second. Note here most especially that there is a basic signalling rate which is $2F$. This is called the *baud rate*. However, each signal can be at one of four levels, giving the possibility of sending $log_2 4 = 2$ bits of information. This gives the *bit rate*, and *is not in general the same as the baud rate* despite the almost general sloppy confusion of the two even by many people and companies working in the communications field, who should know better.

Shannon extended Nyquist's work to allow for the presence of noise. For a perfectly noiseless channel, we have seen that the frequency cutoff limits the baud rate. However, the bit rate can be increased without limit by increasing the number of levels sent, B. We can see subjectively that when the levels become numerous enough (i.e. "close" enough) that the noise in the channel starts to interfere with them, then the noise will limit the value of B, and thus the achievable bit rate. The important measure here is the signal-to-noise ratio. This is the ratio of the signal power (not amplitude) to the noise power. If S is the signal power, and N the noise power, then the signal to noise ratio is S/N. Rather than a straight ratio, this is usually expressed in terms of the logarithmic measure, Bels, where

$$R = \log_{10}\left(\frac{S}{N}\right) \quad \text{Bels}$$

Thus $R = 1$ Bel means that $S = 10 \times N$, and $R = 2$ Bels means $S = 100 \times N$. Since this is obviously a somewhat powerful ratio, a more common measure is deciBels (dB). $R = 10dB$ means $S = 10 \times N$, and $R = 3dB$ means that $R \simeq 2 \times N$.

Shannon's major result is that for noisy channels with bandwidth f and signal-to-noise ratio S/N, the maximum data rate, R, is given by:

$$r = f \log_2\left(1 + \frac{S}{N}\right) \quad \text{bits per second}$$

© in this web service Cambridge University Press www.cambridge.org