APPENDIX A

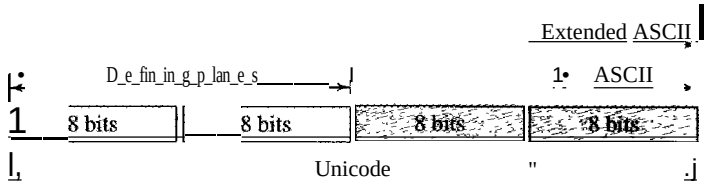
Unicode

Computers use numbers. They store characters by assigning a number for each one. The original coding system was called ASCII (American Standard Code for Information Interchange) and had 128 numbers (0 to 127) each stored as a 7-bit number. ASCII could satisfactorily handle lowercase and uppercase letters, digits, punctuation characters, and some control characters. An attempt was made to extend the ASCII character set to 8 bits. The new code, which was called Extended ASCII, was never internationally standardized.

To overcome the difficulties inherent in ASCII and Extended ASCII, the Unicode Consortium (a group of multilingual software manufacturers) created a universal encoding system to provide a comprehensive character set called Unicode.

Unicode was originally a 2-byte character set. Unicode version 3, however, is a 4-byte code and is fully compatible with ASCII and Extended ASCII. The ASCII set, which is now called *Basic Latin,* is Unicode with the upper 25 bits set to zero. Extended ASCII, which is now called Latin-I, is Unicode with the 24 upper bits set to zero. Figure A.I shows how the different systems are compatible.

Figure A.1 *Unicode compatibility*



A.I UNICODE

The prevalent code today is Unicode. Each character or symbol in this code is defined by a 32-bit number. The code can define up to 2^{32} (4,294,967,296) characters or symbols. The notation uses hexadecimal digits in the following format:

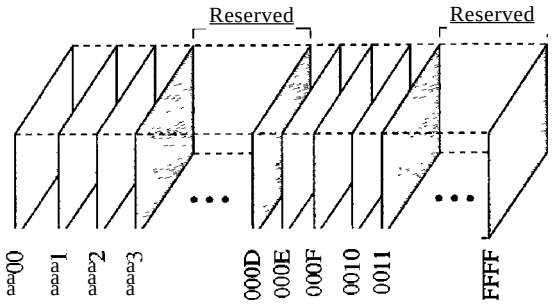
U-XXXXXXXX

Each X is a hexadecimal digit. Therefore, the numbering goes from U-00000000 to U-FFFFFFFF.

Planes

Unicode divides the available space codes into planes. The most significant 16 bits define the plane, which means we can have 65,535 planes. Each plane can define up to 65,536 character or symbols. Figure A.2 shows the structure of Unicode spaces and planes.

Figure A.2 *Unicode planes*



Plane 0000: Basic Multilingual Plane (BMP)
Plane 0001: Supplementary Multilingual Plane (SMP)
Plane 0002: Supplementary Ideographic Plane (SIP)
Plane OOOE: Supplementary Special Plane (SSP)
Plane OOOF: Private Use Plane (PUP)
Plane 0010: Private Use Plane (PUP)

Basic Multilingual Plane (BMP)

Plane 0000, the basic multilingual plane (BMP), is designed to be compatible with the previous 16-bit Unicode. The most significant 16 bits in this plane are all zeros. The codes are normally shown as U+XXXX with the understanding that XXXX defines only the least significant 16 bits. This plane mostly defines character sets in different languages with the exception of some codes used for control or other special characters. Table A.I shows the main classification of codes in plane 0000.

Table A.I *Unicode BMP*

Range	*Description*
A-Zone (Alphabetical Characters and Symbols)	
U+OOOO to U+OOFF	Basic Latin and Latin-l
U+OIOO to U+OIFF	Latin extended
U+0200 to U+02FF	IPA extension, and space modifier letters
U+0300 to U+03FF	Combining diacritical marks, Greek
U+0400 to U+04FF	Cyrillic
U+0500 to U+05FF	Armenian, Hebrew
U+0600 to U+06FF	Arabic

Table A.I *Unicode BMP (continued)*

Range	*Description*
U+0700 to U+08FF	Reserved
U+0900 to U+09FF	Devanagari, Bengali
U+OAOO to U+OAFF	Gumukhi, Gujarati
U+OBOO to U+OBFF	Oriya, Tamil
U+OCOO to U+OCFF	Telugu, Kannda
U+ODOO to U+ODFF	Malayalam
U+OEOO to U+OEFF	Thai, Lao
U+OFOO to U+OFFF	Reserved
U+1000 to U+lOFF	Georgian
U+1100 to U+llFF	Hangul Jamo
U+1200 to U+1DFF	Reserved
U+IEOO to U+1EFF	Latin extended additional
U+1FOO to U+1FFF	Greek extended
U+2000 to U+20FF	Punctuation, sub/superscripts, currency, marks
U+21 00 to U+21FF	Letterlike symbols, number forms, arrows
U+2200 to U+22FF	Mathematical operations
U+2300 to U+23FF	Miscellaneous technical symbols
U+2400 to U+24FF	Control pictures, OCR, and enclosed alphanumeric
U+2500 to U+25FF	Box drawing, block drawing, and geometric shapes
U+2600 to U+26FF	Miscellaneous symbols
U+2700 to U+27FF	Dingbats and Braille patterns
U+2800 to U+2FFF	Reserved
U+3000 to U+30FF	CJK symbols and punctuation, hiragana, katakana
U+3100 to U+31FF	Bopornfo, hangul jambo, cjk miscellaneous
U+3200 to U+32FF	Enclosed CJK letters and months
U+3300 to U+33FF	CJK compatibility
U+3400 to U+4DFF	Hangul
I-Zone (Ideographic Characters)	
U+4EOO to U+9FFF	CJK unified ideographic
O-Zone (Open)	
U+AOOO to U+DFFF	Reserved
R·Zone (Restricted Use)	
U+EOOO to U+F8FF	Private use
U+F900 to U+FAFF	CJK compatibility ideographs
U+FBOO to U+FBFF	Arabic presentation form-A

Table A.I *Unicode BMP (continued)*

Range	*Description*
U+FCOO to U+FDFF	Arabic presentation form-B
U+FEOO to U+FEFF	Half marks, small forms
U+FFOO to U+FFFF	Half-width and full-width forms

Supplementary Multilingual Plane (SMP)

Plane 0001, the supplementary multilingual plane (SMP), is designed to provide more codes for those multilingual characters that are not included in the BMP.

Supplementary Ideographic Plane (SIP)

Plane 0002, the supplementary ideographic plane (SIP), is designed to provide codes for ideographic symbols, symbols that primarily denote an idea (or meaning) in contrast to a sound (or pronunciation).

Supplementary Special Plane (SSP)

Plane OOOE, the supplementary special plane (SSP), is used for special characters.

Private Use Planes (PUPs)

Planes OOOF and 0010, private use planes (PUPs), are for private use.

A.2 ASCII

The American Standard Code for Information Interchange (ASCII) is a 7-bit code that was designed to provide code for 128 symbols, mostly in American English. Today, ASCII, or Basic Latin, is part of Unicode. It occupies the first 128 codes in Unicode (00000000 to 0000007F). Table A.2 contains the decimal, hexadecimal, and graphic codes (symbols) with an English interpretation, if appropriate. The codes in hexadecimal just define the two least significant digits in Unicode. To find the actual code, we prepend 000000 in hexadecimal to the code. The decimal code is just to show the integer value of each symbol when converted.

