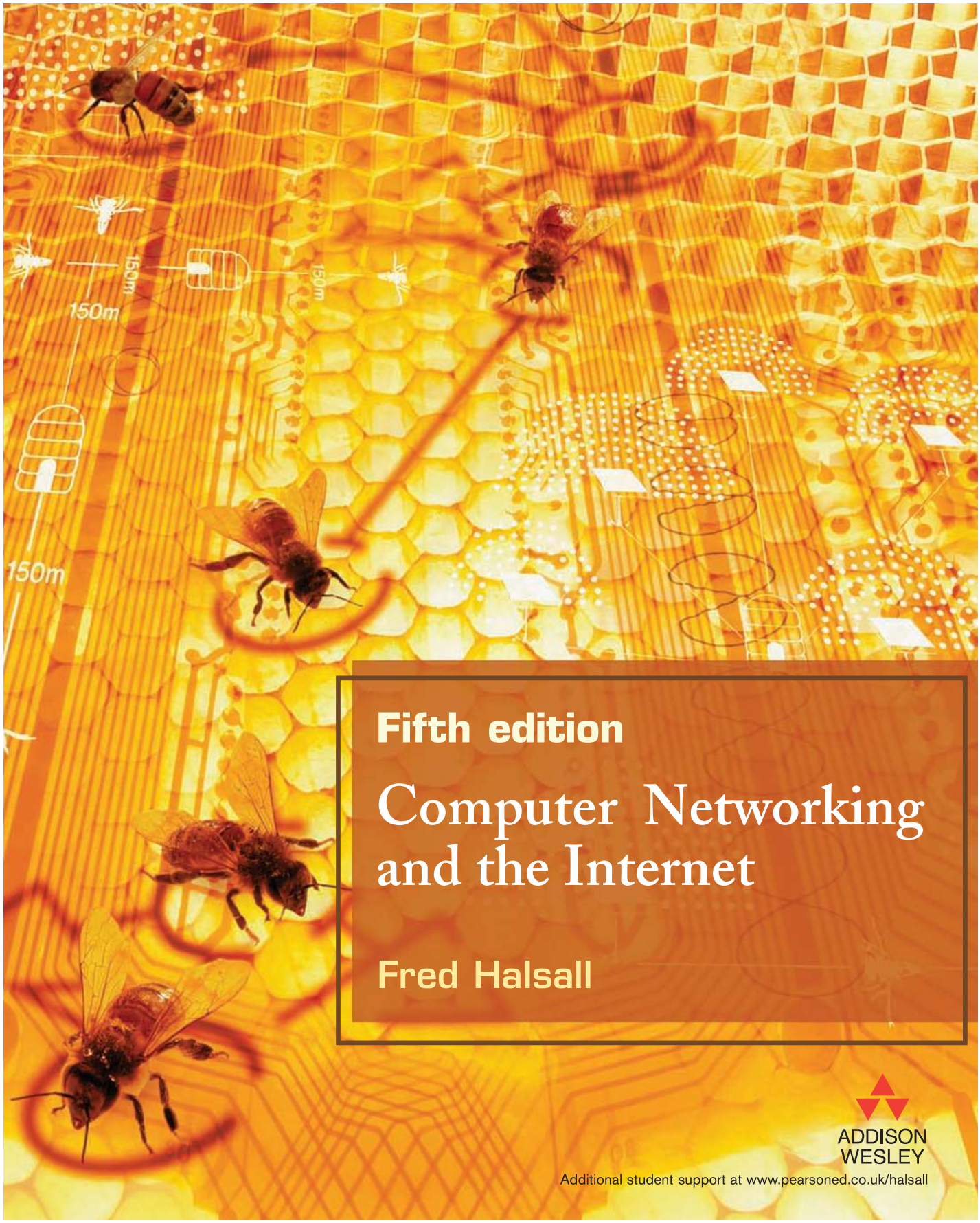


Fifth edition

Computer Networking and the Internet

Fred Halsall

ADDISON WESLEY

Additional student support at www.pearsoned.co.uk/halsall

Computer Networking and the Internet

Visit the *Computer Networking and the Internet, fifth edition* Companion Website at **www.pearsoned.co.uk/halsall** to find valuable **student** learning material including:

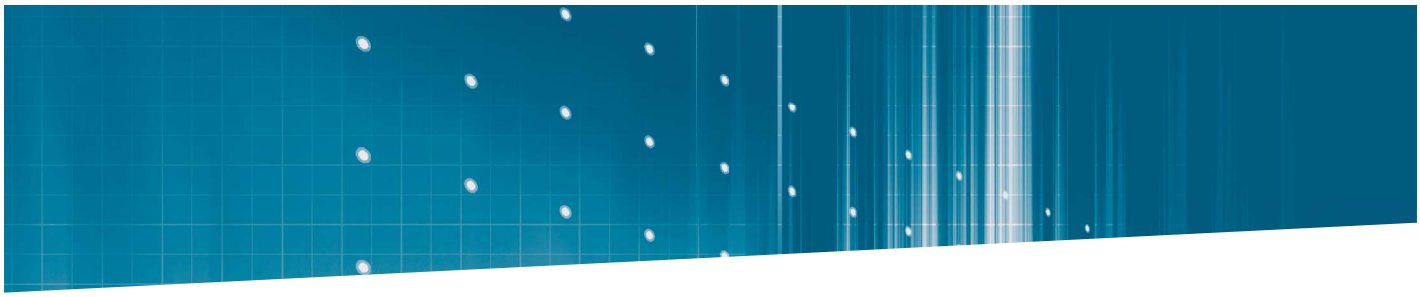
- Multiple choice questions to help test your learning
- Annotated links to relevant sites on the web

Web material prepared by Richard Read of Middlesex University

We work with leading authors to develop the strongest educational materials in computing, bringing cutting-edge thinking and best learning practice to a global market.

Under a range of well-known imprints, including Addison Wesley, we craft high quality print and electronic publications which help readers to understand and apply their content, whether studying or at work.

To find out more about the complete range of our publishing, please visit us on the World Wide Web at: www.pearsoned.co.uk



Computer Networking and the Internet

Fifth edition

Fred Halsall

ADDISON-WESLEY

An imprint of Pearson Education

Harlow, England · London · New York · Reading, Massachusetts · San Francisco · Toronto · Don Mills, Ontario · Sydney
Tokyo · Singapore · Hong Kong · Seoul · Taipei · Cape Town · Madrid · Mexico City · Amsterdam · Munich · Paris · Milan

Pearson Education Limited
Edinburgh Gate
Harlow
Essex CM20 2JE
England

and Associated Companies throughout the world.

Visit us on the World Wide Web at:
www.pearsoned.co.uk

First published 1985
Fifth edition published 2005

ISBN 0-321-26358-8

British Library Cataloguing-in-Publication Data
A catalogue record for this book can be obtained from the British Library.

10 9 8 7 6 5 4 3 2 1
09 08 07 06 05

Typeset by 30 in New Baskerville 10/12pt
Printed and bound in the United States of America

The publisher's policy is to use paper manufactured from sustainable forests.

Contents

Supporting resources

Visit **www.pearsoned.co.uk/halsall** to find valuable online resources

Companion Website for students

- Multiple choice questions to help test your learning
- Annotated links to relevant sites on the web

For instructors

- PowerPoint slides containing all the diagrams from the text and the worked examples, that can be downloaded and used as OHTs

Also: The regularly maintained Companion Website provides the following features:

- Search tool to help locate specific items of content
- E-mail results and profile tools to send results of quizzes to instructors
- Online help and support to assist with website usage and troubleshooting

Web material prepared by Richard Read of Middlesex University

For more information please contact your local Pearson Education sales representative or visit **www.pearsoned.co.uk/halsall**

OneKey: All you and your students need to succeed

OneKey is an exclusive new resource for instructors and students, giving you access to the best online teaching and learning tools 24 hours a day, 7 days a week.

OneKey
OneKey is all you need
Convenience. Simplicity. Success

OneKey means all your resources are in one place for maximum convenience, simplicity and success.

A OneKey product is available for *Computer Networking and the Internet, fifth edition* for use with Blackboard™, WebCT and CourseCompass. It contains:

For students

- Learning objectives for each chapter
- Multiple choice questions to help test your learning
- Annotated links to relevant sites on the web
- Interactive animations of key figures

For instructors

- PowerPoint slides containing all the diagrams from the text and the worked examples, that can be downloaded and used as OHTs

For more information about the OneKey product please contact your local Pearson Education sales representative or visit **www.pearsoned.co.uk/onekey**

Preface

Objectives

Prior to the introduction of the World Wide Web, there were many different types of computer networks used to interconnect geographically distributed sets of computers. Many large corporations and businesses, for example, often used proprietary networks, each with its own protocols and networking infrastructure, while, at the same time, the Internet was used primarily to interconnect distributed sets of computers that were located at academic and research institutions around the world. Typical applications were electronic mail and more general file transfers between computers.

With the advent of the Web, however, since the Internet was the networking infrastructure used for the Web, the Internet has rapidly become the dominant computer network as people at home and at work started to use the Web. Typical applications are interactive entertainment, electronic commerce, and so on, which, of course, are in addition to the standard applications already supported by the Internet. As we can deduce from this, therefore, the subject of computer networking is now synonymous with the study of the Internet and its applications.

The combined effect of these developments means that the number of users of the Internet has expanded rapidly. To support this expansion, instead of most users accessing the Internet through an academic network, a number of different types of access network are now used. For example, most users at home and in small businesses gain access through their local switched telephone network using either a low bit rate modem or, more usually, a broadband modem. Alternatively, for cable television subscribers, access is often through a high bit rate cable modem. In practice, however, both access methods provide only a physical connection to a second network called an Internet service provider (ISP) network. This, as its name implies, provides the access point to the Internet for a set of users that fall within the field of coverage of a particular ISP network. Clearly, therefore, there are many ISPs each of which is a private, commercial company and hence access to the Internet through an ISP must be paid for.

In addition to telephone and cable networks, with the introduction of Internet-enabled mobile phones and laptops with radio interfaces, many users on the move now gain access to the Internet using a regional/national/international cellular phone network. All of these different types of access network

are in addition, of course, to the conventional site networks used by large corporations and businesses. Because of the importance of the Web to their businesses, most of these site networks now use the same protocols as the Internet to facilitate interworking with it.

In many instances, the expanding range of applications supported by the Internet has come about through the technological advances in the way the user data associated with these applications is represented. For example, until relatively recently, a number of the access networks that are now used, in addition to providing their basic service such as telephony, only supported applications in which the application data was composed of text comprising strings of alphanumeric characters entered at a keyboard. As a result of the technological advances in the area of compression, however, the same access networks can now support a much richer set of applications involving multiple data types. These include, in addition to text, digitized images, photos/pictures, speech, audio and video.

As we can deduce from this brief overview, to study the technological issues relating to computer networking and the Internet requires an in-depth understanding not only of the operation of the Internet itself but also the operation of the different types of access network that are used and how they interface with the Internet. In addition, because many of the applications involve the transfer of sensitive information, the topic of security is now essential when describing the operation of the Internet. The aim of this book is to provide this body of knowledge. To do this, the book is divided into two logical parts. The first – Chapters 1 through 5 – is concerned with the fundamentals of digital transmission and communication protocols together with descriptions of the mode of operation of the different types of access network that are now used and how they interface with the global Internet. The second – Chapters 6 through 10 – describes the architecture and communication protocols used by the Internet and the applications that it supports. In addition, the second part describes the techniques that are used to ensure that all the data relating to these applications are transferred in a secure way.

The book has five appendices. In Appendix A we present descriptions of how the different types of data used in the various Internet applications are represented together with an overview of the operation of the compression algorithms that are employed with text, digitized images, photos/pictures, speech, audio and video.

As we shall explain, when transmitting digital data over a network, bit corruptions/errors are often introduced. Hence in Appendix B we describe a number of the different methods that are used to detect the presence of transmission/bit errors in a received block of data.

Normally, when a bit error within a block of data is detected, another copy of the block is requested by the receiving device. In some instances, however – for example when data is being transmitted over a radio/wireless link – the frequency of bit errors is such that the request message for a new copy of a corrupted block may also be corrupted and hence an alternative

approach must be used. This involves adding significantly more what are called *error control bits*. These are added in such a way that the receiver can use them to deduce what the original data block contained. This approach is called forward error control and an introduction to this topic is given in Appendix C.

Wireless networks – that is, networks that use radio as the transmission medium – are now widely used in a number of access networks. In Appendix D, therefore, we give a short introduction to the subject of radio propagation and transmission so that the standards relating to this type of network can be understood.

As we shall see, the global Internet is composed of many thousands of networks that are organized into hierarchical layers. At the higher layers, the networks must route data through them at very high rates and are called backbone networks. In Appendix E we describe the technology that is used to achieve these very high switching rates.

Intended readership

The book has been written primarily as a course textbook for both university and college students studying courses relating to the technical issues associated with computer networking based on the Internet, its protocols and applications. Typically, the students will be studying in a computer science, computer systems, computer engineering or electronic engineering department/school. In addition, the book is suitable for computer professionals and engineers who wish to build up a working knowledge of this rapidly evolving subject. At one extreme this requires the reader to understand the techniques that are used to transmit a digital bitstream over the different types of transmission medium, such as copper wire, coaxial cable, radio and optical fiber. At the other extreme it requires an understanding of the software that is used in the different types of equipment – personal computers, workstations, laptops, mobile phones, set-top boxes, etc. – that are used to support Internet applications. The first is the domain of the electronics engineer and the second of the computer scientist. Care has been taken, however, to ensure that the book is suitable for use with courses for both types of student by ensuring that the level of detail required in each subject area is understandable by both categories of reader.

In order to achieve this goal, an introductory chapter has been included that describes the basic hardware and software techniques that are used to achieve the reliable transfer of a block/stream of digital data over a transmission channel. These include the different methods that are used to detect the presence of transmission errors – bit corruptions – in a received block/stream of data and the procedures that are followed to obtain another copy of the block/stream when this occurs. The latter form what is called a communications protocol. Hence this chapter also includes an introduction to the

subject of protocols to give the reader who has no previous knowledge of this subject the necessary foundation for the later chapters that describe the operation of the different types of access networks and the protocols and applications of the Internet.

Intended usage

To the instructor

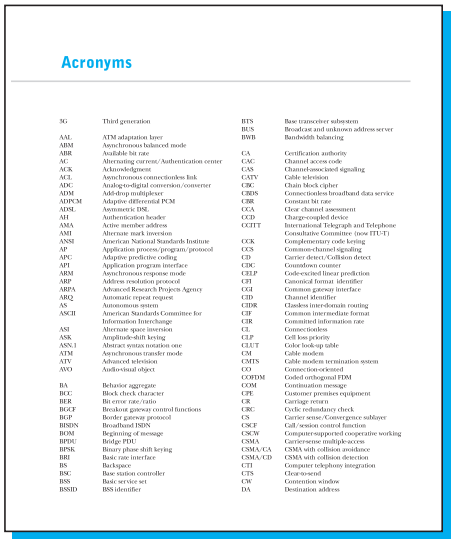
As we can see from the list of contents, the book covers a range of topics each of which is to a depth that makes it interesting and academically challenging. As a result, the book can be used for a number of different courses relating to computer networking and the Internet. Ideally, in order to obtain an in-depth technical understanding of the subject area, a set of courses should be used to collectively cover the total contents of the book from the principles of data communications through to details of the different types of access networks and the protocols and applications of the Internet. Alternatively, it can be used for one or two courses each of which covers a subset of this subject area. For example, one course may cover the basics of digital communications and an overview of the operation of the different types of access network that are used with the Internet. The second course can then cover the architecture and detailed operation of the Internet and its protocols together with its applications including the World Wide Web and the topic of security. The book is considered to be suitable for both undergraduate and taught masters courses.