Table A.2 *ASCII Codes*

Decimal	*Hex*	*Symbol*	*Interpretation*
0	00	null	Null value
1	01	SOH	Start of heading
2	02	STX	Start of text
3	03	ETX	End of text
4	04	EaT	End of transmission

Table A.2 *ASCII Codes (continued)*

Decimal	*Hex*	*Symbol*	*Interpretation*
5	05	ENQ	Enquiry
6	06	ACK	Acknowledgment
7	07	BEL	Ring bell
8	08	BS	Backspace
9	09	HT	Horizontal tab
10	OA	LF	Line feed
11	OB	VT	Vertical tab
12	OC	FF	Fonn feed
13	OD	CR	Carriage return
14	OE	SO	Shift out
15	OF	SI	Shift in
16	10	DLE	Data link escape
17	11	DC1	Device control 1
18	12	DC2	Device control 2
19	13	DC3	Device control 3
20	14	DC4	Device control 4
21	15	NAK	Negative acknowledgment
22	16	SYN	Synchronous idle
23	17	ETB	End of transmission block
24	18	CAN	Cancel
25	19	EM	End of medium
26	1A	SUB	Substitute
27	IB	ESC	Escape
28	1C	FS	File separator
29	10	GS	Group separator
30	IE	RS	Record separator
31	IF	US	Unit separator
32	20	SP	Space
33	21	!	
34	22	"	Double quote
35	23	#	
36	24	$	
37	25	%	
38	26	&	
39	27	'	Apostrophe

Table A.2 *ASCII Codes (continued)*

Decimal	*Hex*	*Symbol*	*Interpretation*
40	28	(	
41	29	)	
42	2A	*	
43	2B	+	
44	2C	,	Comma
45	2D	-	Minus
46	2E		
47	2F	/	
48	30	0	
49	31	1	
50	32	2	
51	33	3	
52	34	4	
53	35	5	
54	36	6	
55	37	7	
56	38	8	
57	39	9	
58	3A		Colon
59	3B	,	Semicolon
60	3C	<	
61	3D	=	
62	3E	>	
63	3F	?	
64	40	@	
65	41	A	
66	42	B	
67	43	C	
68	44	D	
69	45	E	
70	46	F	
71	47	G	
72	48	H	
73	49	I	
74	4A	J	

Table A.2 *ASCII Codes (continued)*

Decimal	*Hex*	*Symbol*	*Interpretation*
75	4B	K	
76	4C	L	
77	4D	M	
78	4E	N	
79	4F	0	
80	50	P	
81	51	Q	
82	52	R	
83	53	S	
84	54	T	
85	55	U	
86	56	V	
87	57	W	
88	58	X	
89	59	y	
90	5A	Z	
91	5B	[	Open bracket
92	5C	\	Backs1ash
93	5D	]	Close bracket
94	5E	^	Caret
95	5F	_	Underscore
96	60	`	Grave accent
97	61	a	
98	62	b	
99	63	c	
100	64	d	
101	65	e	
102	66	f	
103	67	g	
104	68	h	
105	69	i	
106	6A	J	
107	6B	k	
108	6C	1	
109	6D	m	

Table A.2 *ASCII Codes (continued)*

Decimal	*Hex*	*Symbol*	*Interpretation*
110	6E	n	
111	6F	0	
112	70	p	
113	71	q	
114	72	r	
115	73	s	
116	74	t	
117	75	u	
118	76	v	
119	77	w	
120	78	x	
121	79	y	
122	7A	z	
123	7B	{	Open brace
124	7C	I	Bar
125	7D	}	Close brace
126	7E	~	Tilde
127	7F	DEL	Delete

SOllIe Properties of ASCII

ASCII has some interesting properties that we briefly mention here.

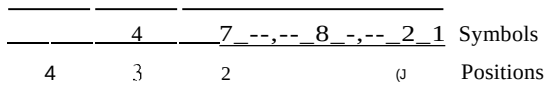
1. The first code (0) is the null character, which means the lack of any character.
2. The first 32 codes, 0 to 31, are control characters.
3. The space character, which is a printable character, is at position 32.
4. The uppercase letters start from 65 (A). The lowercase letters start from 97. When compared, uppercase letters are numerically smaller than lowercase letters. This means that in a sorted list based on ASCII values, the uppercase letters appear before the lowercase letters.
5. The uppercase and lowercase letters differ by only one bit in the 7-bit code. For example, character A is 1000001 (Ox4l) and character a is 110000I(Ox61). The difference is in bit 6, which is 0 in uppercase letters and 1 in lowercase letters. If we know the code for one case, we can easily find the code for the other by adding or subtracting 32 in decimal (Ox20 in hexadecimal), or we can just flip the sixth bit.
6. The uppercase letters are not immediately followed by lowercase letters. There are some punctuation characters in between.
7. Digits (0 to 9) start from 48 (Ox3). This means that if you want to change a numeric character to its face value as an integer, you need to subtract 48.

APPENDIXB

Numbering Systems

We use different numbering systems: base 10 (decimal), base 2 (binary), base 8 (octal), base 16 (hexadecimal), base 256, and so on. All the numbering systems examined here are positional, meaning that the position of a symbol in relation to other symbols determines its value. Each symbol in a number has a position. The position traditionally starts from 0 and goes to $n - 1$, where n is the number of symbols. For example, in Figure B.1, the decimal number 14,782 has five symbols in positions 0 to 4.

Figure B.1 *Positions and symbols in a number*



As we will see, the difference between different numbering systems is based on the *weight* assigned to each position.

B.1 BASE 10: DECIMAL

The base-10 or decimal system is the one most familiar to us in everyday life. All our terms for indicating countable quantities are based on it, and, in fact, when we speak of other numbering systems, we tend to refer to their quantities by their decimal equivalents. The term *decimal* is derived from the Latin stem *deci,* meaning 10. The decimal system uses 10 symbols to represent quantitative values: 0, 1,2, 3,4,5,6, 7,8, and 9.

Decimal numbers use 10 symbols: 0, 1,2,3,4,5,6,7,8, and 9.

Weights

In the decimal system, each weight equals 10 raised to the power of its position. The weight of the symbol at position 0 is 100 (1); the weight of the symbol at position 1 is 10^1 (10); and so on.

B.2 BASE 2: BINARY

The binary number system provides the basis for all computer operations. Computers work by turning electric current on and off. The binary system uses two symbols, 0 and 1, so it corresponds naturally to a two-state device, such as a switch, with 0 to represent the off state and 1 to represent the on state. The word *binary* derives from the Latin stem *bi,* meaning 2.

Binary numbers use two symbols: **0 and 1.**

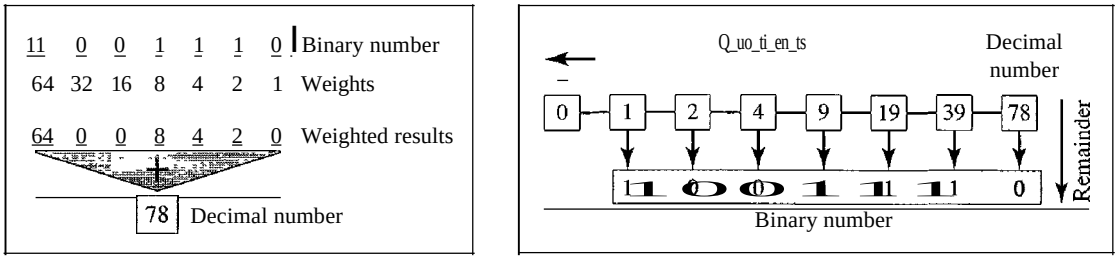
Weights

In the binary system, each weight equals 2 raised to the power of its position. The weight of the symbol at position 0 is 20 (1); the weight of the symbol at position 1 is 2^1 (2); and so on.

Conversion

Now let us see how we can convert binary to decimal and decimal to binary. Figure B.2 show the two processes.

Figure B.2 *Binary-to-decimal and decimal-to-binary conversion*



a. Binary to decimal

b. Decimal to binary

To convert a binary number to decimal, we use the weights. We multiply each symbol by its weight and add all the weighted results. Figure B.2 shows how we can change binary 1001110 to its decimal equivalent 78.

A simple division trick gives us a convenient way to convert a decimal number to its binary equivalent, as shown in Figure B.2. To convert a number from decimal to binary, divide the number by 2 and write down the remainder (1 or 0). That remainder is the least significant binary digit. Now, divide the quotient of that division by 2 and

write down the new remainder in the second position. Repeat this process until the quotient becomes zero.

B.3 BASE 16: HEXADECIMAL

Another system used in this text is base 16. The term *hexadecimal* is derived from the Greek term *hexadec,* meaning 16. The hexadecimal number system is convenient for identifying a large binary number in a shorter form. The hexadecimal system uses 16 symbols: 0, 1, ... , 9, A, B, C, D, E, and F. The hexadecimal system uses the same first 10 symbols as the decimal system, but instead of using 10, 11, 12, 13, 14, and 15, it uses A, B, C, D, E, and F. This prevents any confusion between two adjacent symbols.

Hexadecimal numbers use 16 symbols: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F.

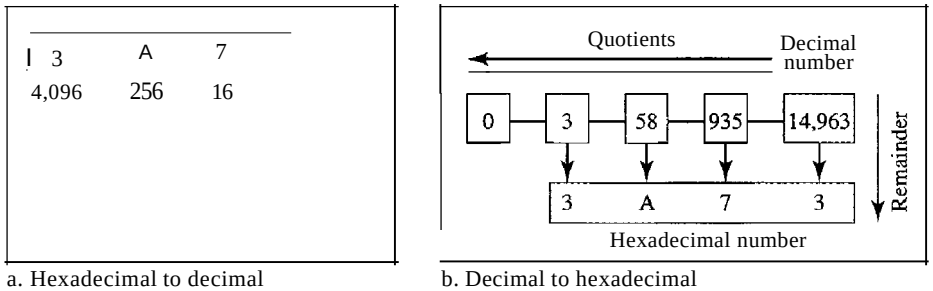
Weights

In the hexadecimal system, each weight equals 16 raised to the power of its position. The weight of the symbol at position 0 is 16^0 (1); the weight of the symbol at position 1 is 16^1 (16); and so on.

Conversion

Now let us see how we can convert hexadecimal to decimal and decimal to hexadecimal. Figure B.3 show the two processes.

Figure B.3 *Hexadecimal-to-decimal and decimal-to-hexadecimal conversion*



a. Hexadecimal to decimal

b. Decimal to hexadecimal

To convert a hexadecimal number to decimal, we use the weights. We multiply each symbol by its weight and add all the weighted results. Figure B.3 shows how hexadecimal Ox3A73 is transformed to its decimal equivalent 14,963.

We use the same trick we used for changing decimal to binary to transform a decimal to hexadecimal. The only difference is that we divide the number by 16 instead of 2. The figure also shows how 14,963 in decimal is converted to hexadecimal Ox3A73.

A Comparison

Table B.1 shows how systems represent the decimal numbers 0 through 15. As you can see, decimal 13 is equivalent to binary 1101, which is equivalent to hexadecimal D.

Table B.I *Comparison of three systems*

Decimal	*Binary*	*Hexadecimal*
0	0	0
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7

Decimal	*Binary*	*Hexadecimal*
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

B.4 BASE 256: IP ADDRESSES

One numbering system that is used in the Internet is base 256. IPv4 addresses use this base to represent an address in dotted decimal notation. When we define an IPv4 address as 131.32.7.8, we are using a base-256 number. In this base, we could have used 256 unique symbols, but remembering that many symbols and their values is burdensome. The designers of the IPv4 address decided to use decimal numbers 0 to 255 as symbols and to distinguish between the symbols, a *dot* is used. The dot is used to separate the symbols; it marks the boundary between the positions. For example, the IPv4 address 131.32.7.8 is made of the four symbols 8, 7, 32, and 131 at positions 0, 1,2, and 3, respectively.

IPv4 addresses use the base-256 numbering system.

The symbols in 1Pv4 are decimal numbers between 0 and 255;
the separator is a dot.

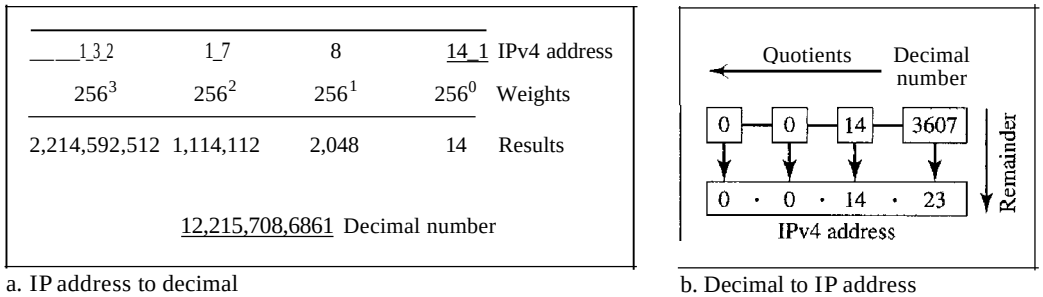
Weights

In base 256, each weight equals 256 raised to the power of its position. The weight of the symbol at position 0 is 256^0 (1); the weight of the symbol at position 1 is 256^1 (256); and so on.

Conversion

Now let us see how we can convert hexadecimal to decimal and decimal to hexadecimal. Figure B.4 show the two processes.

Figure B.4 *IPv4 address to decimal transformation*



a. IP address to decimal

b. Decimal to IP address

To convert an IPv4 address to decimal, we use the weights. We multiply each symbol by its weight and add all the weighted results. The figure shows how the IPv4 address 131.32.7.8 is transformed to its decimal equivalent.

We use the same trick we used for changing decimal to binary to transform a decimal to an IPv4 address. The only difference is that we divide the number by 256 instead of 2. However, we need to remember that the IPv4 address has four positions. This means that when we are dealing with an IPv4 address, we must stop after we have found four values. Figure *BA* shows an example for an IPv4 address.

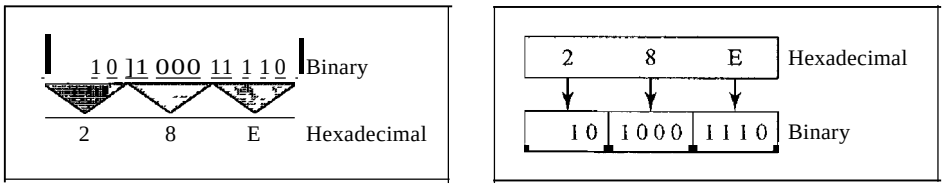
B.5 OTHER CONVERSIONS

There are other transformations such as base 2 to base 16 or base 16 to base 256. It is easy to use base 10 as the intermediate system. In other words, to change a number from binary to hexadecimal we first change the binary to decimal and then change the decimal to hexadecimal. We discuss some easy methods for common transformations.

Binary and Hexadecimal

There is a simple way to convert binary to hexadecimal and vice versa as shown in Figure B.5.

Figure B.5 *Transformation from binary to hexadecimal*



a. Binary to hexadecimal

b. Hexadecimal to binary

To change a number from binary to hexadecimal, we group the binary digits from the right by fours. Then we convert each 4-bit group to its hexadecimal equivalent,

using Table B.1. In the figure, we convert binary 1010001110 to hexadecimal Ox28E. To change a hexadecimal number to binary, we convert each hexadecimal digit to its equivalent binary number, using Table B.1, and concatenate the results. In Figure B.5 we convert hexadecimal Ox28E to binary.

Base 256 and Binary

To convert a base 256 number to binary, we first need to convert the number in each position to an 8-bit binary group and then concatenate the groups. To convert from binary to base 256, we need to divide the binary number into groups of 8 bits, convert each group to decimal, and then insert separators (dots) between the decimal numbers.

APPENDIXC

Mathematical Review

In this appendix, we review some mathematical concepts that may help you to better understand the topics covered in the book. Perhaps the most important concept in data communications is signals and their representation. We start with a brief review of trigonometric functions, as discussed in a typical precalculus book. We then briefly discuss Fourier analysis, which provides a tool for the transformation between the time and frequency domains. We finally give a brief treatment of exponential and logarithmic functions.

C.l TRIGONOMETRIC FUNCTIONS

Let us briefly discuss some characteristics of the trigonometric functions as used in the book.