As indicated earlier, all of the topics are covered to a depth that enables the reader to build up an in-depth technical understanding of the subject. Hence because of the technical nature of the subject, to help the reader to understand each topic within an area, either a worked example or a relatively detailed diagram is used to illustrate the concepts involved. This is considered to be one of the main advantages of the book over competing texts owing to the technical detail associated with many of the diagrams. Also, both the examples and diagrams are seen as being particularly useful for instructors as they can be used directly for lectures. To facilitate this, therefore, both the worked examples and all the diagrams are available to instructors in their electronic form so reducing considerably the time required to prepare a set of lectures for a course. These can be downloaded from *www.booksites.net/halsall*. In addition, each chapter has a comprehensive set of exercises that have been structured to help the student to revise the topics covered in that chapter in a systematic way.

To the student

The book has been structured to be used for self-study. Worked examples are included in most chapters and, to aid understanding of all the topics that are covered, associated with each topic is a relatively detailed diagram that illustrates the concepts involved. These you should find particularly useful since they facilitate understanding the technical details relating to the many topics covered. In addition, the comprehensive set of exercises at the end of each chapter have been structured to help you to test your knowledge and understanding of each of the topics covered in that chapter in a systematic way. In order to aid self-study, there is an exercise for each topic discussed within a section. Hence for each question within a section heading, you can relate back to the topic within that section of the book to find the answer.

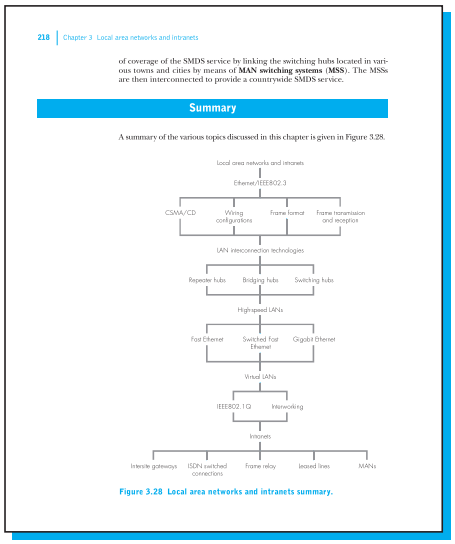
Guided Tour



Acronyms

3G	Third generation	BTS	Base transceiver subsystem
		BUS	Broadcast and unknown address server
AAL	ATM adaptation layer	BWB	Bandwidth balancing
ABM	Asynchronous balanced mode		
ABR	Available bit rate	CA	Certification authority
AC	Alternating current/Authentication center	CAC	Channel access code
ACK	Acknowledgment	CAS	Channel-associated signalling
ACL	Asynchronous connectionless link	CATV	Cable television
ADC	Analog-to-digital conversion/converter	CBC	Chain block cipher
ADM	Add-drop multiplexer	CBDS	Connectionless broadband data service
ADPCM	Adaptive differential PCM	CBR	Constant bit rate
ADSL	Asymmetric DSL	CCA	Clear channel assessment
AH	Authentication header	CCD	Charge-coupled device
AMA	Active member address	CCITT	International Telegraph and Telephone Consultative Committee (now ITU-T)
AMI	Alternate mark inversion		
ANSI	American National Standards Institute	CCK	Complementary code keying
AP	Application process/program/protocol	CCS	Common-channel signalling
APC	Adaptive predictive coding	CD	Carrier detect/Collision detect
API	Application program interface	CDC	Countdown counter
ARM	Asynchronous response mode	CELP	Code-excited linear prediction
ARP	Address resolution protocol	CFI	Canonical format identifier
ARPA	Advanced Research Projects Agency	CGI	Common gateway interface
ARQ	Automatic repeat request	CID	Channel identifier
AS	Autonomous system	CIDR	Classless inter-domain routing
ASCII	American Standards Committee for Information Interchange	CIF	Common intermediate format
		CIR	Committed information rate
ASI	Alternate space inversion	CL	Connectionless
ASK	Amplitude-shift keying	CLP	Cell loss priority
ASN.1	Abstract syntax notation one	CLUT	Color look-up table
ATM	Asynchronous transfer mode	CM	Cable modem
ATV	Advanced television	CMTS	Cable modem termination system
AVO	Audio-visual object	CO	Connection-oriented
		COFDM	Coded orthogonal FDM
BA	Behavior aggregate	COM	Continuation message
BCC	Block check character	CPE	Customer premises equipment
BER	Bit error rate/ratio	CR	Carriage return
BGCF	Breakout gateway control functions	CRC	Cyclic redundancy check
BGP	Border gateway protocol	CS	Carrier sense/Convergence sublayer
BISDN	Broadband ISDN	CSCF	Call/session control function
BOM	Beginning of message	CSCW	Computer-supported cooperative working
BPDU	Bridge PDU	CSMA	Carrier-sense multiple-access
BPSK	Binary phase-shift keying	CSMA/CA	CSMA with collision avoidance
BRI	Basic rate interface	CSMA/CD	CSMA with collision detection
BS	Backspace	CTI	Computer telephony integration
BSC	Base station controller	CTS	Clear-to-send
BSS	Basic service set	CW	Contention window
BSSID	BSS identifier	DA	Destination address

Acronyms: a full list is provided at the front of the book for easy reference.



of coverage of the SMDS service by linking the switching hubs located in various towns and cities by means of **MAN switching systems** (**MSS**). The MSSs are then interconnected to provide a countrywide SMDS service.

Summary

A summary of the various topics discussed in this chapter is given in Figure 3.28.

Figure 3.28 Local area networks and intranets summary.

Visual Summaries at the end of each chapter help you to learn and revise key concepts at a glance.

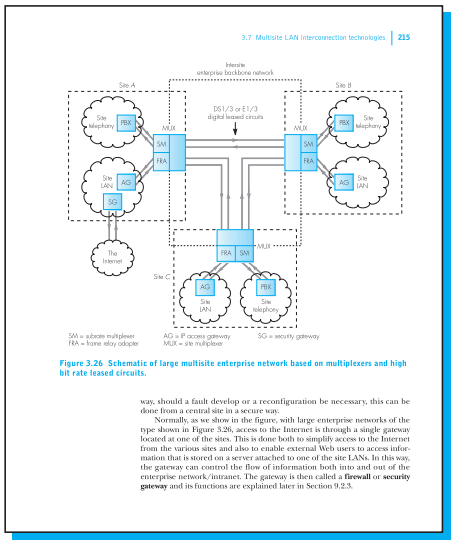
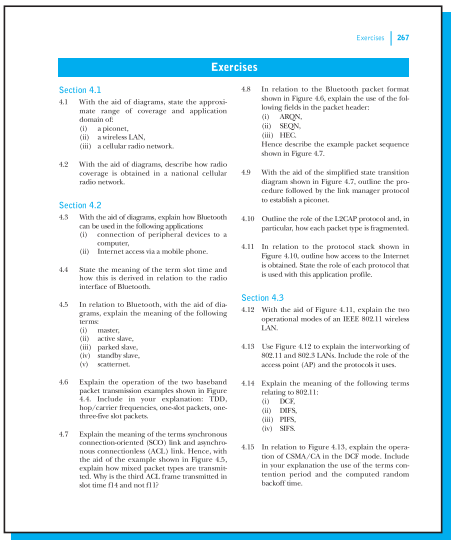


Figure 3.26 Schematic of large multisite enterprise network based on multiplexers and high bit rate leased circuits.

way, should a fault develop or a reconfiguration be necessary, this can be done from a central site in a secure way.

Normally, as we show in the figure, with large enterprise networks of the type shown in Figure 3.26, access to the Internet is through a single gateway located at one of the sites. This is done both to simplify access to the Internet from the various sites and also to enable external Web users to access information that is stored on a server attached to one of the site LANs. In this way, the gateway can control the flow of information both into and out of the enterprise network/intranet. The gateway is then called a **firewall** or **security gateway** and its functions are explained later in Section 9.2.3.

Diagrams provide a clear visual understanding of complex processes in Networking.



Exercises

Section 4.1

4.1 With the aid of diagrams, state the approximate range of coverage and application domain of:
(i) a piconet,
(ii) a wireless LAN,
(iii) a cellular radio network.

4.2 With the aid of diagrams, describe how radio coverage is obtained in a national cellular radio network.

Section 4.2

4.3 With the aid of diagrams, explain how Bluetooth can be used in the following applications:
(i) connection of peripheral devices to a computer,
(ii) Internet access via a mobile phone.

4.4 State the meaning of the term slot time and how this is derived in relation to the radio interface of Bluetooth.

4.5 In relation to Bluetooth, with the aid of diagrams, explain the meaning of the following terms:
(i) master,
(ii) active slave,
(iii) parked slave,
(iv) standby slave,
(v) scatternet.

4.6 Explain the operation of the two baseband packet transmission examples shown in Figure 4.4. Include in your explanation: TDD, hop/carrier frequencies, one-slot packets, one-three-five slot packets.

4.7 Explain the meaning of the terms synchronous connection-oriented (SCO) link and asynchronous connectionless (ACL) link. Hence, with the aid of the example shown in Figure 4.5, explain how mixed packet types are transmitted. Why is the third ACL frame transmitted in slot time f14 and not f13?

4.8 In relation to the Bluetooth packet format shown in Figure 4.6, explain the use of the following fields in the packet header:
(i) ARQN,
(ii) SEQN,
(iii) HEC.
Hence describe the example packet sequence shown in Figure 4.7.

4.9 With the aid of the simplified state transition diagram shown in Figure 4.7, outline the procedure followed by the link manager protocol to establish a piconet.

4.10 Outline the role of the L2CAP protocol and, in particular, how each packet type is fragmented.

4.11 In relation to the protocol stack shown in Figure 4.10, outline how access to the Internet is obtained. State the role of each protocol that is used with this application profile.

Section 4.3

4.12 With the aid of Figure 4.11, explain the two operational modes of an IEEE 802.11 wireless LAN.

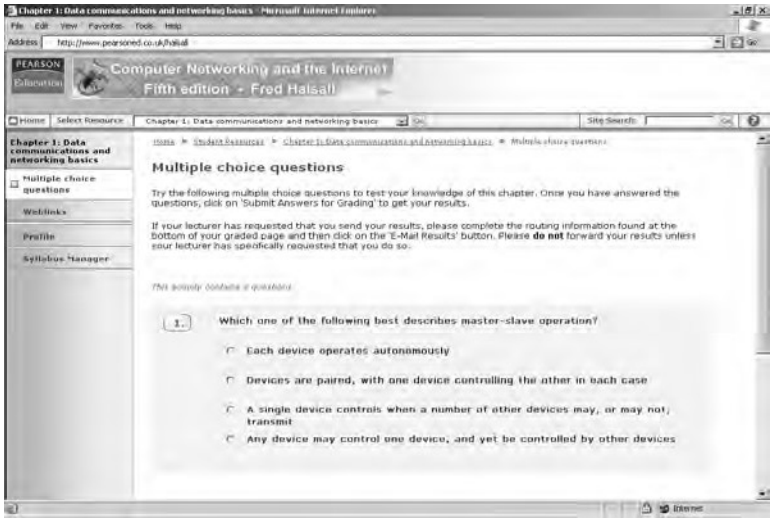
4.13 Use Figure 4.12 to explain the interworking of 802.11 and 802.3 LANs. Include the role of the access point (AP) and the protocols it uses.

4.14 Explain the meaning of the following terms relating to 802.11:
(i) DCF,
(ii) DIFS,
(iii) PIFS,
(iv) SIFS.

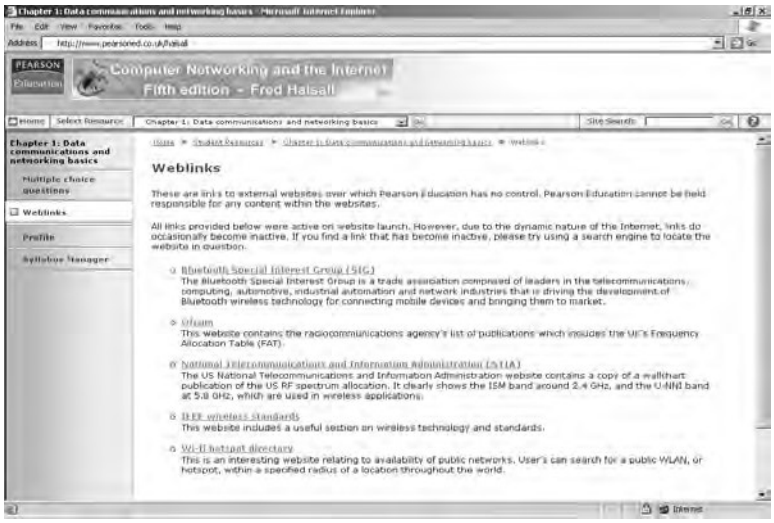
4.15 In relation to Figure 4.13, explain the operation of CSMA/CA in the DCF mode. Include in your explanation the use of the terms contention period and the computed random backoff time.

Exercises are provided for each section, to check your understanding of key topics.

Guided Tour of the Website



Multiple choice questions



Weblinks

Acknowledgments

I should like to take this opportunity to thank various people for their help during the period I was preparing the manuscript. Firstly to Simon Plumtree and Keith Mansfield at Pearson Education for their enthusiasm and motivation that encouraged me to write the book. Secondly to my wife Rhiannon for her unwavering support, patience, and understanding while I was writing the book. Finally, to our first grandchild Annelie Grace; I promise that now I have finished writing my book, I shall take you out much more frequently. It is to Annelie and her mother Lisa that I dedicate the book.

Fred Halsall
December 2004

Acronyms

3G	Third generation
AAL	ATM adaptation layer
ABM	Asynchronous balanced mode
ABR	Available bit rate
AC	Alternating current/Authentication center
ACK	Acknowledgment
ACL	Asynchronous connectionless link
ADC	Analog-to-digital conversion/converter
ADM	Add-drop multiplexer
ADPCM	Adaptive differential PCM
ADSL	Asymmetric DSL
AH	Authentication header
AMA	Active member address
AMI	Alternate mark inversion
ANSI	American National Standards Institute
AP	Application process/program/protocol
APC	Adaptive predictive coding
API	Application program interface
ARM	Asynchronous response mode
ARP	Address resolution protocol
ARPA	Advanced Research Projects Agency
ARQ	Automatic repeat request
AS	Autonomous system
ASCII	American Standards Committee for Information Interchange
ASI	Alternate space inversion
ASK	Amplitude-shift keying
ASN.1	Abstract syntax notation one
ATM	Asynchronous transfer mode
ATV	Advanced television
AVO	Audio-visual object
BA	Behavior aggregate
BCC	Block check character
BER	Bit error rate/ratio
BGCF	Breakout gateway control functions
BGP	Border gateway protocol
BISDN	Broadband ISDN
BOM	Beginning of message
BPDU	Bridge PDU
BPSK	Binary phase shift keying
BRI	Basic rate interface
BS	Backspace
BSC	Base station controller
BSS	Basic service set
BSSID	BSS identifier
BTS	Base transceiver subsystem
BUS	Broadcast and unknown address server
BWB	Bandwidth balancing
CA	Certification authority
CAC	Channel access code
CAS	Channel-associated signaling
CATV	Cable television
CBC	Chain block cipher
CBDS	Connectionless broadband data service
CBR	Constant bit rate
CCA	Clear channel assessment
CCD	Charge-coupled device
CCITT	International Telegraph and Telephone Consultative Committee (now ITU-T)
CCK	Complementary code keying
CCS	Common-channel signaling
CD	Carrier detect/Collision detect
CDC	Countdown counter
CELP	Code-excited linear prediction
CFI	Canonical format identifier
CGI	Common gateway interface
CID	Channel identifier
CIDR	Classless inter-domain routing
CIF	Common intermediate format
CIR	Committed information rate
CL	Connectionless
CLP	Cell loss priority
CLUT	Color look-up table
CM	Cable modem
CMTS	Cable modem termination system
CO	Connection-oriented
COFDM	Coded orthogonal FDM
COM	Continuation message
CPE	Customer premises equipment
CR	Carriage return
CRC	Cyclic redundancy check
CS	Carrier sense/Convergence sublayer
CSCF	Call/session control function
CSCW	Computer-supported cooperative working
CSMA	Carrier-sense multiple-access
CSMA/CA	CSMA with collision avoidance
CSMA/CD	CSMA with collision detection
CTI	Computer telephony integration
CTS	Clear-to-send
CW	Contention window
DA	Destination address