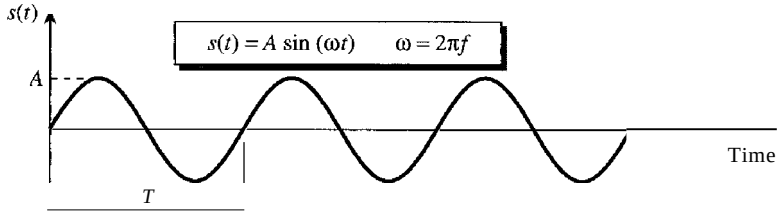
Sine Wave

We can mathematically describe a sine wave as

$$set) = A\sin(2\pi ft) = A\sin\left(\frac{2\pi}{T}t\right)$$

where s is the instantaneous amplitude, *A* is the peak amplitude, *f* is the frequency, and *T* is the period (phase will be discussed later). Figure C.l shows a sine wave.

Figure C.I *A sine wave*



Note that the value of 21tfis called the radian frequency and written as ω (omega), which means that a sine function can written as *set)* =*A* sin (ω*t*).

Example C.1

Find the peak value, frequency, and period of the following sine waves.

a. *set)* = 5 sin (10m)

b. *set)* = sin *(lOt)*

Solution

a. Peak amplitude: $A = 5$
Frequency: $10\pi = 2\pi f$, so $f = 5$
Period: $T = 1/f = 1/5$ s

b. Peak amplitude: $A = 1$
Frequency: $10 = 2\pi f$, so $f = 10/(2\pi) = 1.60$
Period: $T = 1/f = 1/1.60 = 0.628$ s

Example C.2

Show the mathematical representation of a sine wave with a peak amplitude of 2 and a frequency of 1000 Hz.

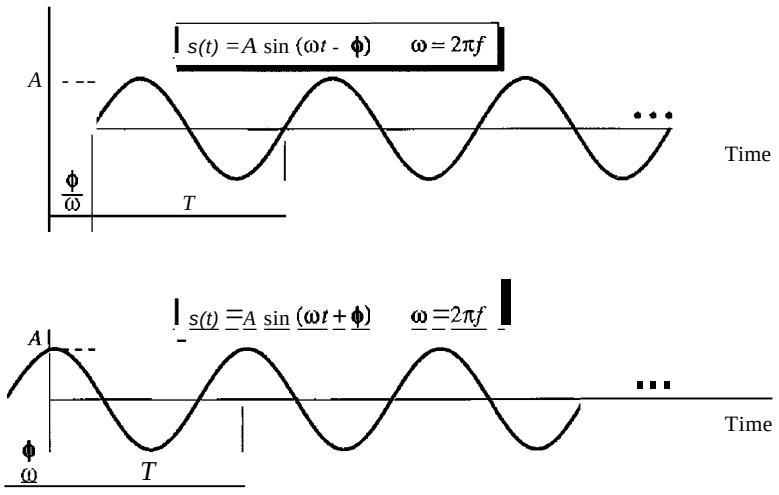
Solution

The mathematical representation is *set)* = 2 sin *(20001tt).*

Horizontal Shifting (Phase)

All the sine functions we discussed so far have an amplitude of value 0 at the origin. What if we shift the signal to the left or to the right? Figure C.2 shows two simple sine waves, one shifted to the right and one to the left.

Figure C.2 *Two horizontally shifted sine waves*



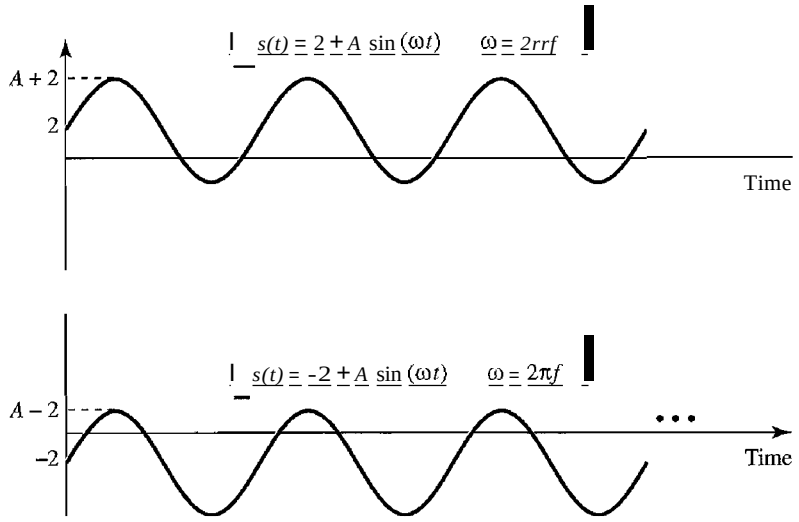
When a signal is shifted to the left or right, its first zero crossing will be at a point in time other than the origin. To show this, we need to add or subtract another constant to ωt, as shown in the figure.

Shifting a sine wave to the left or right is a positive or negative **shift,** respectively.

Vertical Shifting

When a sine wave is shifted vertically, a constant is added to the instantaneous amplitude of the signal. For example, if we shift a sine wave 2 units of amplitude upward, the signal becomes *set)* = 2 + sin (rot); if we shift it 2 units of amplitude downward, we have *set)* = -2 + sin (rot). Figure C.3 shows the idea.

Figure C.3 *Vertical shifting of sine waves*



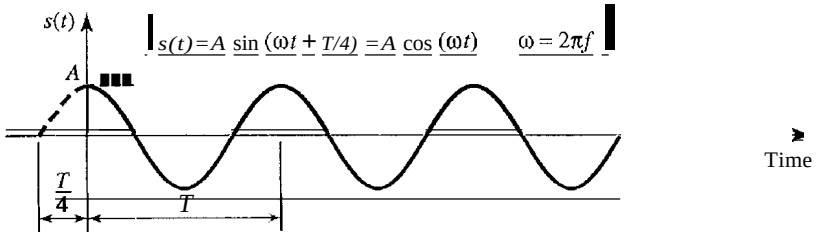
Cosine Wave

If we shift a sine wave *T/2* to the left, we get what is called a cosine wave (cos).

$$A \sin(\omega t + \pi/2) = A \cos(\omega t)$$

Figure C.4 shows a cosine wave.

Figure C.4 *A cosine wave*



Other Trigonometric Functions

There are many trigonometric functions; two of the more common are $\tan(\omega t)$ and $\cot(\omega t)$. They are defined as $\tan(\omega t) = \sin(\omega t)/\cos(\omega t)$ and $\cot(\omega t) = \cos(\omega t)/\sin(\omega t)$. Note that tan and cot are the inverse of each other.

Trigonometric Identities

There are several identities between trigonometric functions that we sometimes need to know. Table C.1 gives these identities for reference. Other identities can be easily derived from these.

Table C.1 *Some trigonometric identities*

Name	*Formula*
Pythagorean	$\sin^2 x + \cos^2 x = 1$
Even/odd	$\sin(-x) = -\sin(x)$ $\quad \cos(-x) = \cos(x)$
Sum	$\sin(x+y) = \sin(x)\cos(y) + \cos(x)\sin(y)$ $\cos(x+y) = \cos(x)\cos(y) - \sin(x)\sin(y)$
Difference	$\sin(x-y) = \sin(x)\cos(y) - \cos(x)\sin(y)$ $\cos(x-y) = \cos(x)\cos(Y) + \sin(x)\sin(y)$
Product to sum	$\sin(x)\sin(y) = 1/2\,[\cos(x-y) - \cos(x+y)]$ $\cos(x)\cos(y) = 1/2\,[\cos(x-y) + \cos(x+y)]$ $\sin(x)\cos(y) = 1/2\,[\sin(x+y) + \sin(x-y)]$ $\cos(x)\sin(y) = 1/2\,[\sin(x+y) - \sin(x-y)]$

C.2 **FOURIER** ANALYSIS

Fourier analysis is a tool that changes a time-domain signal to a frequency-domain signal and vice versa.

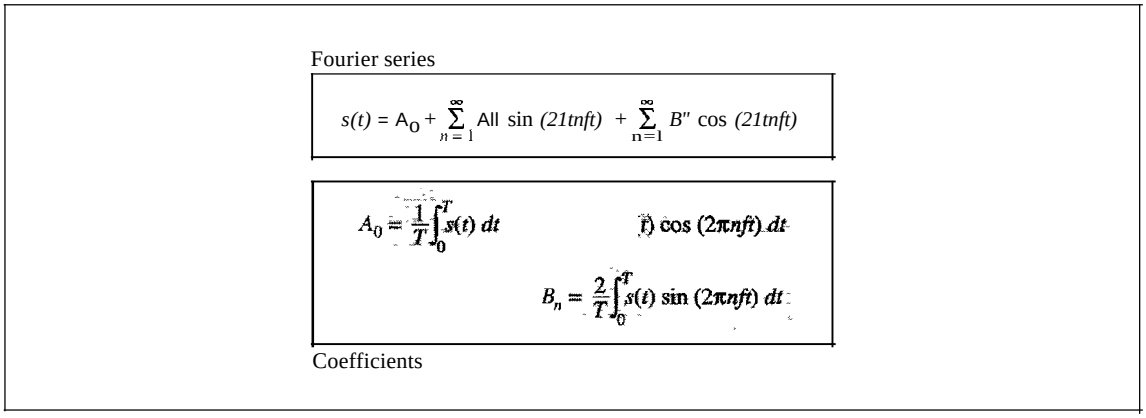
Fourier Series

Fourier proved that a composite periodic signal with period T (frequency f) can be decomposed into a series of sine and cosine functions in which each function is an integral harmonic of the fundamental frequency f of the composite signal. The result is called the Fourier series. In other words, we can write a composite signal as shown in Figure C.S. Using the series, we can decompose any periodic signal into its harmonics. Note that A_0 is the average value of the signal over a period, An is the coefficient of the nth cosine component, and B_n is the coefficient of the *nth* sine component.

Example C.3

Let us show the components of a square wave signal as seen in Figure C.6. The figure also shows the time domain and the frequency domain. According to the figure, such a square wave signal has only An coefficients. Note also that the value of $A_0 = 0$ because the average value of the signal is 0; it is oscillating above and below the time axis. The frequency domain of the signal is discrete;

Figure C.S *Fourier series and coefficients oftenns*

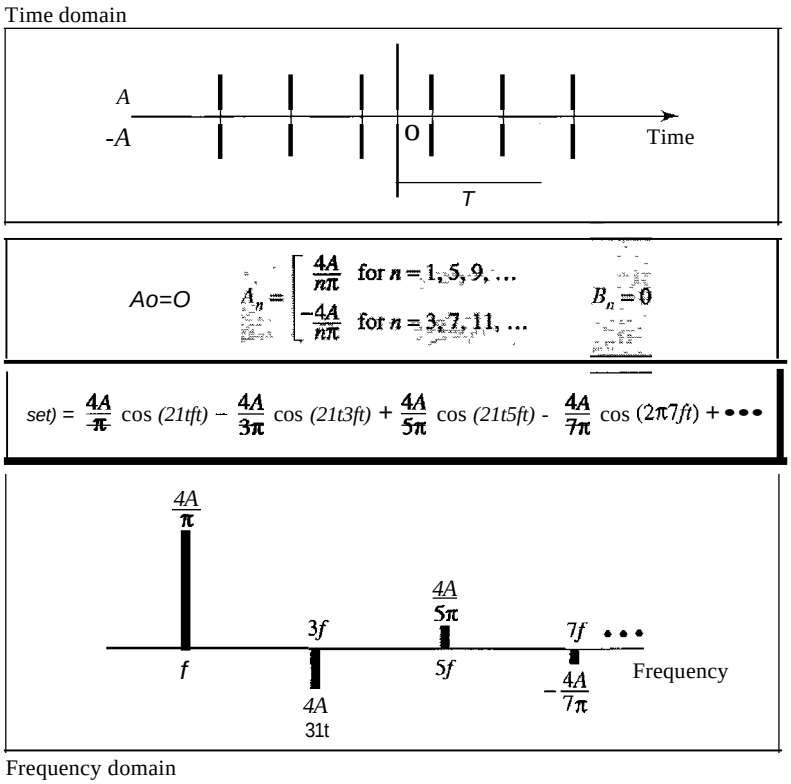


Fourier series

Time domain: periodic Frequency domain: discrete

only odd harmonics are present and the amplitudes are alternatively positive and negative. A very important point is that the amplitude of the harmonics approaches zero as we move toward infinity. Something which is not shown in the figure is the phase. However, we know that all components are cosine waves, which means that each has a phase of 90°.

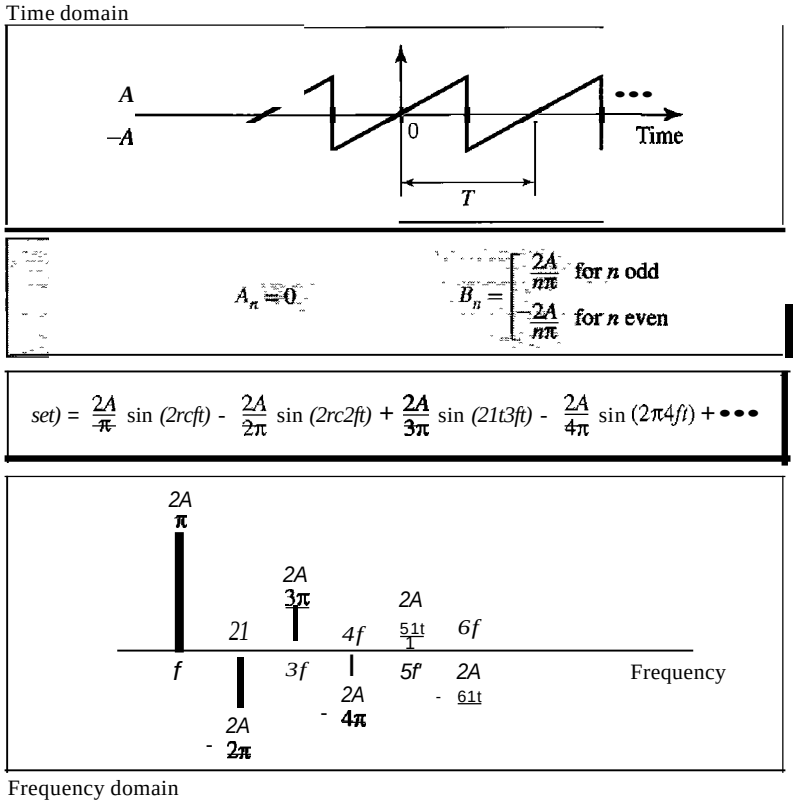
Figure C.6 *Finding the Fourier series ofa periodic square function*



Example C.4

Now let us show the components of a sawtooth signal as seen in Figure C.7. This time, we have only *En* components (sine waves). The frequency spectrum, however, is denser; we have all harmonics (*f, 2j, 3j, ...*). A point which is not clear from the diagram is the phase. All components are sine waves, which means each component has a phase of 0°.

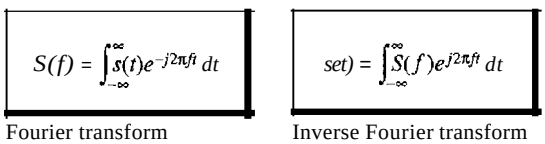
Figure C.7 *Finding the Fourier series for a sawtooth signal*



Fourier Transform

While the Fourier series gives the discrete frequency domain of a periodic signal, the Fourier transform gives the continuous frequency domain of a nonperiodic signal. Figure C.S shows how we can create a continuous frequency domain from a nonperiodic time-domain function and vice versa.