DAC	Digital-to-analog conversion/converter
DBS	Digital broadcast satellite
DC	Direct current
DCE	Data circuit terminating equipment
DCF	Distributed coordination function
DCT	Discrete cosine transform
DEL	Delete
DES	Data encryption standard
DFT	Discrete Fourier transform
DHCP	Dynamic host configuration protocol
DIFS	DCF inter-frame spacing
DLC	Data link control
DLE	Data link escape (character)
DMPDU	Derived MAC PDU
DMT	Discrete multitone
DN	Distinguished name
DNS	Domain name server
DOCSIS	Data-over-cable service interface specification
DPC	Designated port cost
DPCM	Differential PCM
DPLL	Digital phase-locked line
DQDB	Distributed queue dual bus
DS	Differentiated services/Downstream
DSL	Digital subscriber line
DSLAM	Digital subscriber line access multiplexer
DSSS	Direct sequence spread spectrum
DTE	Data terminal equipment
DTMF	Dual-tone multiple frequency
DVA	Distance vector algorithm
DVB	Digital video broadcast
DVB-S/T	DVB-satellite/terrestrial
DVD	Digital versatile disk
DVMRP	Distance vector MRP
EBCDIC	Extended binary coded decimal interchange code
ECB	Electronic code book/Event control block
ECN	Explicit congestion notification
ED	End delimiter
EF	Expedited forwarding
EGP	Exterior gateway protocol
EIA	Electrical Industries Association
EIR	Equipment identity register
EMS	Enhanced message service
EOM	End of message
EOS	End of stream
ES	End system
ESC	Escape
ESP	Encryption security payload
ETX	End of text
FCS	Frame check sequence
FDD	Frequency division duplex
FDDI	Fiber distributed data interface
FDM	Frequency-division multiplexing
FF	Form feed
FHSS	Frequency-hopping spread spectrum
FIFO	First-in, first-out
FM	Frequency modulation
FN	Fiber node
FQDN	Fully-qualified DN
FRA	Frame relay adapter
FS	File separator
FSK	Frequency-shift keying
FTP	File transfer protocol
FTTB	Fiber-to-the-building
FTTC/K	Fiber-to-the-curb/kerb
FTTcab	Fiber-to-the-cabinet
FTTH	Fiber-to-the-home
GA	Grand Alliance
GB	Guard-band
GEO	Geostationary/geosynchronous earth orbit
GGSN	Gateway GPRS support node
GIF	Graphics interchange format
GLP	Gateway location protocol
GOB	Group of blocks
GOP	Group of pictures
GPRS	General packet radio service
GSM	Global system for mobile communications
GSTN	General switched telephone network
GTP	GPRS tunneling protocol
GW	Gateway
HDB3	High density bipolar 3
HDLC	High-level data link contol
HDSL	High-speed DSL
HDTV	High-definition television
HE	Headend
HEC	Header error checksum
HFC	Hybrid fiber coax
HLR	Home location register
HMAC	Hash message authentication code
HS	Home subscriber server
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPD	HTTP daemon
IA5	International alphabet number five
IAC	Inquiry access code
ICANN	Internet Corporation for Assigned Names and Numbers
ICMP	Internet control message protocol
IDCT	Inverse DCT
IDEA	International data encryption algorithm
IDFT	Inverse DFT
IDSL	ISDN-DSL
IEE	Institution of Electrical Engineers
IEEE	Institute of Electrical and Electronics Engineers

IETF	Internet Engineering Task Force
IGE	International gateway exchange
IGMP	Internet group management protocol
IGP	Interior gateway protocol
IKE	Internet key exchange
IMEI	International mobile equipment identity
IMG	IMS media gateway
IMS	IP multimedia services
INIC	Internet Network Information Center
IP	Internet protocol
IPsec	IP security
IS	Integrated services/ Intermediate system
ISDN	Integrated services digital network
ISI	Intersymbol interference
ISO	International Standards Organization
ISP	Internet service provider
ITU-T	International Telecommunications Union – Telecommunications (Sector)
IV	Initialization vector
IWF	Interworking function
IWU	Interworking unit
IXC	Interexchange carrier
JPEG	Joint Photographic Experts Group
Kc	Cipher key
ki	Authentication key
L2CAP	Logical link control and adaptation protocol
LA	Location area
LAI	Location area identification
LAN	Local area network
LAPB	Link access procedure, balanced
LAPM	Link access procedure for modems
LCN	Logical channel number
LCP	Link control protocol
LE	LAN emulation
LEC	LE client
LECS	LE configuration server
LED	Light-emitting diode
LEP	LE protocol
LES	LE server
LF	Line feed
LGN	Logical group number
LL	Link layer
LLC	Logical link control
LMDS	Local MDS
LNB/C	Low noise block/converter
LPC	Linear predictive coding
LS	Link state
LWE	Lower window edge
LXC	Local exchange carrier
LZ	Lempel–Ziv
LZW	Lempel–Ziv–Welsh
MA	Multiple access
MAC	Medium access control
MAN	Metropolitan area network
MAT	Multicast address table
M-bone	Multicast (Internet) backbone
MCM	Multi-carrier modulation
MCU	Multipoint control unit
MD	Message digest
MDS	Multipoint distribution system
MFN	Multifrequency network
MGCF	Media gateway control function
MGW	Media gateway
MIB	Management information base
MIDI	Music Instrument Digital Interface
MII	Media-independent interface
MIME	Multipurpose Internet mail extension
MIT	Management information tree
MMDS	Multichannel MDS
MMR	Modified–modified read
MMS	Multimedia message service
MO	Managed object
MOD	Movie on demand
MOSPF	Multicast OSPF
MP3	MPEG layer 3 (audio)
MPEG	Motion Picture Experts Group
MPLS	MultiProtocol Label Switching
MRFC	Media resource function controller
MRFP	Media resource function processor
MRP	Multicast routing protocol
MS	Message store/Mobile station
MSC	Mobile switching center
MSISDN	Mobile subscriber ISDN number
MSL	Maximum segment lifetime
MSRN	Mobile subscriber roaming number
MSS	MAN switching system
MTA	Message transfer agent
MUX	Multiplexer
NAK	Negative acknowledgment
NAP	Network access point
NAPT	Network address and port translation
NAV	Network allocation vector
NBS	National Bureau of Standards
NCP	Network control protocol
NEXT	Near-end crosstalk
NMOD	Near movie-on-demand
NMS	Network management system
NNTP	Network news transfer protocol
NOP	No operation
NPA	Network point of attachment
NRM	(Unbalanced) normal response mode
NRZ	Non-return to zero
NRZI	Non-return to zero inverted
NSAP	Network service access point

NSDU	Network service data unit
NSS	Network and switching subsystem
NT	Network termination
NTE	Network termination equipment
NTSC	National Television Standards Committee
NTU	Network termination unit
NVT	Network virtual terminal
OMC	Operation and maintenance center
ONU	Optical network (termination) unit
OSA	Open service access
OSC	Open service capability
OSI	Open systems interconnection
OSPF	Open shortest path first
OSS	Operation subsystem
PAL	Phase alternation line
PAWS	Protection against wrapped sequence (numbers)
PBX	Private branch exchange
PC	Personal computer
PCF	Point coordination function
PCI	Protocol connection identifier
PCI	Protocol control information
PCM	Pulse-code modulation
PDA	Personal digital assistant
PDH	Plesiochronous digital hierarchy
PDU	Protocol data unit
PEL	Picture element
PGP	Pretty good privacy
PHB	Per-hop behavior
PIFS	PCF inter-frame spacing
PIN	Personal identity number
PISO	Parallel in, serial out
Pixel	Picture element
PL	Physical layer
PMA	Parked member address
PMD	Physical medium dependent
POP	Point of presence/Post Office Protocol
POTS	Plain old telephone service
PPP	Point-to-point protocol
PRI	Primary rate interface
PS	Parked slave
PSE	Packet-switching exchange
PSK	Phase-shift keying
PSM	Protocol/service multiplexer
PSTN	Public-switched telephone network
PTE	Path termination equipment
PTT	Post, telephone, and telecommunications (authority)
PVC	Permanent virtual connection
QA	Queued arbitrated
QAM	Quadrature amplitude modulation
QCIF	Quarter CIF
QoS	Quality of service
QPDS	Queued-packet distributed-switch
QPSK	Quadrature phase shift keying
RAM	Random access meomory
RARP	Reverse ARP
RCU	Remote concentrator unit
RDN	Relative distinguished name
READ	Relative element address designate
RED	Random early detection
RF	Radio frequency
RGB	Red, green, blue
RI	Ring indication
RIP	Routing information protocol
RNC	Radio network controller
ROM	Read only memory
RP	Root port
RPC	Root path cost
RS	Record separator
RSA	Rivest, Shamir, Adelman
RSS	Radio subsystem
RSU	Remote switching unit
RSVP	Resource reservation protocol
RTC	Round-trip correction
RTCP	Real-time transport control protocol
RTD	Round-trip delay
RTO	Retransmission timeout
RTP	Real-time transport protocol
RTS	Request to send
RTSP	Real-time streaming protocol
RTT	Round-trip time
SA	Security association/Source address
SAAL	Signaling AAL
SACK	Selective ACK
SAP	Service access point
SAR	Segmentation and reassembly
SAS	Single attach station
SAT-IF	Satellite intermediate frequency
SCO	Synchronous connection-oriented link
SCP	Signaling control point
SD	Start delimiter
SDD	Service discovery database
SDH	Synchronous digital hierarchy
SDP	Session description protocol
SDSL	Single-pair DSL
SET	Secure electronic transaction
SFD	Start-of-frame delimiter
SFN	Single-frequency network
SG	Signaling gateway
SGML	Standard Generalized Mark-up Language
SGSN	Serving GPRS support node
SHC	Secure hash coding

SIF	Source intermediate format
SIFS	Short inter-frame spacing
SIG	Special Interest Group
SIM	Subscriber identity module
SIP	Session initiation protocol
SIP	SMDS interface protocol
SIPO	Serial in, parallel out
SMDS	Switched multimegabit data service
SMP	Standby monitor present
SMS	Short message service
SMT	Station management
SMTP	Simple mail transfer protocol
SNMP	Simple network management protocol
SNR	Signal-to-noise ratio
SOH	Start of heading
SONET	Synchronous optical network
SOS	Start of stream
SP	Space
SPF	Shortest-path first
SPI	Security parameter index
SRGP	Simple raster graphics package
SS	Standby slave
SS7	Signaling system number 7
SSL	Secure socket layer
SST	Slow start threshold
STB	Set-top box
STE	Section termination equipment
STM	Synchronous transport module
STP	Shielded twisted-pair
STS	Synchronous transport signal
STX	Start of text
SVCC	Signaling virtual channel connection
SVGA	Super VGA
SWS	Silly window syndrome
SYN	Synchronous idle
TA	Terminal adapter
TCM	Trellis coded modulation
TCP	Transmission control protocol
TCS	BIN telephony control specification binary
TCU	Trunk coupling unit
TDD	Time-division duplex
TDM	Time-division multiplexing
TE	Terminal equipment
TEI	Terminal endpoint identifier
TFTP	Trivial file transfer protocol
THT	Token hold time
TIFF	Tagged image file format
TIFF/EP	TIFF for electronic photography
TLS	Transport layer security
TRT	Token rotation time
TTRT	Target TRT
TU	Tributary unit
TUG	Tributary unit group
UA	Unnumbered acknowledgment/User agent
UART	Universal asynchronous receiver transmitter
UBR	Unspecified bit rate
UD	User database
UDB	User data buffer
UDP	User datagram protocol
UIP	User interface part
UMTS	Universal mobile telecommunications system
UNA	Upstream neighbor's address
UNI	User–network interface
U-NII	Unlicensed national information infrastructure
URI	Uniform resource identifier
URL	Uniform resource locator
US	Upstream (channel)
USRT	Universal synchronous receiver transmitter
UTP	Unshielded twisted-pair
UWE	Upper window edge
VBR	Variable bit rate
VC	Virtual connection
VCC	Virtual channel connection
VCI	Virtual channel identifier
VCR	Video cassette recorder
VDSL	Very high DSL
VG	Voice grade
VGA	Video graphics array
VLC	Variable-length code
VLR	Visitor location register
VOD	Video-on-demand
VOIP	Voice over IP
VOP	Video object plane
VPI	Virtual path identifier
VPN	Virtual private network
VT	Virtual terminal
WAN	Wide area network
WAP	Wireless Application Protocol
WEP	Wired equivalent privacy
WFQ	Weighted fair queuing
WPAN	Wireless personal area network
WTLS	Wireless transport layer security
WWW	World Wide Web
WYSIWYG	"What you see is what you get"
XHTML	Extended HTML
XID	Exchange identification
XML	Extensible Markup Language

data communications and networking basics

1.1 Overview

The rapid expansion in the geographical coverage and the range of applications supported by the Internet means that it is now the dominant network for most networked applications. As we can see in Figure 1.1, users gain access to the Internet and its applications in a variety of ways. Most users at home and in small offices, for example, gain access through, say, a personal computer (PC) that is connected to either a **telephone network** – that is, a public switched telephone network **(PSTN)** or an integrated services digital network **(ISDN)** – or, in some cases, a cable television **(CATV)** network. In practice, these provide only a physical connection to a second network called an **Internet service provider (ISP) network**. This, as its name implies, provides the access point to the Internet for such users and also support for a range of applications.

Similarly, users at work gain access through, say, a PC or workstation that is connected to either an office or a campus network – known as a **local area network** or **LAN** – or, for a large business/corporation, to an **enterprise-wide private network**. As we can see in the figure, these are composed of multiple

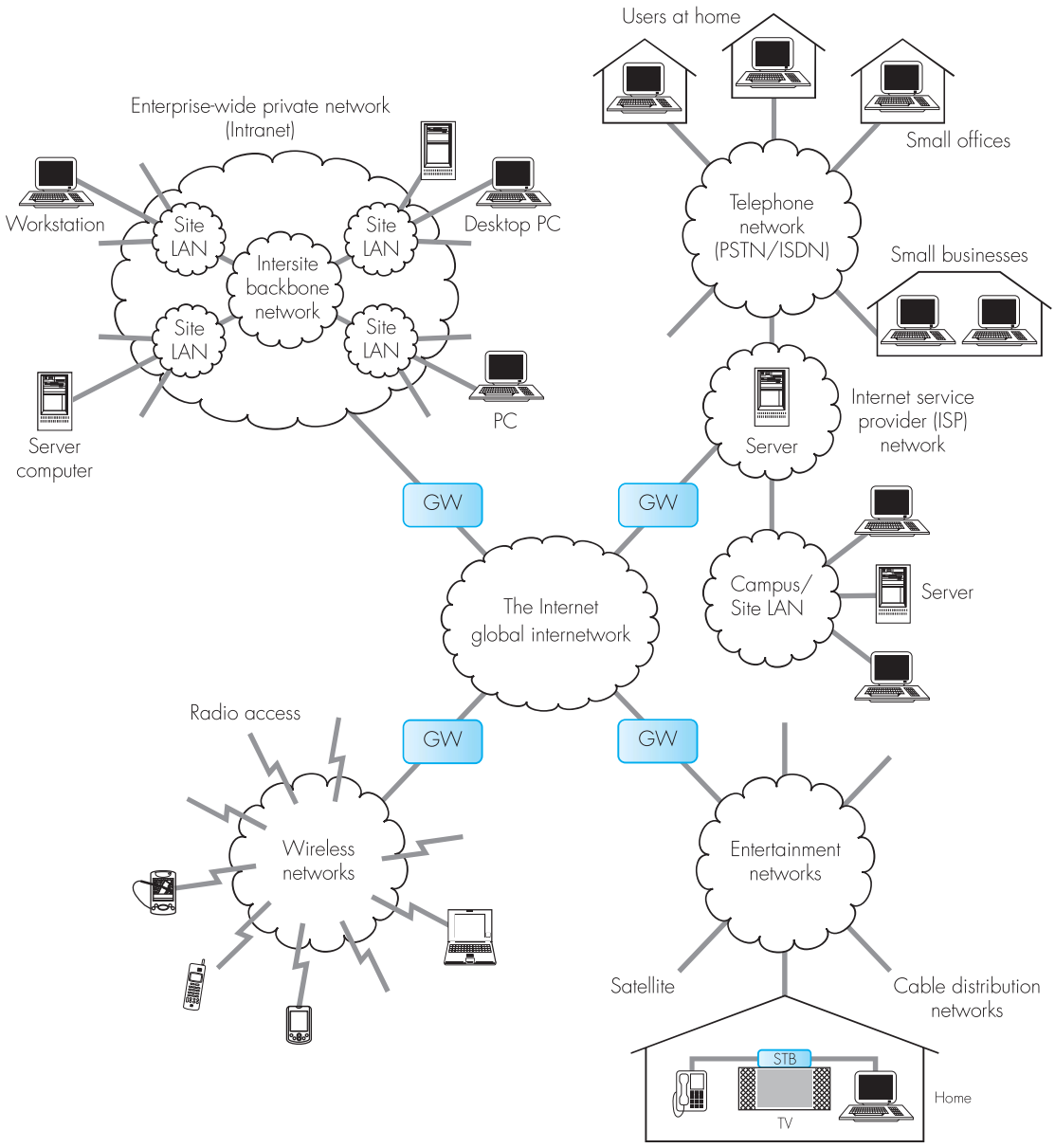


Figure 1.1 The Internet and its access networks.

site LANs that are interconnected by an inter-site backbone network. In practice, each of these networks now utilize the same operational mode and protocols as the Internet itself and hence, in the case of an enterprise network, these are known also as **intranets**.