Figure C.S *Fourier transform and inverse Fourier transform*

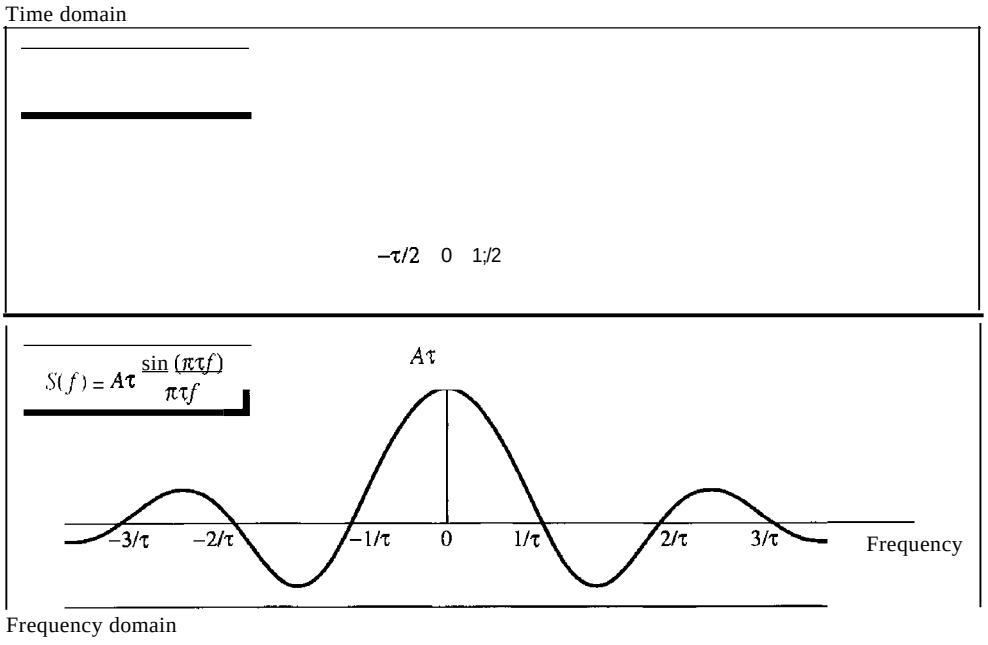


Fourier transform
Time domain: nonperiodic Frequency domain: continuous

Example C.S

Figure C.9 shows the time and frequency domains of one single square pulse. The time domain is between $-\tau/2$ and $\tau/2$; the frequency domain is a continuous function that stretches from negative infinity to positive infinity. Unlike the previous examples, the frequency domain is continuous; all frequencies are there, not just the integral ones.

Figure C.9 *Finding the Fourier transform of a square pulse*



Time-Limited and Band-Limited Signals

Two very interesting concepts related to the Fourier transform are the time-limited and band-limited signals. A time-limited signal is a signal for which the amplitude of *set)* is nonzero only during a period of time; the amplitude is zero everywhere else. A band-limited signal, on the other hand, is the signal for which the amplitude of *Sif)* is nonzero only for a range of frequencies; the amplitude is zero everywhere else. A band-limited signal plays a very important role in the sampling theorem and Nyquist frequency because the corresponding time domain can be represented as a series of samples.

Time-limited signal: $s(t) = 0$ for $|t| \leq T$
Band-limited signal: $Sif) = 0$ for $If| \leq B$

C.3 EXPONENT AND LOGARITHM

In solving networking problems, we often need to know how to handle exponential and logarithmic functions. This section briefly reviews these two concepts.

Exponential Function

The exponential function with **base** *a* is defined as

$$y = a^x$$

If x is an integer (integral value), we can easily calculate the value of y by multiplying the value of a by itself x times.

Example C.6

Calculate the value of the following exponential functions.

a. $y = 3^2$

b. $Y = 5.2^6$

Solution

a. $y = 3 \times 3 = 9$

b. $y = 5.2 \times 5.2 \times 5.2 \times 5.2 \times 5.2 \times 5.2 = 19{,}770.609664$

If x is not integer, we need to use a calculator.

Example C.7

Calculate the value of the following exponential functions.

a. $y = 3^{2.2}$

b. $y = 5.2^{6.3}$

Solution

a. $y = 11.212$ (approximately)

b. $y = 32{,}424.60$ (approximately)

Natural Base

One very common base used in science and mathematics is the **natural base** *e,* which has the value 2.71828183.... Most calculators show this function as e^x, which can be calculated easily by entering only the value of the exponent.

Example C.S

Calculate the value of the following exponential functions.

a. $y = e^4$

b. $y = e^{6.3}$

Solution

a. $y = 54.56$ (approximately)

b. $y = 544.57$ (approximately)

Properties of the Exponential Function

Exponential functions have several properties; some are useful to us in this text:

$$\text{First:} \quad y = a^0 = 1$$
$$\text{Second:} \quad y = a^1 = a$$
$$\text{Third:} \quad y = a^{-x} = \frac{1}{a^x}$$

Example C.9

The third property is useful to us because we can calculate the value of an exponential function with a negative value. We first calculate the positive value and we then invert the result.

a. $y = e^{-4}$

b. $y = e^{-6.3}$

Solution

a. $y = 1/54.56 = 0.0183$

b. $y = 1/544.57 = 0.00183$

Logarithmic Function

A logarithmic function is the inverse of an exponential function, as shown below. Just as in the exponential function, *a* is called the base of the logarithmic function:

$$y = a^x \quad \Longleftrightarrow \quad x = \log_a y$$

In other words, if *x* is given, we can calculate *y* by using the exponential function; if *y* is given, we can calculate *x* by using the logarithmic function.

Exponential and logarithmic functions are the inverse of each other.

Example C.10

Calculate the value of the following logarithmic functions.

a. $x = \log_3 9$

b. $x = \log_2 16$

Solution

We have not yet shown how to calculate the log function in different bases, but we can solve this problem intuitively.

a. Because $3^2 = 9$, we can say that $\log_3 9 = 2$, using the fact that the two functions are the inverse of each other.

b. Because $2^4 = 16$, we can say that $\log_2 16 = 4$ by using the previous fact.

Two Common Bases

The two common bases for logarithmic functions, those that can be handled by a calculator, are base *e* and base 10. The logarithm in base *e* is normally shown as ln (natural logarithm); the logarithm in base 10 is normally shown as log (omitting the base).

Example C11

Calculate the value of the following logarithmic functions.

a. $x = \log 233$

b. $x = \ln 45$

Solution

For these two bases we can use a calculator.

a. $x = \log 233 = 2.367$

b. $x = \ln 45 = 3.81$

Base Transformation

We often need to find the value of a logarithmic function in a base other than *e* or 10. If the available calculator cannot give the result in our desired base, we can use a very fundamental property of the logarithm, base transformation, as shown:

$$\log_a y = \frac{\log_b y}{\log_b a}$$

Note that the right-hand side is two log functions with base *b,* which is different from the base *a* at the left-hand side. This means that we can choose a base that is available in our calculator (base *b)* and find the log of a base that is not available (base *a).*

Example C12

Calculate the value of the following logarithmic functions.

a. $x = \log_3 810$

b. $x = \log_5 600$

Solution

These two bases, 3 and 5, are not available on a calculator, but we can use base 10 which is available.

a. $x = \log_3 810 = \frac{\log 810}{\log 3} = \frac{2.908}{0.477} = 6.095$

b. $x = \log_5 600 = \frac{\log 600}{\log 5} = \frac{2.778}{0.699} = 3.975$

Properties of Logarithmic Functions

Like an exponential function, a logarithmic function has some properties that are useful in: simplifying the calculation of a log function.