Two more recent access networks are also shown in the figure. Users on the move, for example, can now gain access to the services of the Internet through, say, a mobile phone, a personal digital assistant (PDA) or a laptop that communicates over a radio link that is the access point to a **wireless network**. In this way, the point of attachment of the user device to the network can vary, so supporting user mobility. Secondly, in addition to providing basic radio and television broadcasts, **entertainment networks** now provide an access point to the Internet and hence access to a much richer set of applications.

In many instances, the expanding range of applications supported by the Internet has come about through the technological advances in the way the user information associated with these applications is represented. For example, until recently, the various access networks shown in the figure, in addition to providing their basic service such as telephony, only supported applications in which the user information was composed of text comprising strings of alphanumeric characters. As a result of the technological advances in the area of **compression**, however, the same access networks can now support a much richer set of applications involving multiple media types. These include not only text but digitized images, photos/pictures, speech, video and audio.

As we can conclude from this brief overview, to study the technological issues relating to the Internet and its applications requires an in-depth understanding not only of the operation of the Internet itself but also of the operation of the different types of access network and how they interface with the Internet. In addition, because many of the applications involve the transfer over a network of sensitive information, the topic of **security** is now essential when describing the structure of the information being transferred. The aim of this book is to provide this body of knowledge.

In Chapter 1, an introduction to the subjects of data communications and computer networking is given. This includes a definition of the various terms that are used and the operational characteristics of the different types of network. In addition, it presents a detailed introduction to the subject of data communications which forms the foundation of all computer networks. Chapters 2–5 are then devoted to describing the operation of the different types of access network that are used and how they interface with the global Internet. This is then followed in Chapters 6–10 by a detailed description of the protocols and applications of the Internet including a separate chapter devoted to the subject of security. There are five appendices that give an introduction to the subjects of multimedia data representation and compression, error detection methods, forward error control, radio propagation and transmission, and ATM networks.

1.2 Application and networking terminology

Before we describe the operation of the different types of access networks, it will be helpful if we first review some of the terminology used in relation to the different types of data that are used in Internet applications. Also, we describe the terminology and operational characteristics of the different types of communication channels that are provided by the various types of access network.

1.2.1 Data types and their characteristics

As we show in Figure 1.2, there are four basic types of data: text, images, video and audio. We shall discuss each separately.

Text

Essentially, there are three types of text that are used to produce pages of documents:

- **unformatted text**: this is known also as **plaintext** and enables pages to be created which comprise strings of fixed-sized characters from a limited character set;
- **formatted text**: this is known also as **richtext** and enables pages and complete documents to be created that are comprised of strings of characters of different styles, size and shape with tables, graphics and images inserted at appropriate points;
- **hypertext**: this enables Web pages to be created in the form of an integrated set of documents (each comprising formatted text) with defined linkages between them.

Images

There are two types of images:

- **computer-generated images**: more generally referred to as computer graphics or simply graphics;
- **digitized images**: these include digitized documents and digitized pictures/photos.

Ultimately, both types of image are displayed and printed in the form of a two-dimensional matrix of individual picture elements known as **pixels**. In the case of continuous-tone monochromatic images – such as a printed picture or scene – good quality black-and-white pictures can be obtained with 8 bits per pixel – that is, 256 levels of grey. For color images, each pixel may

contain 8, 16, 24 or more bits to produce a good range of colors. Typical screen sizes are 640×480 (VGA) and 1024×768 (SVGA). With 8 bits per pixel the VGA format requires 307.2 kbytes to store an image and with 24 bits per pixel the SVGA format requires 2.359296 Mbytes.

Video

Most video cameras now operate digitally and generate a sequence of digitized images, each of which is called a **frame**. With broadcast television, however, the video signal is in an analog form and hence the source video signal must first be converted into a digital form. To do this, there are a number of what are called **digitization formats** used, each of which is targeted at a particular screen size. Then, to obtain a moving picture, the frames are stored/played-out at a rate determined by the digitization format that is used. For example, the raw bit rate of digital television is 162 Mbps and that for a small-screen picture phone is 40 Mbps.

Audio

All audio signals are generated in an **analog** form; that is, the signal that is generated by a microphone varies continuously with time as the amplitude of the speech/audio varies. Hence before an audio signal can be stored in a computer and integrated with the other media types, it must first be converted into a digital form. This is done using an electronic circuit called an **analog-to-digital converter (ADC).** This involves sampling the amplitude of the speech/audio signal at regular time intervals; the higher the sampling rate, the better quality is the reproduced analog signal. The reverse conversion is done using a circuit called a **digital-to-analog converter** (**DAC**) and the reproduced signal is played out via speakers. In the case of a speech signal – as used in telephony – a typical sampling rate is 8 kHz with 8 bits per sample. This yields a digital signal of 64 kbps, which is the bit rate used in telephone networks. For CD-quality audio, however, a common sampling rate is 44.1 kHz with 16 bits per sample, which produces a bit rate of 705.6 kbps or, for stereo, 1.411 Mbps.

As we can deduce from these figures – and especially those for video and audio – the raw transmission bit rate required of a communications channel far exceeds the available capacity of most network types. Hence, as we show in Figure 1.2, in almost all Internet applications the source signals are first compressed using a suitable **compression algorithm**. As an introduction to the subject, we discuss various compression algorithms in Appendix A.

As we show in the figure, the flow of data associated with the different applications can be either continuous or block-mode. In the case of **continuous data**, this means that the data stream is generated by the source continuously in a time-dependent way. In general, therefore, continuous data is passed directly to the destination as it is generated and, at the destination, the data stream is played out directly as it is received. This mode of operation is called **streaming** and, since continuous data is generated in a time-dependent way, it is also known as **real-time data**. With continuous data, therefore,

the bit rate of the communications channel that is used must be compatible with the rate at which the (compressed) source data is being generated. Two examples of media types that generate continuous streams of data in real time are audio and video.

In terms of the bit rate at which the source data stream is generated, this may be at either a **constant bit rate** (**CBR**) or a **variable bit rate** (**VBR**). With audio, for example, the digitized audio stream is generated at a constant bit rate. In the case of video, however, although the individual pictures/frames that make up the video are generated at a constant rate, after compression the amount of data associated with each frame varies. In general, therefore, the data stream associated with compressed video is generated at fixed time intervals but the resulting bit rate is variable.

In the case of **block-mode data**, the source data comprises one or more blocks of data that is/are created in a time-independent way, for example, a

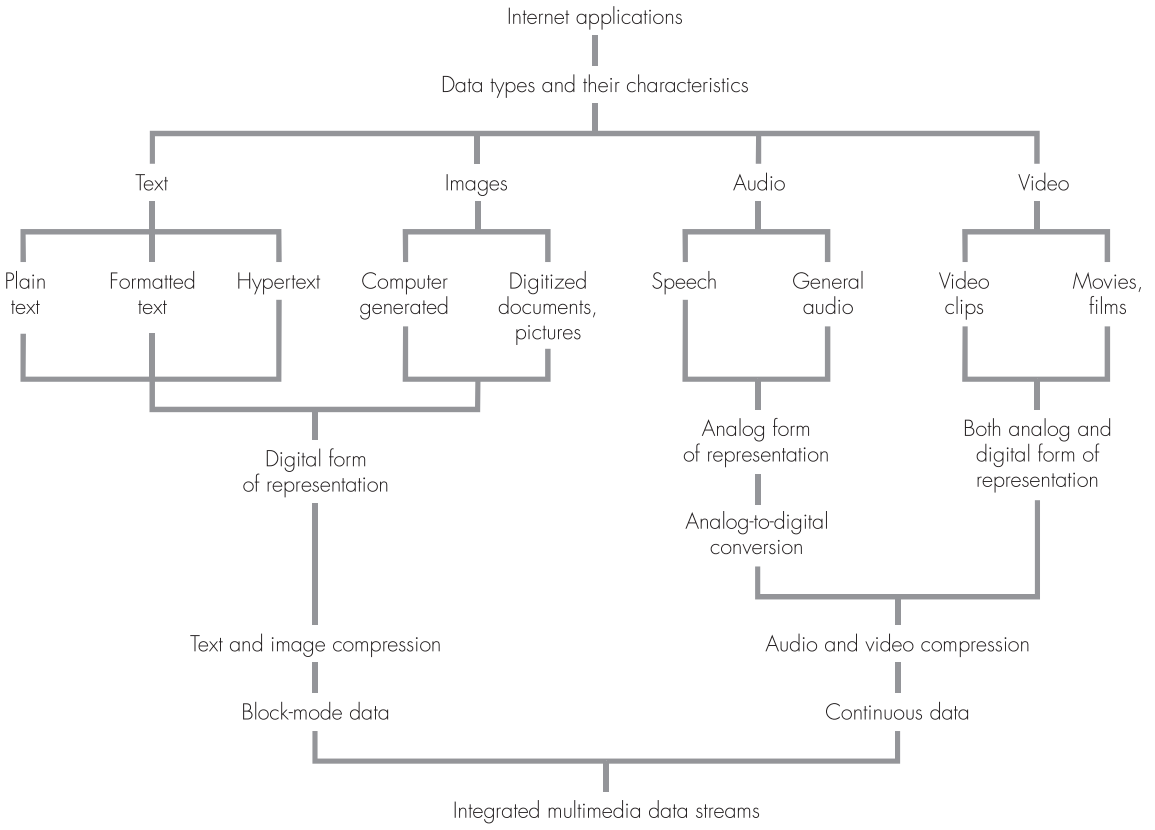


Figure 1.2 Data types used in Internet applications.

block of text representing an e-mail, a two-dimensional matrix of pixel values that represents an image, and so on. Normally, therefore, block-mode data is created in a time-independent way and is often stored at the source in, say, a file. Then, when it is requested, the blocks of data are transferred across the network to the destination where they are again stored and subsequently output/displayed at a time determined by the requesting application program. This mode of operation is known as **downloading** and, as we can deduce from this, with block-mode data the bit rate of the communications channel need not be constant but must be such that, when a block is requested, the delay between the request being made and the contents of the block being output at the destination is within an acceptable time interval. This is known as the **round-trip delay** (**RTD**) and, for human–computer interaction, ideally should be no more than a few seconds.

1.2.2 Data communications and networking terminology

A selection of the terms relating to data communications and computer networking is shown in Figure 1.3. In terms of the communication modes and channels that are provided by the various network types these are illustrated in Figure 1.4. As we show in the figure, the transfer of the data streams associated with an application can take place in one of five modes:

- **simplex**: this means the data associated with the application flows in one direction only. An example is the transmission of photographic images from a deep-space probe at predetermined times since this involves just a unidirectional flow of data from the probe to an earth station;
- **half-duplex**: this means that data flows in both directions but alternately. This mode is also known as **two-way alternate** and an example is a user making a request for some data from a remote server and the latter returning the requested data;
- **duplex**: this means that data flows in both directions simultaneously. It is also known as **two-way simultaneous** and an example is the two-way flow of digitized speech associated with a telephony application;
- **broadcast**: this means that the data output by a single source device is received by all the other devices – computers, and so on – that are connected to the same network. An example is the broadcast of a television program over a cable network as all the television receivers that are connected to the network receive the same set of programs;
- **Multicast**: this is similar to a broadcast except that the data output by the source is received by only a specific subset of the devices that are connected to the network. The latter form what is called a **multicast group** and an example application is videoconferencing, which involves a predefined group of terminals/computers connected to a network exchanging integrated speech and video streams.

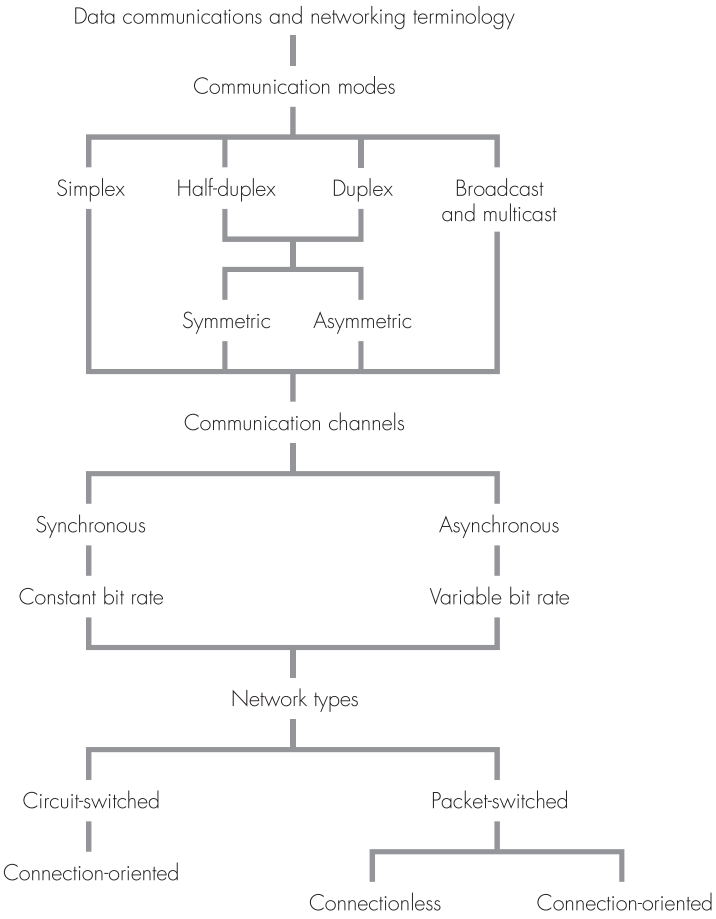


Figure 1.3 Data communications and networking terminology.

In the case of half-duplex and duplex communications, the bit rate associated with the flow of data in each direction can be either equal or different; if the flows are equal, the data flow is said to be **symmetric**, and if the flows are different, **asymmetric**. For example, a video telephone call involves the exchange of an integrated digitized speech and video stream in both directions simultaneously and hence a symmetric duplex communications channel is required. Alternatively, in an application involving a browser (program) and a Web server, a low bit rate channel from the browser to the Web server is required for request and control purposes and a higher bit rate channel from the server to the subscriber for the transfer of, say, the requested Web page. Hence for this type of application, an asymmetric half-duplex communications channel is sufficient.

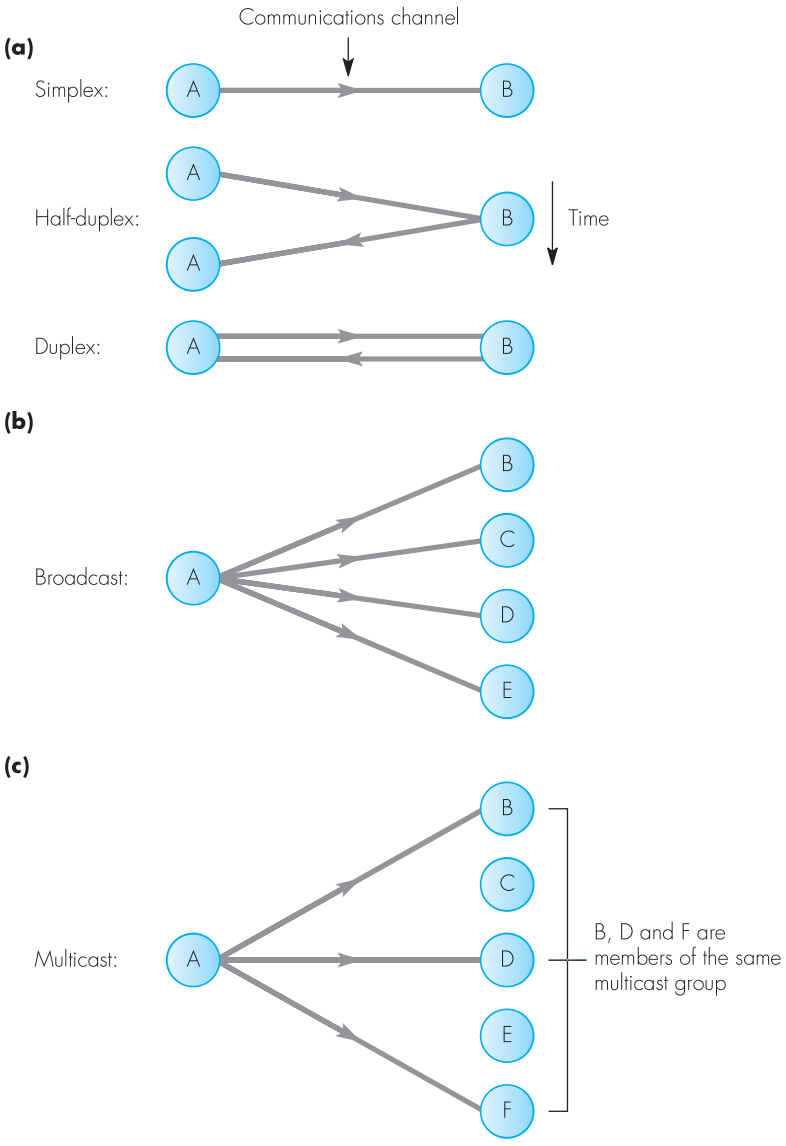


Figure 1.4 Communication modes: (a) unicast; (b) broadcast; (c) multicast.

1.2.3 Network types

In the same way that there are two types of data stream associated with the different media types – continuous and block-mode – so there are two types of

communications channel associated with the various network types: one that operates in a time-dependent way known as **circuit-mode** and the other in a time-varying way known as **packet-mode**. The first is known as a **synchronous communications channel** since it provides a constant bit rate service at a specified rate. The second is known as an **asynchronous communications channel** since it provides a variable bit rate service, the actual rate being determined by the (variable) transfer rate of packets across the network.

Circuit-switched

A circuit-switched network is shown in Figure 1.5 and, as we can see, it comprises an interconnected set of **switching offices/exchanges** to which the subscriber terminals/computers are connected. This type of network is known as a **circuit-switched network** and, prior to sending any data, the source must first set up a connection through the network. Each subscriber terminal/computer has a unique network-wide number/address associated with it and, to make a call, the source first enters the number/address of the intended communication partner. The local switching office/exchange then uses this to set up a connection through the network to the switching office/exchange to which the destination is connected and, assuming the destination is free and ready to receive a call, a message is returned to the source indicating that it can now start to transfer/exchange data. Finally, after all the data has been transferred/exchanged, either the source or the destination requests for the connection to be cleared. The bit rate associated with the connection is fixed and, in general, is determined by the bit rate that is used over the access circuits that connect the source and destination terminal/computer to the network.

The messages associated with the setting up and clearing of a connection are known as **signaling messages**. As we can deduce from the above, with a circuit-switched network there is a time delay while a connection is being established. This is known as the **call/connection setup delay** and two examples of networks that operate in this way are a PSTN and an ISDN. With a PSTN, the call setup delay can range from a fraction of a second for a local call through to several seconds for an international call. With an ISDN, however, the delay ranges from tens of milliseconds through to several hundred milliseconds.

Packet-switched

As we see in Figure 1.6, there are two types of packet-switched network: **connection-oriented** (**CO**) and **connectionless** (**CL**). The principle of operation of a connection-oriented network is shown in Figure 1.6(a) and, as we can see, it comprises an interconnected set of **packet-switching exchanges** (**PSEs**). This type of network is known as a **packet-switched network** and, as with a circuit-switched network, each terminal/computer that is connected to the network has a unique network-wide number/address associated with it. With a connection-oriented network, as the name implies, prior to sending any data, a connection is first set up through the network using the addresses

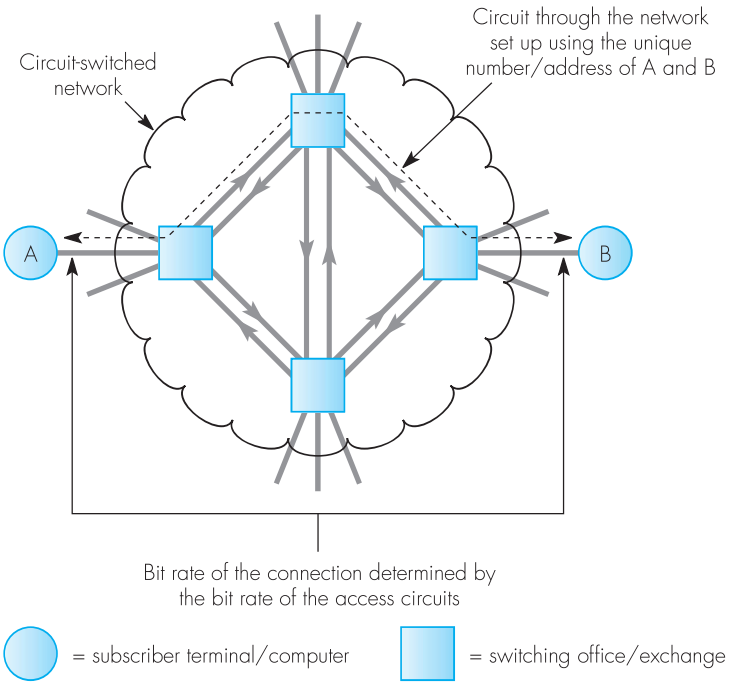


Figure 1.5 Circuit-switched network schematic.

of the source and destination terminals. However, in a packet-switched network, the connection/circuit that is set up utilizes only a variable portion of the bandwidth of each link and hence the connection is known as a **virtual connection** or, more usually, a **virtual circuit** (**VC**).

To set up a VC, the source terminal/computer sends a *call request* control packet to its local PSE which contains, in addition to the address of the source and destination terminal/computer, a short identifier known as a **virtual circuit identifier** (**VCI**). Each PSE maintains a table that specifies the outgoing link that should be used to reach each network address and, on receipt of the *call request* packet, the PSE uses the destination address within the packet to determine the outgoing link to be used. The next free identifier (VCI) for this link is then selected and two entries are made in a **routing table**. The first specifies the incoming link/VCI and the corresponding outgoing link/VCI; the second, in order to route packets in the reverse direction, is the inverse of these, as we show in the example in the figure. The *call request* packet is then forwarded on the selected outgoing link and the same procedure is followed at each PSE along the route until the destination terminal/computer is reached.

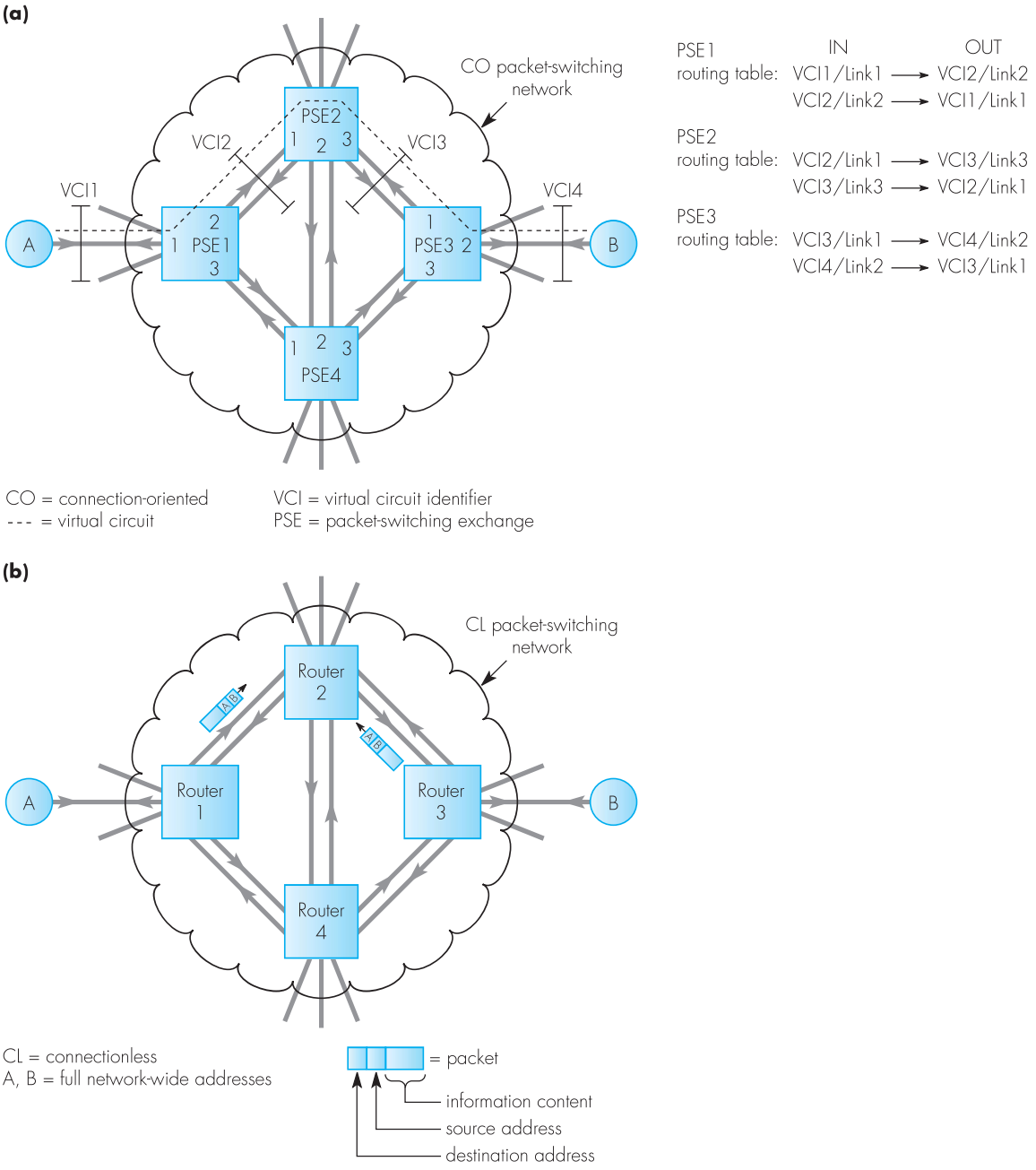


Figure 1.6 Packet-switching network principles: (a) connection-oriented; (b) connectionless.

Collectively, the VCIs that are used on the various links form the virtual circuit and, at the destination, assuming the call is accepted, a *call accepted* packet is returned to the source over the same route/virtual circuit. The data transfer phase can then start but, since a VC is now in place, only the VCI is needed in the packet header instead of the full network-wide address. Each PSE first uses the incoming link/VCI to determine the outgoing link/VCI from the routing table. The existing VCI in the packet header is then replaced with that obtained from the routing table and the packet is forwarded on the identified outgoing link. The same procedure is followed to return data in the reverse direction and, when all the data has been transferred/exchanged, the VC is cleared and the appropriate VCIs are released by passing a *call clear* packet along the VC.

In contrast, with a connectionless network, the establishment of a connection is not required and the two communicating terminals/computers can communicate and exchange data as and when they wish. In order to do this, however, as we show in Figure 1.6(b), each packet must carry the full source and destination addresses in its header in order for each PSE to route the packet onto the appropriate outgoing link. In a connectionless network, therefore, the term **router** is normally used rather than packet-switching exchange.

In both network types, as each packet is received by a PSE/router on an incoming link, it is stored in its entirety in a memory buffer. A check is then made to determine if any transmission/bit errors are present in the packet header – that is, the signal that is used to represent a binary 0 is corrupted and is interpreted by the receiver as a binary 1 and vice versa – and, if an error is detected, the packet is simply discarded. The service offered by a connectionless network is said, therefore, to be a **best-effort service**. If no errors are detected then the addresses/VCIs carried in the packet header are read to determine the outgoing link that should be used and the packet is placed in a queue ready for forwarding on the selected outgoing link. All packets are transmitted at the maximum link bit rate. With this mode of operation, however, it is possible for a sequence of packets to be received on a number of incoming links all of which need forwarding on the same outgoing link. Hence a packet may experience an additional delay while it is in the output queue for a link waiting to be transmitted. Clearly, this delay will be variable since it depends on the number of packets that are currently present in the queue when a new packet arrives for forwarding. This mode of operation is known as (packet) **store-and-forward** and, as we can see, there is a packet store-and-forward delay in each PSE/router. The sum of the store-and-forward delays in each PSE/router contributes to the overall transfer delay of the packet across the network. The mean of this delay is known as the **mean packet transfer delay** and the variation about the mean the **delay variation** or **jitter**.

An example of a packet-switched network that operates in the connectionless mode is the Internet, which we shall describe in some detail in Chapter 6. Two examples of networks that operate in the connection-oriented mode are

the international X.25 packet-switching network and asynchronous transfer mode (ATM) networks. The X.25 network is used primarily for the transfer of large files containing text and binary data between large computers that belong to a collection of banks for interbank transfers. In contrast, ATM networks have been designed from the outset to support high data transfer rates. This is achieved by using high bit rate interconnecting links and, once a virtual circuit has been set up, a very small fixed-sized packet of 53 bytes is used to transfer the data over the link. Each small packet is known as a **cell** and includes a short 5-byte header that enables cells to be switched at the very high link bit rates that are used. It is for this reason that ATM networks are also known as **fast packet-switching networks** or sometimes **cell-switching networks**. Because of the very high switching rates of ATM switches, they are used extensively in the core networks of both the PSTN/ISDN and the Internet and many wireless networks.

1.2.4 Network QoS

The operational parameters associated with a digital communications channel through a network are known as the **network Quality of Service** (**QoS**) **parameters** and collectively they determine the suitability of the channel in relation to its use for a particular application. In practice, the QoS parameters associated with a circuit-switched network are different from those associated with a packet-switched network and hence we shall discuss each separately.

Circuit-switched network

The QoS parameters associated with a constant bit rate channel that is set up through a circuit-switched network include:

- the bit rate,
- the mean bit error rate,
- the transmission delay.

The mean **bit error rate** (**BER**) of a digital channel is the probability of a binary bit being corrupted during its transmission across the channel over a defined time interval. Hence, for a constant bit rate channel, this equates to the probability of a bit being corrupted in a defined number of bits. A mean BER of 10^{-3}, therefore, means that, on average, for every 1000 bits that are transmitted, one of these bits will be corrupted. In some applications, providing the occurrence of bit errors is relatively infrequent, their presence is acceptable while in other applications it is imperative that no residual bit errors are present in the received data. For example, if the application involves speech, then an occasional bit error will go unnoticed but in an application involving the transfer of, say, financial information, it is essential that the received information contains no errors. Hence with such applications, prior to

transmission the source data is normally divided into blocks the maximum size of which is determined by the mean BER of the communications channel.