First:	$\log_a 1 = 0$	Fourth:	$\log_a (x \times y) = \log_a x + \log_a y$
Second:	$\log_a a = 1$	Fifth:	$\log_a \frac{x}{y} = \log_a x - \log_a y$
Third:	$\log_a \frac{1}{x} = -\log_a x$	Sixth:	$\log_a x^y = y \times \log_a x$

Example C13

Calculate the value of the following logarithmic functions.

a. $x = \log_3 1$
b. $x = \log_3 3$
c. $x = \log_{10} (1/10)$
d. $\log_a (x \times y)$ if we know that $\log_a x = 2$ and $\log_a y = 3$
e. $\log_z (1024)$ without using a calculator

Solution

We use the property of log functions to solve the problems.

a. $x = \log_3 1 = 0$
b. $x = \log_3 3 = 1$
c. $x = \log_{10} (1110) = \log_{10} 10^{-1} = -\log_{10} 10 = -1$
d. $\log_a (x \times y) = \log_a x + \log_a Y = 2 + 3 = 5$
e. $\log_z (1024) = \log_z (2^{10}) = 10 \log_z 2 = 10 \times 1 = 10$

APPENDIXD

8B/6TCode

This appendix is a tabulation of 8B/6T code pairs. The 8-bit data are shown in hexadecimal format. The 6T code is shown as + (positive signal), - (negative signal), and 0 (lack of signal) notation.

Table D.l *8B/6T code*

Data	*Code*	*Data*	*Code*	*Data*	*Code*	*Data*	*Code*
00	-+00-+	20	-++-00	40	-00+0+	60	0++0-0
01	0-+-+0	21	+00+--	41	0-00++	61	+0+-00
02	0-+0-+	22	-+0-++	42	0-0+0+	62	+0+0-0
03	0-++0-	23	+-0-++	43	0-0++0	63	+0+00-
04	-+0+0-	24	+-0+00	44	-00++0	64	0++00-
05	+0--+0	25	-+0+00	45	00-0++	65	++0-00
06	+0-0-+	26	+00-00	46	00-+0+	66	++00-0
07	+0-+0-	27	-+++--	47	00-++0	67	++000-
08	-+00+-	28	0++-0-	48	00+000	68	0++-+-
09	0-++-0	29	+0+0--	49	++-000	69	+0++--
OA	0-+0+-	2A	+0+-0-	4A	+-+000	6A	+0+-+-
OB	0-+-0+	2B	+0+--0	4B	-++000	6B	+0+--+
OC	-+0-0+	2C	0++--0	4C	0+-000	6C	0++--+
OD	+0-+-0	2D	++00--	4D	+0-000	6D	++0+--
OE	+0-0+-	2E	++0-0-	4E	0-+000	6E	++0-+-
OF	+0--0+	2F	++0--0	4F	-0+000	6F	++0--+
10	0--+0+	30	+-00-+	50	+--+0+	70	000++-
11	-0-0++	31	0+--+0	51	-+-0++	71	000+-+
12	-0-+0+	32	0+-0-+	52	-+-+0+	72	000-++
13	-0-++0	33	0+-+0-	53	-+-++0	73	000+00

Table D.I *8B/6T code (continued)*

Data	*Code*	*Data*	*Code*	*Data*	*Code*	*Data*	*Code*
14	0−−++0	34	+−0+0−	54	+--++0	74	000+0-
15	--00++	35	-0+-+0	55	−−+0++	75	000+-0
16	--0+0+	36	-0+0-+	56	--++0+	76	000-0+
17	--0++0	37	-0++0-	57	--+++0	77	000-+0
18	-+0-+0	38	+-00+-	58	--0+++	78	+++--0
19	+-0-+0	39	0+−+−0	59	-0-+++	79	+++-0-
lA	-++-+0	3A	0+-0+-	5A	0--+++	7A	+++0- −
IB	+00-+0	3B	0+--0+	5B	0--0++	7B	0++0- -
lC	+00+-0	3C	+−0−0+	5C	+--0++	7C	-00-++
ID	-+++-0	30	-0++-0	50	-000++	7D	-00+00
IE	+−0+−0	3E	-0+0+-	5E	0+++--	7E	+- --++
IF	-+0+-0	3F	-0+-0+	5F	0++-00	7F	+- -+00
80	-00+-+	AO	-++0-0	CO	-+0+-+	EO	-++0-+
81	0-0-++	Al	+-+-00	Cl	0-+-++	El	+-++0
82	0-0+-+	A2	+-+0-0	C2	0-++-+	E2	+-+0-+
83	0−0++−	A3	+−+00−	C3	0-+++-	E3	+-++0-
84	-00++-	A4	-++00-	C4	−+0++−	E4	-+++0-
85	00--++	A5	++--00	C5	+0--++	E5	++--+0
86	00-+-+	A6	++-0-0	C6	+0-+-+	E6	++-0-+
87	00−++−	A7	++−00−	C7	+0-++-	E7	++−+0−
88	-000+0	A8	-++-+-	C8	-+00+0	E8	-++0+-
89	0-0+00	A9	+-++--	C9	0-++00	E9	+-++-0
8A	0-00+0	AA	+−+−+−	CA	0-+0+0	EA	+−+0+−
8B	0-000+	AB	+-+--+	CB	0-+00+	EB	+-+-0+
8C	-0000+	AC	-++--+	CC	-+000+	EC	-++-0+
80	00-+00	AD	++-+--	CD	+0-+00	ED	++-+-0
8E	00-0+0	AE	++--+-	CE	+0-0+0	EE	++-0+-
8F	00-00+	AF	++---+	CF	+0-00+	EF	++--0+
90	+--+-+	BO	+000-0	DO	+-0+-+	FO	+000-+
91	-+--++	Bl	0+0-00	01	0+--++	Fl	0+0-+0
92	−+−+−+	B2	0+00-0	02	0+-+-+	F2	0+00-+
93	-+-++-	B3	0+000-	D3	0+-++-	F3	0+0+0-
94	+--++-	B4	+0000-	D4	+-0++-	F4	+00+0-

Table D.I *8B/6Tcode (continued)*

Data	*Code*	*Data*	*Code*	*Data*	*Code*	*Data*	*Code*
95	--+-++	B5	00+-00	05	-0+-++	F5	00+-+0
96	--++-+	B6	00+0-0	06	-0++-+	F6	00+0-+
97	--+++-	B7	00+00-	07	-0+++-	F7	00++0-
98	+--0+0	B8	+00-+-	08	+-00+0	F8	+000+-
99	-+-+00	B9	0+0+--	09	0+-+00	F9	0+0+-0
9A	-+-0+0	BA	0+0-+-	OA	0+-0+0	FA	0+00+-
9B	-+-00+	BB	0+0--+	OB	0+-00+	FB	0+0-0+
9C	+--00+	BC	+00--+	OC	+-000+	FC	+00-0+
90	--++00	BO	00++--	OO	-0++00	FD	00++-0
9E	--+0+0	BE	00+-+-	OE	-0+0+0	FE	00+0+-
9F	--+00+	BF	00+--+	OF	-0+00+	FF	00+-0+

APPENDIXE

Telephone History

In Chapter 9, we discussed telephone networks. In this appendix, we briefly review the history of telephone networks. The history in the United States can be divided into three eras: prior to 1984, between 1984 and 1996, and after 1996.

Before 1984

Before 1984, almost all local and long-distance services were provided by the AT&T Bell System. In 1970, the U.S. government, believing that the Bell System was monopolizing the telephone service industry, sued the company. The verdict was in favor of the government and resulted in a document called the Modified Final Judgment (MFJ). Beginning on January 1, 1984, AT&T was broken into AT&T Long Lines, 23 Bell Operating Companies (BOCs), and others. The 23 BOCs were grouped to make several Regional Bell Operating Companies (RBOSs). This landmark event, the AT&T divestiture of 1984, was beneficial to customers of telephone services. Telephone rates were lowered.

Between 1984 and 1996

The divestiture divided the country into more than 200 LATAs; some companies were allowed to provide services inside a LATA (LECs), and others were allowed to provide services between LATAs (IXCs). Competition, particularly between long-distance carriers, increased as new companies were formed. However, no LEC could provide long-distance services, and no IXCs could provide local services.