For example, if the mean BER of the channel is 10^{-3}, then the number of bits in a block must be considerably less than 1000 otherwise, on average, every block will contain an error and will be discarded. Normally, however, bit errors occur randomly and hence, even with a block size of, say, 100 bits, blocks may still contain an error but the probability of this occurring is considerably less. In general, if the BER probability is P and the number of bits in a block is N, then, assuming random errors, the probability of a block containing a bit error, P_B, is given by:

$$P_B = 1 - (1 - P)^N$$

which approximates to $N \times P$ if $N \times P$ is less than 1.

In practice, most networks – both circuit-switched and packet-switched – provide an **unreliable service** which is also known as a **best-try** or **best-effort** service. This means that any blocks containing bit errors will be discarded either within the network – packet-switched networks – or in the network interface at the destination – both packet-switched and circuit-switched networks. Hence if the application dictates that only error-free blocks are acceptable, it is necessary for the sending terminal/computer to divide the source information into blocks of a defined maximum size and for the destination to detect when a block is missing. When this occurs the destination must request that the source send another copy of the missing block. The service offered is then said to be a **reliable service**. Clearly, this will introduce a delay so the retransmission procedure should be invoked relatively infrequently, which dictates a small block size. This, however, leads to high overheads since each block must contain the additional information that is associated with the retransmission procedure. Normally, therefore, the choice of block size is a compromise between the increased delay resulting from a large block size – and hence retransmissions – and the loss of transmission bandwidth resulting from the high overheads of using a smaller block size.

Example 1.1

Derive the maximum block size that should be used over a channel which has a mean BER probability of 10^{-4} if the probability of a block containing an error – and hence being discarded – is to be 10^{-1}.

Answer:

$$P_B = 1 - (1 - P)^N$$

Hence $0.1 = 1 - (1 - 10^{-4})^N$ and $N = 950$ bits

Alternatively, $P_B = N \times P$

Hence $0.1 = N \times 10^{-4}$ and $N = 1000$ bits

The **transmission delay** associated with a channel is determined not only by the bit rate that is used but also by delays that occur in the terminal/computer network interfaces (known as **codec delays**), plus the propagation delay of the digital signals as they pass from the source to the destination across the network. This is determined by the physical separation of the two communicating devices and the velocity of propagation of a signal across the transmission medium. In free space, for example, the latter is equal to the speed of light ($3\times10^8\,m\,s^{-1}$) while it is a fraction of this in physical media, a typical value being $2\times10^8\,m\,s^{-1}$.

Notice that the propagation delay in each case is independent of the bit rate of the communications channel and, assuming the codec delay remains constant, is the same whether the bit rate is 1 kbps, 1 Mbps, or 1 Gbps.

Packet-switched network

The QoS parameters associated with a packet-switched network include:

- the maximum packet size,
- the mean packet transfer rate,
- the mean packet error rate,
- the mean packet transfer delay,

Example 1.2

Determine the propagation delay associated with the following communication channels:

(i) a connection through a private telephone network of 1 km,

(ii) a connection through a PSTN of 200 km,

(iii) a connection over a satellite channel of 50 000 km.

Assume that the velocity of propagation of a signal in the case of (i) and (ii) is $2\times10^8\,m\,s^{-1}$ and in the case of (iii) $3\times10^8\,m\,s^{-1}$.

Answer:

Propagation delay T_p = physical separation/velocity of propagation

(i) $$T_p = \frac{10^3}{2\times10^8} = 5\times10^{-6}\,s$$

(ii) $$T_p = \frac{200\times10^3}{2\times10^8} = 10^{-3}\,s$$

(iii) $$T_p = \frac{5\times10^7}{3\times10^8} = 1.67\times10^{-1}\,s$$

- the worst-case jitter,
- the transmission delay.

In a packet-switched network, although the rate at which packets are transferred across the network is influenced strongly by the bit rate of the interconnecting links, because of the variable store-and-forward delays in each PSE/router, the actual rate of transfer of packets across the network is also variable. Hence the **mean packet transfer rate** is a measure of the average number of packets that are transferred across the network per second and, coupled with the packet size being used, determines the equivalent mean bit rate of the channel.

The **mean packet error rate** or **PER** is the probability of a received packet containing one or more bit errors. It is the same, therefore, as the block error rate associated with a circuit-switched network which we derived in the previous section. Hence it is related to both the maximum packet size and the worst-case BER of the transmission links that interconnect the PSEs/routers that make up the network.

We defined the meaning of the term "mean packet transfer delay" in Section 1.2.3 when we described the operation of packet-mode networks. It is the summation of the mean store-and-forward delay that a packet experiences in each PSE/router that it encounters along a route and the term "jitter" is the worst-case variation in this delay. As we just explained, the transmission delay is the same whether the network operates in a packet mode or a circuit mode and includes the codec delay in each of the two communicating computers and the signal propagation delay.

1.2.5 Application QoS

The network QoS parameters define what the particular network being used provides rather than what the application requires. The application itself, however, also has QoS parameters associated with it. In an application involving images, for example, the parameters may include a minimum image resolution and size, while in an application involving video, the digitization format and refresh rate may be defined. The application QoS parameters that relate to the network include:

- the required bit rate or mean packet transfer rate,
- the maximum startup delay,
- the maximum end-to-end delay,
- the maximum delay variation/jitter,
- the maximum round-trip delay.

For applications involving the transfer of a constant bit rate stream, the important parameters are the required bit rate/mean packet transfer rate,

the end-to-end delay, and, equally important, the delay variation/jitter since this can cause problems in the destination decoder if the rate of arrival of the bitstream is variable. For interactive applications, however, the **startup delay** defines the amount of time that elapses between an application making a request to start a session and the confirmation being received from the application at the destination – a server, for example – that it is prepared to accept the request. Hence this includes, in addition to the time required to establish a network connection – if this is required – the delay introduced in both the source and the destination computers while negotiating that the session can take place. As we saw earlier in Section 1.2.1, the round-trip delay is important because, for human–computer interaction to be successful, the delay between a request for some information being made and the start of the information being received/displayed should be as short as possible and, ideally, should be less than a few seconds.

As we can see from the above, for applications that involve the transfer of a constant bit rate stream, a circuit-switched network would appear to be most appropriate since, firstly, the call setup delay is often not important and secondly, the channel provides a constant bit rate service of a known rate. Conversely, for interactive applications, a connectionless packet-switched network would appear to be most appropriate since with this there is no network call setup delay and any variations in the packet transfer delay are not important.

An example application that illustrates the benefits of a packet-switched network over a circuit-switched network is the transfer of a large file of data from a server computer connected to the Internet to a client PC/workstation in a home. As we showed earlier in Figure 1.1, access to the Internet from home can be by means of a PSTN (with a modem), an ISDN connection, or a cable modem. In the case of a PSTN and an ISDN, these operate in a circuit-switched mode and provide a constant bit rate channel of in the order of 28.8 kbps (PSTN with modem) and 64/128 kbps (ISDN). In contrast, cable modems operate in a packet mode and, as we shall see later in Section 5.2.1, the modems in each of the homes in a cable region time-share the use of a single high bit rate channel/circuit. A typical bit rate of the shared channel is 27 Mbps and the number of concurrent users of the channel may be several hundred. Hence, assuming 270 concurrent users, each user would get a mean data rate of 100 kbps.

With this type of application, however, the main parameter of interest is not the mean data/bit rate but the time to transmit the complete file. With a PSTN and an ISDN, this is directly related to the channel bit rate and the size of the file. With a cable modem, however, although they time-share the use of the 27 Mbps channel, when they gain access to it, the file transfer takes place at the full rate. Hence assuming the file size is 100 Mbits, the minimum time to transmit the file using the different Internet access modes is:

PSTN and 28.8 kbps modem:	57.8 minutes
ISDN at 64 kbps:	26 minutes
ISDN at 128 kbps:	13 minutes
cable modem at 27 Mbps:	3.7 seconds

In the case of a cable modem, if other transfer requests occur during the time the file is being transmitted, then the completion time of each transfer request will increase as they share the use of the channel. Nevertheless, with this type of application, the probability of multiple users requesting a transfer in this short window of time is relatively low.

In many instances, however, this does not mean that the alternative network types cannot be used. For interactive applications, for example, the call setup delay with an ISDN network – and a PSTN for local calls – is very fast and, for many applications, quite acceptable. Similarly, for constant bit rate applications, providing the equivalent mean bit rate provided by the network is greater than the input bit rate and the maximum jitter is less than a defined value, then a packet-switched network can be used. To overcome the effect of jitter a technique known as **buffering** is used, the general principles of which are shown in Figure 1.7.

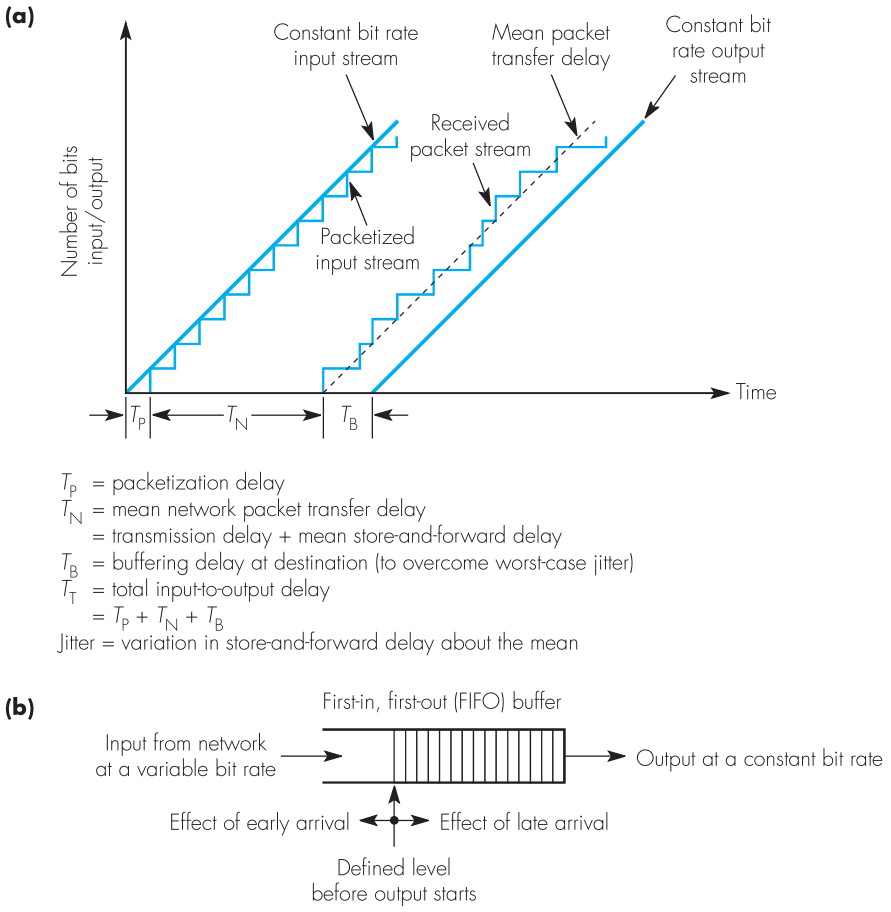


Figure 1.7 Transmission of a constant bit rate stream over a packet-switched network: (a) timing schematic; (b) FIFO buffer operation.

As we show in the figure, the effect of jitter is overcome by retaining a defined number of packets in a memory buffer at the destination before playout of the information bitstream is started. The memory buffer operates using a first-in, first-out (FIFO) discipline and the number of packets retained in the buffer before output starts is determined by the worst-case jitter and the bit rate of the information stream. However, as we show in part (a) of the figure, when using a packet-switched network for this type of application, an additional delay is incurred at the source as the information bitstream is converted into packets. This is known as the **packetization delay** and adds to the transmission delay of the channel. Hence in order to minimize the overall input-to-output delay, the packet size used for an application is made as small as possible but of sufficient size to overcome the effect of the worst-case jitter.

Example 1.3

A packet-switched network with a worst-case jitter of 10 ms is to be used for a number of applications each of which involves a constant bit rate information stream. Determine the minimum amount of memory that is required at the destination and a suitable packet size for each of the following input bit rates. It can be assumed that the mean packet transfer rate of the network exceeds the equivalent input bit rate in each case.

(i) 64 kbps

(ii) 256 kbps

(iii) 1.5 Mbps.

Answer:

(i) At 64 kbps, 10 ms = 640 bits
Hence choose a packet size of, say, 800 bits with a FIFO buffer of 1600 bits – two packets – and start playout of the bitstream after the first packet has been received.

(ii) At 256 kbps, 10 ms = 2560 bits
Hence choose a packet size of, say, 2800 bits with a FIFO buffer of 4800 bits.

(iii) At 1.5 Mbps, 10 ms = 15000 bits
Hence choose a packet size of, say, 16 000 bits with a FIFO buffer of 32 000 bits.

Notice that if the computed packet size exceeds the network maximum packet size, then the equivalent number of packets must be sent before playout starts. For example, if the maximum network packet size was 8000 bits, then for case (iii) above playout would not start until two packets have been received and the FIFO buffer should hold four packets.

In order to simplify the process of determining whether a particular network can meet the QoS requirements of an application, a number of standard application **service classes** have been defined. Associated with each service class is a specific set of QoS parameters and a network can either meet this set of parameters or not. Also, for networks that support a number of different service classes – the Internet for example – in order to ensure the QoS parameters associated with each class are met, the packets relating to each class are given a different **priority**. It is then possible to treat the packets relating to each class differently.

For example, as we shall see in Chapter 6, in the Internet, packets relating to multimedia applications involving real-time streams are given a higher priority than the packets relating to applications such as e-mail. Typically, packets containing real-time streams such as audio and video are also more sensitive to delay and jitter than the packets containing textual information. Hence during periods of network congestion, the packets containing real-time streams are transmitted first. Packets containing video are more sensitive to packet loss than packets containing audio and hence are given a higher priority.

1.3 Digital communications basics

Associated with all the access networks that are used to support Internet applications is a standard network interface to which all the end systems/hosts/stations that are attached to the network must adhere. Hence in each end system is a **network interface card** (**NIC**) – consisting of hardware controlled by associated software – that performs the related network interface functions. In this section we describe these functions.

In the last section we found that, in general, the integrated information stream generated by the various applications is a series of blocks of binary data of varying size. In some instances, the application involves a dialog using these blocks while in others it involves the transfer of a stream of blocks. In terms of the network interface, however, the data relating to the different applications that is to be transferred over the network is treated simply as a string of one or more blocks containing what is referred to as (network) user data.

Although the various networks that are used operate in different ways, the physical and link layers of both the network interface standards and the internal network standards have a number of common features associated with them. So before we describe the operation of the various networks and their interfaces, we shall discuss in this section the basic principles associated with digital communications that are common for all networks.

The access lines (and the internal transmission lines used within the various networks) all use bit-serial transmission. In general, therefore, the signal output by the NIC simply varies between two voltage levels ($+V$ and $-V$ for example) at a rate determined by the transmission bit rate. This mode of transmission is known as **baseband transmission** and is illustrated in Figure 1.8(a).

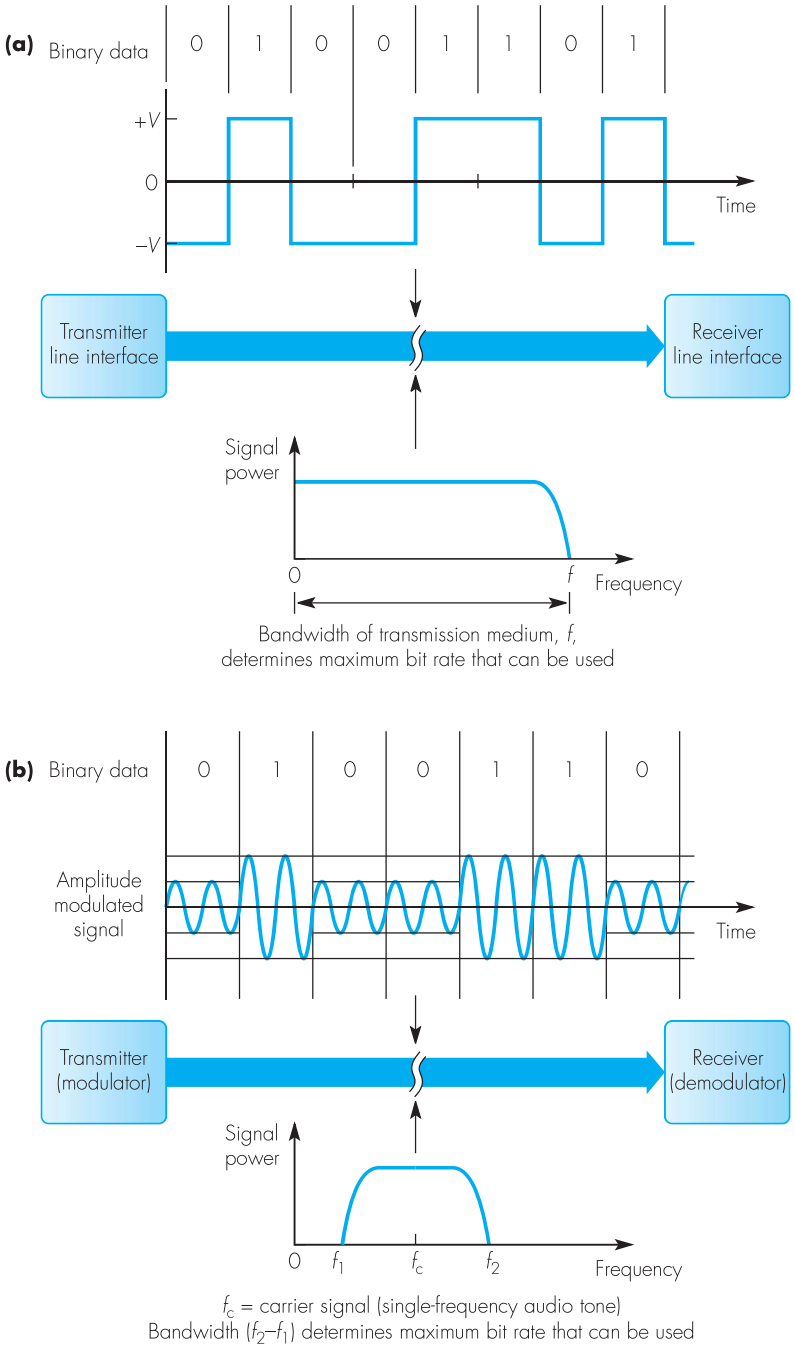


Figure 1.8 Modes of transmission: (a) baseband transmission; (b) modulated transmission.

Thus with networks that provide a digital interface, such as a LAN and an ISDN, baseband transmission is also used over the access lines to the network. With networks such as a PSTN, however, analog transmission is used over the access lines since, as we shall expand upon in Section 2.2.1, the presence of a transformer in the speech path means a DC signal – for example a long string of binary 0s or 1s – will not be discernible. The bandwidth used over these lines is that of a speech signal and is from 200 Hz through to 3400 Hz. Hence as we show in Figure 1.8(b), prior to transmitting the baseband signal output by the NIC, it is necessary first to convert this signal into an analog signal within the same bandwidth that is used for speech. This is done, for example, by modulating – also known as mixing or multiplying – a single-frequency audio signal/tone – chosen from within the bandwidth used for a speech signal and known as the **carrier signal** – by the binary signal to be transmitted. This mode of transmission is known as **modulated transmission** and the unit that performs the modulation and demodulation functions is a **modem**.

We shall describe modems and this mode of transmission further in Section 2.2.2. It should be noted, however, that even though modulated transmission is sometimes used over the access lines to the network – and also within the various types of broadcast entertainment networks – normally, the output from the NIC within each end system is a baseband signal.

When transmitting any type of electrical signal over a transmission line, the signal is **attenuated** (decreased in amplitude) and **distorted** (misshapen) by the transmission medium. Also, present with all types of transmission medium is an electrical signal known as **noise**. The amplitude of the noise signal varies randomly with time and adds itself to the electrical signal being transmitted over the line. The combined effect is that at some stage, the receiver is unable to determine from the attenuated received signal with noise whether the transmitted signal was a binary 1 or 0. An example showing the combined effect of attenuation, distortion, and noise on a baseband signal is shown in Figure 1.9.

In practice, the level of signal impairment is determined by:

- the type of transmission medium,
- the length of the transmission medium,
- the bandwidth of the medium,
- the bit rate of the data being transmitted.

We shall discuss the characteristics of the different types of transmission media in the following subsection.

As we can see in Figure 1.9, the received signal is at its peak amplitude in the centre of each **bit cell period**. Hence in order for the receiving electronics to determine the signal level (and hence bit) present on the line during each bit cell period, it endeavors to sample the received signal at the center of each bit cell. When the receiver is doing this, it is said to be in **bit**

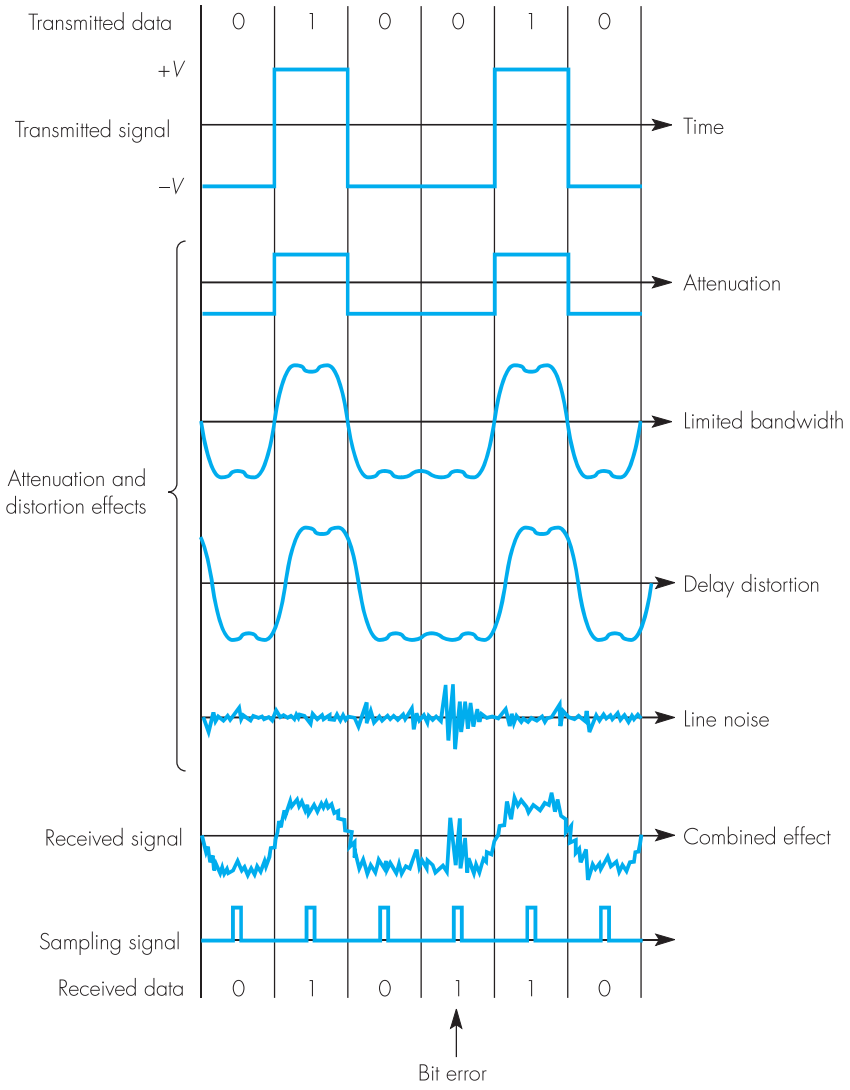


Figure 1.9 Sources of signal impairment.

synchronism with the incoming bitstream. However, although the receiver knows the nominal bit rate being used by the transmitter to transmit the bitstream, the receiver electronics operates quite independently of the transmitter, with the effect that the receiver clock used to sample the signal will be slightly different from that used at the transmitter. In practice, therefore, sampling the signal present on the line at the correct time instant is a non-trivial task.

Achieving bit synchronism is only the initial step in achieving the reliable transfer of information over a transmission line. Most of the bit synchronization methods that are used take a variable number of bits before bit synchronism is achieved. Hence in order to interpret the received bitstream on the correct character/byte boundaries, the receiver electronics must also be able to determine the start of each new character/byte and, where appropriate, the start and end of each block of characters/bytes. Again, when the receiver is doing this, it is said to be in **character/byte synchronism** and **block/frame synchronism** respectively. In practice, there are two alternative types of baseband transmission – asynchronous and synchronous – and each uses different methods to achieve the three levels of synchronization. We describe the methods used with asynchronous transmission in Section 1.3.3 and those used with synchronous transmission in Section 1.3.4.

As we showed in Figure 1.9, if the amplitude of the received signal falls below the noise signal level, then the received signal may be incorrectly interpreted and, if it is, a transmission/bit error will result. To allow for this possibility, additional bits are added to each transmitted block to enable the receiver to determine – to high probability – when transmission errors are present in a received block. We describe a selection of the methods that are used to detect the presence of transmission errors in Appendix B.

In some applications, the loss of occasional blocks of information from the received bitstream can be tolerated and hence blocks that are found to contain transmission errors are simply discarded. In other applications, however, the loss of a block is unacceptable and it then becomes necessary for the receiver to request another copy of those blocks that contain errors. This involves the receiver sending what are called error control messages back to the transmitter. This is done in one of two ways and depends on whether the network interface offers a best-effort service or a reliable service. If the service offered is best-effort, both the network and link layers simply discard blocks in error and the error recovery procedure is carried out as part of the transport protocol in the two communicating end systems. If a reliable network service is offered, then the error recovery is part of the link protocol. We discuss link protocols in Section 1.4 and a practical example in Section 1.4.9. We shall defer the discussion of transport protocols until Chapter 7.

1.3.1 Transmission media

The transmission of an electrical signal requires a transmission medium which, normally, takes the form of a transmission line. In some cases, this consists of a pair of conductors or wires. Common alternatives are a beam of light guided by a glass fiber and electromagnetic waves propagating through free space. The type of transmission medium is important, since the various types of media have different bandwidths associated with them which, in turn, determines the maximum bit rate that can be used. We discuss the more common types of transmission media in the following subsections.

Twisted-pair lines

As the name implies, a twisted-pair line is composed of a pair of wires twisted together. The proximity of the signal and ground reference wires means that any interference signal is picked up by both wires, reducing its effect on the difference signal. Furthermore, if multiple twisted pairs are enclosed within the same cable, the twisting of each pair within the cable reduces crosstalk. A schematic of two types of twisted-pair line is shown in Figure 1.10(a).

Twisted-pair lines are suitable, with appropriate line driver and receiver circuits that exploit the potential advantages gained by using such a geometry, for bit rates in order of 1 Mbps over short distances (less than 100 m) and lower bit rates over longer distances. More sophisticated driver and receiver circuits enable bit rates of tens of Mbps to be achieved over similar distances. Such lines, known as **unshielded twisted pairs** (**UTPs**), are used extensively in telephone networks and (with special integrated circuits) in many local area networks. With **shielded twisted pairs** (**STPs**), a protective screen or shield is used to reduce further the effects of interference signals. This was introduced by IBM and is shown in Figure 1.10(b). It is more rigid than UTP and hence is more difficult to install. For this reason, UTP is the most widely used cable in networking applications.

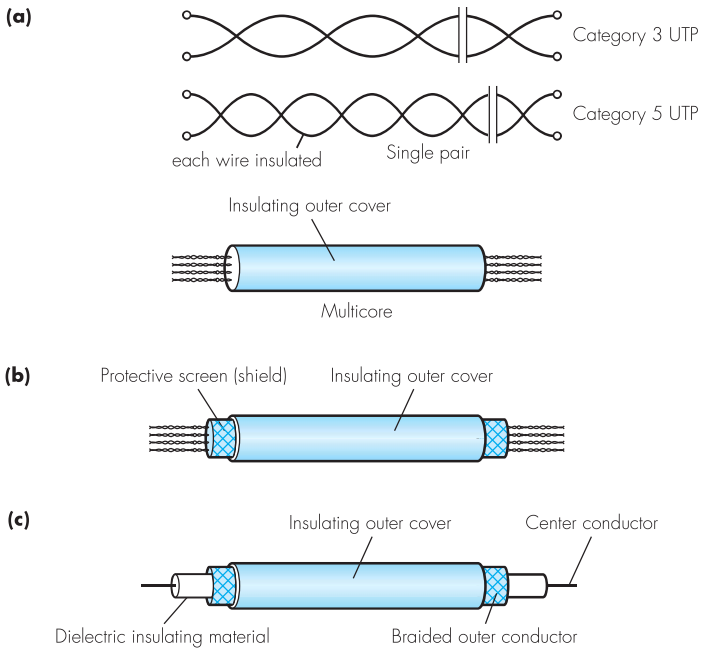


Figure 1.10 Copper wire transmission media: (a) unshielded twisted pair (UTP); (b) shielded twisted pair (STP); (c) coaxial cable.

Coaxial cable

The main limiting factors of a twisted-pair line are its (electrical) **capacity** and a phenomenon known as the **skin effect**. As a result of these, as the bit rate (and hence frequency) of the transmitted signal increases, the current flowing in the wires tends to flow only on the outer surface of the wire, thus using less of the available cross-section. This increases the electrical resistance of the wires for higher-frequency signals, leading to higher attenuation. In addition, at higher frequencies, more signal power is lost as a result of radiation effects. Hence, for applications that demand a high bit rate over long distances, coaxial cable is often used as the transmission medium.

Coaxial cable minimizes both these effects. Figure 1.10(c) shows the signal and ground reference wires as a solid center conductor running concentrically (coaxially) inside a solid (or braided) outer circular conductor. Ideally the space between the two conductors should be filled with air, but in practice it is normally filled with a dielectric insulating material with a solid or honeycomb structure.

The center conductor is effectively shielded from external interference signals by the outer conductor. Also only minimal losses occur as a result of electromagnetic radiation and the skin effect because of the presence of the outer conductor. Coaxial cable can be used with either baseband or modulated transmission, but typically 10 Mbps over several hundred meters – or much higher with modulation – is perfectly feasible. As we shall see in Section 5.2, coaxial cable is used extensively in cable television (CATV) networks.

Optical fiber

While the geometry of coaxial cable significantly reduces the various limiting effects, the maximum signal frequency, and hence the bit rate that can be transmitted using a solid (normally copper) conductor, although very high, is limited. This is also the case for twisted-pair cable. **Optical fiber cable** differs from both these transmission media in that it carries the transmitted bitstream in the form of a fluctuating beam of light in a glass fiber, rather than as an electrical signal on a wire. Light waves have a much wider bandwidth than electrical waves, enabling optical fiber cable to achieve transmission rates of hundreds of Mbps. It is used extensively in the core transmission network of PSTNs, ISDNs and LANs and also CATV networks.

Light waves are also immune to electromagnetic interference and crosstalk. Hence optical fiber cable is extremely useful for the transmission of lower bit rate signals in electrically noisy environments, in steel plants, for example, which employ much high-voltage and current-switching equipments. It is also being used increasingly where security is important, since it is difficult physically to tap.