After 1996

Another major change in telecommunications occurred in 1996. The Telecommunications Act of 1996 combined the different services provided by different companies under the umbrella of telecommunication services; this included local services, long-distance voice and data services, video services, and so on. In addition, the act allowed any company to provide any of these services at the local and long-distance levels. In other words, a common carrier company provides services both inside the LATA and between the LATAs. However, to prevent the recabling of residents, the carriers that were given intra-LATA services (ILECs) continued to provide the main services; the new competitors (CLECs) provided other services.

APPENDIXF

Contact Addresses

The following is a list of contact addresses for various organizations mentioned in the text.

- **ATMForum**
  Presidio of San Francisco
  P.O. Box 29920 (mail)
  572B Ruger Street (surface)
  San Francisco, CA 94129-0920
  Telephone: 415 561-6275
  E-mail: info@atmforum.com
  www.atmforum.com
- **Federal Communications Commission** (FCC)
  445 12th Street S.W.
  Washington, DC 20554
  Telephone: 1-888-225-5322
  E-mail: fccinfo@fcc.gov
  www.fcc.gov
- **Institute of Electrical and Electronics Engineers (IEEE)**
  Operations Center
  445 Hoes Lane
  Piscataway, NJ 08854-1331
  Telephone: 732981-0060
  www.ieee.org

- International Organization for Standardization (ISO)
  1, rue de Varembe
  Caisse Posta1e 56
  CH-1211 Geneve 20
  Switzerland
  Telephone: 41 227490111
  E-mail: central@iso.ch
  www.iso.org
- International Telecommunication Union (ITU)
  Place des Nations
  CH-1211 Geneva 20
  Switzerland
  Telephone: 41 22 730 5852
  E-mail: tsbmail@itu.int
  www.itu.intlhome
- Internet Architecture Board (lAB)
  E-mail: lAB @isi.edu
  www.iab.org
- Internet Corporation for Assigned Names and Numbers (ICANN)
  4676 Admiralty Way, Suite 330
  Marina del Rey, CA 90292-6601
  Telephone: 310 823-9358
  E-mail: icann@icann.org
  www.icann.org
- Internet Engineering Steering Group (IESG)
  E-mail: iesg@ietf.org
  www.ietf.org/iesg.html
- Internet Engineering Task Force (IETF)
  E-mail: ietf-infor@ietf.org
  www.ietf.org
- Internet Research Task Force (IRTF)
  E-mail: irtf-chair@ietf.org
  www.irtf.org
- Internet Society (ISOC)
  1775 Weihle Avenue, Suite 102
  Reston, VA 20190-5108
  Telephone: 703.326-9880
  E-mail: info@isoc.org
  www.isoc.org

APPENDIXG

RFCs

In Table G.1, we list alphabetically by protocol the RFCs that are directly related to the material in this text. For more information go to the following site: http://www.rfc-editor.org.

Table G.t *RFCs for each protocol*

Protocol	*RFC*
ARPandRARP	826,903,925,1027,1293,1329,1433,1868,1931,2390
BGP	1092,1105,1163,1265,1266,1267,1364,1392,1403,1565, 1654,1655,1665,1771,1772,1745,1774,2283
BOOTP and DHCP	951, 1048, 1084, 1395, 1497, 1531, 1532, 1533, 1534, 1541, 1542,2131,2132
CIDR	1322,1478,1479,1517,1817
DHCP	See BOOTP and DHCP
DNS	799,811,819,830,881,882,883,897,920,921,1034,1035, 1386,1480,1535,1536,1537,1591,1637,1664,1706,1712, 1713,1982,2065,2137,2317,2535,2671
FTP	114,133,141,163,171,172,238,242,250,256,264,269,281, 291,354,385,412,414,418,430,438,448,463,468,478,486, 505,506,542,553,624,630,640,691,765,913,959,1635, 1785,2228,2577
HTML	1866
HTTP	2068,2109
ICMP	777,792,1016,1018,1256,1788,2521
IGMP	966,988,1054,1112,1301,1458,1469,1768,2236,2357,2365, 2502,2588
IMAP	See SMTP, MIME, POP, IMAP
IP	760,781,791,815, 1025, 1063, 1071, 1141, 1190, 1191, 1624,2113

Table G.! *RFCs for each protocol (continued)*

Protocol	*RFC*
IPv6	1365,1550,1678, 1680,1682,1683,1686,1688,1726,1752, 1826,1883,1884,1886,1887,1955,2080,2373,2452,2463, 2465,2466,2472,2492,2545,2590
MIB	See SNMP, MIB, SMI
MIME	See SMTP, MIME, POP, IMAP
Multicast Routing	1584,1585,2117,2362
NAT	1361,2663,2694
OSPF	1131,1245,1246,1247,1370,1583, 1584, 1585, 1586, 1587, 2178,2328,2329,2370
POP	See SMTP, MIME, POP, IMAP
RARP	See ARP and RARP
RIP	1131,1245,1246,1247,1370,1583, 1584, 1585, 1586, 1587, 1722,1723,2082,2453
SCTP	2960,3257,3284,3285,3286,3309,3436,3554,3708,3758
SMI	See SNMP, MIB, SMI
SMTP, MIME, POP, IMAP	196,221,224,278,524,539,753,772,780,806,821,934,974, 1047, 1081, 1082, 1225, 1460, 1496, 1426, 1427, 1652, 1653, 1711,1725,1734,1740,1741,1767,1869,1870,2045,2046, 2047,2048,2177,2180,2192,2193,2221,2342,2359,2449, 2683,2503
SNMP, MIB, SMI	1065, 1067, 1098, 1155, 1157, 1212, 1213, 1229, 1231,1243, 1284, 1351, 1352, 1354, 1389, 1398, 1414, 1441,1442,1443, 1444, 1445, 1446, 1447, 1448, 1449, 1450, 1451, 1452, 1461, 1472, 1474, 1537, 1623, 1643, 1650, 1657, 1665, 1666, 1696, 1697, 1724, 1742, 1743, 1748, 1749
TCP	675,700,721,761,793,879,896,1078,1106,1110,1144,1145, 1146,1263,1323,1337,1379,1644,1693,1901,1905,2001, 2018,2488,2580
TELNET	137,340,393,426,435,452,466,495,513,529,562,595,596, 599,669,679,701,702,703,728,764,782,818,854,855,1184, 1205,2355
TFfP	1350, 1782, 1783, 1784
UDP	768
VPN	2547,2637,2685
WWW	1614, 1630, 1737, 1738

APPENDIXH

UDP and TCP Ports

Table H.I lists the common well-known ports ordered by port number.

Table H.I *Ports by port number*

Port Number	*UDP/TCP*	*Protocol*
7	TCP	ECHO
13	UDPrrCp	DAYTIME
19	UDPITCP	CHARACTER GENERATOR
20	TCP	FTP-DATA
21	TCP	FTP-CONTROL
23	TCP	TELNET
25	TCP	SMTP
37	UDPrrCp	TIME
67	UDP	BOOTP-SERVER
68	UDP	BOOTP-CLIENT
69	UDP	TFTP
70	TCP	GOPHER
79	TCP	FINGER
80	TCP	HTTP
109	TCP	POP-2
110	TCP	POP-3
111	UOPITCP	RPC
161	UOP	SNMP
162	UOP	SNMP-TRAP
179	TCP	BGP
520	UOP	RIP

Table H.2 lists the ports, ordered alphabetically by protocol.

Table H.2 *Port numbers by protocol*

Protocol	*UDPITCP*	*Port Number*
BGP	TCP	179
BOOTP-SERVER	UDP	67
BOOTP-CLIENT	UDP	68
CHARACTER GENERATOR	UDP/TCP	19
DAYTIME	UDP/TCP	13
ECHO	TCP	7
FINGER	TCP	79
FTP-CONTROL	TCP	21
FTP-DATA	TCP	20
GOPHER	TCP	70
HTTP	TCP	80
POP-2	TCP	109
POP-3	TCP	110
RIP	UDP	520
RPC	UDP/TCP	111
SMTP	TCP	25
SNMP	UDP	161
SNMP-TRAP	UDP	162
TELNET	TCP	23
TFTP	UDP	69
TIME	*UDP/TCP*	37