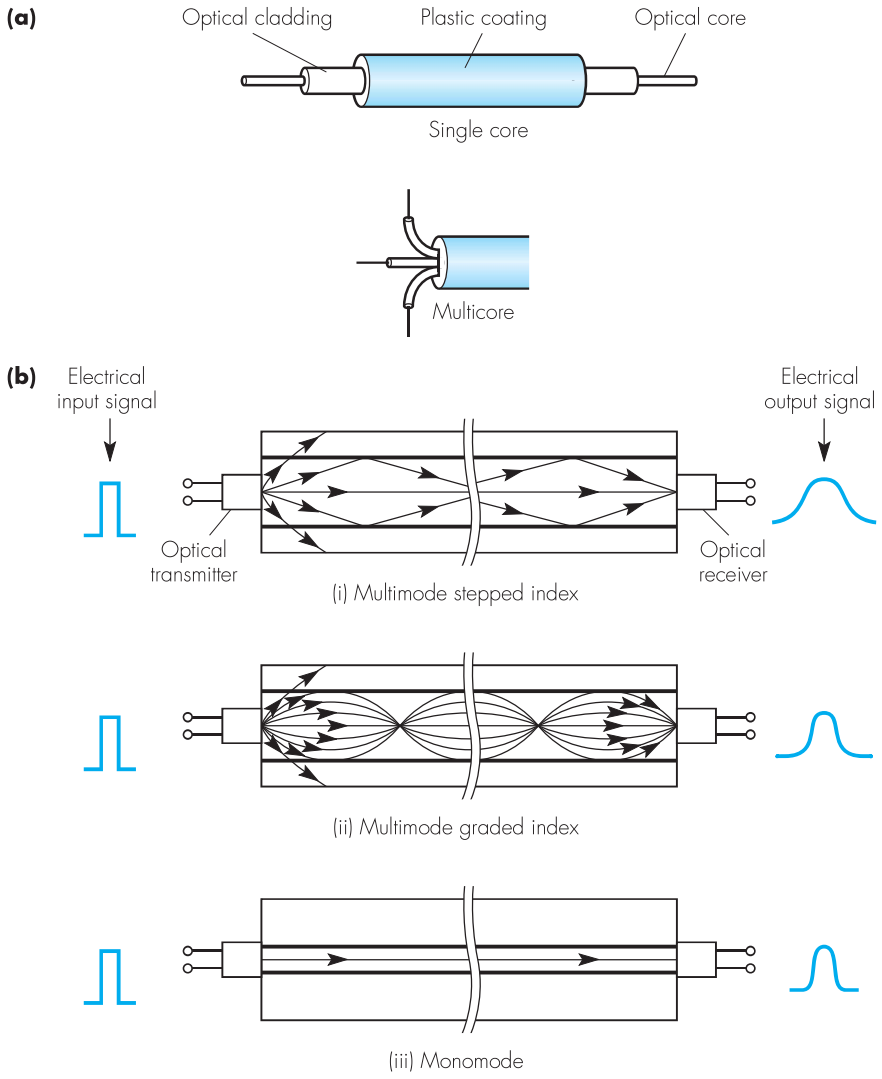


Figure 1.11 Optical fiber transmission media: (a) cable structures; (b) transmission modes.

As we show in Figure 1.11(a) an optical fiber cable consists of a single glass fiber for each signal to be transmitted, contained within the cable's protective coating, which also shields the fiber from any external light sources. The light signal is generated by an optical transmitter, which

performs the conversion from a normal electrical signal as used in a NIC. An optical receiver is used to perform the reverse function at the receiving end. Typically, the transmitter uses a **light-emitting diode** (**LED**) or **laser diode** (**LD**) to perform the conversion operation while the receiver uses a light-sensitive **photodiode** or **photo transistor**.

The fiber itself consists of two parts: an optical core and an optical cladding with a lower refractive index. Light propagates along the optical fiber core in one of three ways depending on the type and width of core material used. These transmission modes are shown in Figure 1.11(b).

In a **multimode stepped index fiber** the cladding and core material each has a different but uniform refractive index. All the light emitted by the diode at an angle less than the critical angle is reflected at the cladding interface and propagates along the core by means of multiple (internal) reflections. Depending on the angle at which it is emitted by the diode, the light will take a variable amount of time to propagate along the cable. Therefore the received signal has a wider pulse width than the input signal with a corresponding decrease in the maximum permissible bit rate. This effect is known as **dispersion** and means this type of cable is used primarily for modest bit rates with relatively inexpensive LEDs compared to laser diodes.

Dispersion can be reduced by using a core material that has a variable (rather than constant) refractive index. As we show in Figure 1.11(b), in a **multi-mode graded index fiber** light is refracted by an increasing amount as it moves away from the core. This has the effect of narrowing the pulse width of the received signal compared with stepped index fiber, allowing a corresponding increase in maximum bit rate.

Further improvements can be obtained by reducing the core diameter to that of a single wavelength (3–10 μm) so that all the emitted light propagates along a single (dispersionless) path. Consequently, the received signal is of a comparable width to the input signal and is called **monomode fiber**. It is normally used with LDs and can operate at hundreds of Mbps.

Alternatively, multiple high bit rate transmission channels can be derived from the same fiber by using different portions of the optical bandwidth for each channel. This mode of operation is known as **wave-division multiplexing** (**WDM**) and, when using this, bit rates in excess of tens of Gbps can be achieved.

Satellites

All the transmission media we have discussed so far have used a physical line to carry the transmitted data. However, data can also be transmitted using electromagnetic (radio) waves through free space as in **satellite** systems. A collimated **microwave beam**, onto which the data is modulated, is transmitted to the satellite from the ground. This beam is received and retransmitted (relayed) to the predetermined destination(s) using an on-board circuit known as a **transponder**. A single satellite has many transponders, each

covering a particular band of frequencies. A typical satellite channel has an extremely high bandwidth (500 MHz) and can provide many hundreds of high bit rate data links using a technique known as **time division multiplexing** (**TDM**). We shall describe this in Section 5.3 but, essentially, the total available capacity of the channel is divided into a number of subchannels, each of which can support a high bit rate link.

Satellites used for communication purposes are normally **geostationary**, which means that the satellite orbits the earth once every 24 hours in synchronism with the earth's rotation and hence appears stationary from the ground. The orbit of the satellite is chosen so that it provides a line-of-sight communication path to the transmitting station(s) and receiving station(s). The degree of the collimation of the microwave beam retransmitted by the satellite can be either coarse, so that the signal can be picked up over a wide geographical area, or finely focused, so that it can be picked up only over a limited area. In the latter case the signal power is higher, allowing smaller-diameter receivers called **antennas** or **dishes** to be used. Satellites are widely used for data transmission applications ranging from interconnecting different national computer communication networks to providing high bit rate paths to link communication networks in different parts of the same country. They are also used in entertainment applications for the broadcast of TV programs.

A typical satellite system used for TV broadcast applications is shown in Figure 1.12(a). Only a unidirectional transmission path is used with the up and down channels operating at different frequencies. For data communication applications, however, a more common configuration involves a central hub ground station that communicates with a number of ground stations distributed around the country. Each ground station has a small antenna associated with it – typically 1 meter in diameter – which, in addition to receiving the signal transmitted by the hub station, allows the station to transmit back to the hub. Such ground stations are known as **very small aperture terminals** or **VSATs**. Typically, as we show in Figure 1.12(b), the computer associated with each VSAT communicates with a central computer connected to the hub. Normally, the central site broadcasts to all VSATs at a high bit rate of 0.5–2 Mbps while in the reverse direction each VSAT transmits at a lower bit rate of up to 64 kbps.

To communicate with a particular VSAT, the central site broadcasts the message with the identity of the intended VSAT at the head of the message. For applications that require VSAT-to-VSAT communication, all messages are first sent to the central site – via the satellite – which then broadcasts them to the intended recipients. With the next generation of higher-powered satellites it will be possible for the routing to be carried out on board the satellite without passing through a central site. Direct VSAT-to-VSAT communication is then possible.

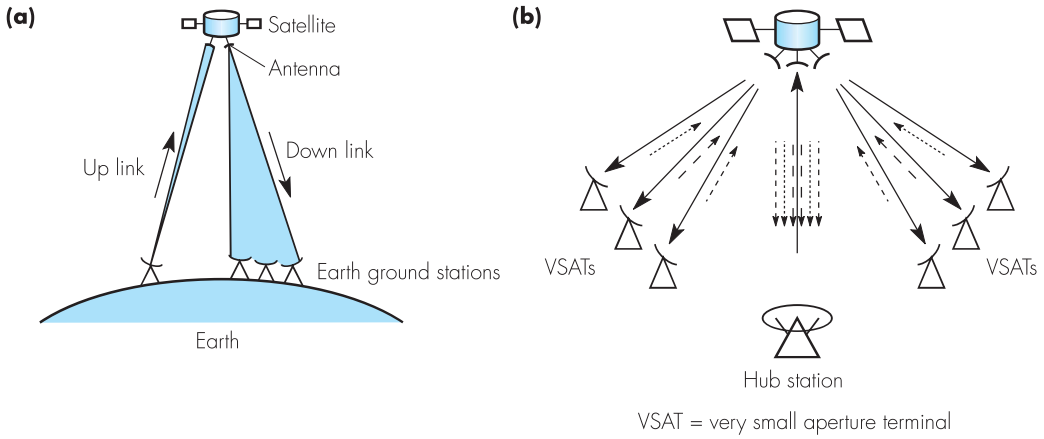


Figure 1.12 Satellite systems: (a) broadcast television; (b) data communications.

Terrestrial microwave

Terrestrial microwave links are widely used to provide communication links when it is impractical or too expensive to install physical transmission media, for example across a river or perhaps a swamp or desert. As the collimated microwave beam travels through the earth's atmosphere, it can be disturbed by such factors as manufactured structures and adverse weather conditions. With a satellite link, on the other hand, the beam travels mainly through free space and is therefore less prone to such effects. Nevertheless, line-of-sight microwave communication through the earth's atmosphere can be used reliably for the transmission of relatively high bit rates over distances in excess of 50 km.

Radio

Radio transmission using lower-frequency radio waves (cf. microwaves) is also used for the transmission of digital information in place of fixed-wire links over large distances.

Modulated transmission is used and example applications include mobile telephony and more general mobile data applications. However, since radio/wireless networks now form an important part of the Internet, a more detailed explanation of radio transmission is given in Appendix D.

Signal propagation delay

As we saw in Section 1.2.4, there is always a short but finite time delay for a signal (electrical, optical, or radio) to propagate (travel) from one end of a transmission medium to the other. This is known as the **propagation delay**, T_p, of the medium. At best, signals propagate (radiate) through the medium at the speed of light ($3 \times 10^8 \text{ m s}^{-1}$). The speed of propagation for twisted-pair wire or coaxial cable is a fraction of this figure. Typically, it is in the region of

2×10^8 m s^{-1}, that is, a signal will take 0.5×10^{-8} s to travel 1 m through the medium. Although this may seem insignificant, in some situations the resulting delay is important.

As we explain later in Section 1.4.1, in many instances, on receipt of each block of data, an acknowledgment of correct (or otherwise) receipt is returned to the sender. An important parameter of a transmission link, therefore, is the **round-trip delay** associated with the link, that is, the time delay between the first bit of a block being transmitted by the sender and the last bit of its associated acknowledgment being received. Clearly, this is a function not only of the time taken to transmit the frame at the link bit rate – known as the transmission delay, T_x – but also of the propagation delay of the link, T_p. The relative weighting of the two times varies for different types of link and hence the two times are often expressed as a ratio a such that:

$$a = \frac{T_p}{T_x}$$

where $$T_p = \frac{\text{physical separation } S \text{ in meters}}{\text{velocity of propagation } V \text{ in meters per second}}$$

and $$T_x = \frac{\text{number of bits to be transmitted } N}{\text{link bit rate } R \text{ in bits per second}}$$

We can conclude from Example 1.4:

- If a is less than 1, then the round-trip delay is determined primarily by the transmission delay.
- If a is equal to 1, then both delays have equal effect.
- If a is greater than 1, then the propagation delay dominates.

Furthermore, in the third case it is interesting to note that, providing blocks are transmitted contiguously, there will be:

$$10 \times 10^6 \times 1.67 \times 10^{-1} = 1.67 \times 10^6 \text{ bits}$$

in transit between the two end systems at any one time, that is, the sending system will have transmitted 1.67×10^6 bits before the first bit arrives at the receiving system. Such links are said, therefore, to have a large **bandwidth/delay product**, where bandwidth relates to the bit rate of the link and delay to the propagation delay of the link. We shall discuss these implications further in Section 1.4.2.

As we have explained, thermal noise is present in a line even when nothing is being transmitted. In addition, there are other noise signals present that are caused by electrical activity external to the transmission line. We identified one source of this, crosstalk, in Section 1.3.1 when we discussed

Example 1.4

A 1000-bit block of data is to be transmitted between two computers. Determine the ratio of the propagation delay to the transmission delay, a, for the following types of data link:

(i) 100 m of twisted-pair wire and a transmission rate of 10 kbps,
(ii) 10 km of coaxial cable and a transmission rate of 1 Mbps,
(iii) 50 000 km of free space (satellite link) and a transmission rate of 10 Mbps.

Assume that the velocity of propagation of an electrical signal within each type of cable is $2\times10^8\,\text{m s}^{-1}$, and that of free space $3\times10^8\,\text{m s}^{-1}$.

Answer:

(i) $$T_p = \frac{S}{V} = \frac{100}{2\times10^8} = 5\times10^{-7}\,\text{s}$$

$$T_x = \frac{N}{R} = \frac{1000}{10\times10^3} = 0.1\,\text{s}$$

$$a = \frac{T_p}{T_x} = \frac{5\times10^{-7}}{0.1} = 5\times10^{-6}$$

(ii) $$T_p = \frac{S}{V} = \frac{10\times10^3}{2\times10^8} = 5\times10^{-5}\,\text{s}$$

$$T_x = \frac{N}{R} = \frac{1000}{1\times10^6} = 1\times10^{-3}\,\text{s}$$

$$a = \frac{T_p}{T_x} = \frac{5\times10^{-5}}{1\times10^{-3}} = 5\times10^{-2}$$

(iii) $$T_p = \frac{S}{V} = \frac{5\times10^{-7}}{3\times10^8} = 1.67\times10^{-1}\,\text{s}$$

$$T_x = \frac{N}{R} = \frac{1000}{10\times10^6} = 1\times10^{-4}\,\text{s}$$

$$a = \frac{T_p}{T_x} = \frac{1.67\times10^{-1}}{1\times10^{-4}} = 1.67\times10^3$$

twisted-pair transmission lines. Crosstalk is caused by unwanted electrical coupling between adjacent lines. This coupling results in a signal that is being transmitted in one line being picked up by adjacent lines as a small but finite (noise) signal.

There are several types of crosstalk but in most cases the most limiting impairment is **near-end crosstalk** or **NEXT**. This is also known as **self-crosstalk** since it is caused by the strong signal output by a transmitter circuit being coupled (and hence interfering) with the much weaker signal at the input to the local receiver circuit. As we showed in Figure 1.9, the received signal is normally significantly attenuated and distorted and hence the amplitude of the coupled signal from the transmit section can be comparable with the amplitude of the received signal.

Special integrated circuits known as **adaptive NEXT cancelers** are now used to overcome this type of impairment. A typical arrangement is shown in Figure 1.13. The canceler circuit adaptively forms an attenuated replica of the crosstalk signal that is coupled into the receive line from the local transmitter and this is subtracted from the received signal. Such circuits are now used in many applications involving UTP cable, for example, to transmit data at high bit rates.

1.3.2 Transmission control schemes

There are two types of transmission control scheme used to transmit the serial bitstream relating to an application over a transmission line. These are called asynchronous transmission and synchronous transmission. We shall describe each separately.

Asynchronous transmission

With asynchronous transmission, each character or byte that makes up a block/message is treated independently for transmission purposes. Hence it can be used both for the transfer of, say, single characters that are entered at a keyboard, and for the transfer of blocks of characters/bytes across a low bit rate transmission line/channel.

The most widely used character set is the **American Standard Code for Information Interchange (ASCII)**. The complete set of characters is shown in tabular form in Figure 1.14. The table includes the binary codewords used to represent each character entered at a keyboard plus some additional characters.

As we can see, each character is represented by a unique 7-bit binary codeword. The use of 7 bits means that there are 128 alternative characters and the codeword used to identify each character is obtained by combining the corresponding column (bits 7–5) and row (bits 4–1) bits together. Bit 7 is the most significant bit and hence the codeword for uppercase M, for example, is 1001101.

In addition to all the normal alphabetic, numeric and punctuation characters – collectively referred to as **printable characters** – the total ASCII character set also includes a number of **control characters**. These include:

- **format control characters**: BS (backspace), LF (line feed), CR (carriage return), SP (space), DEL (delete), ESC (escape) and FF (form feed);

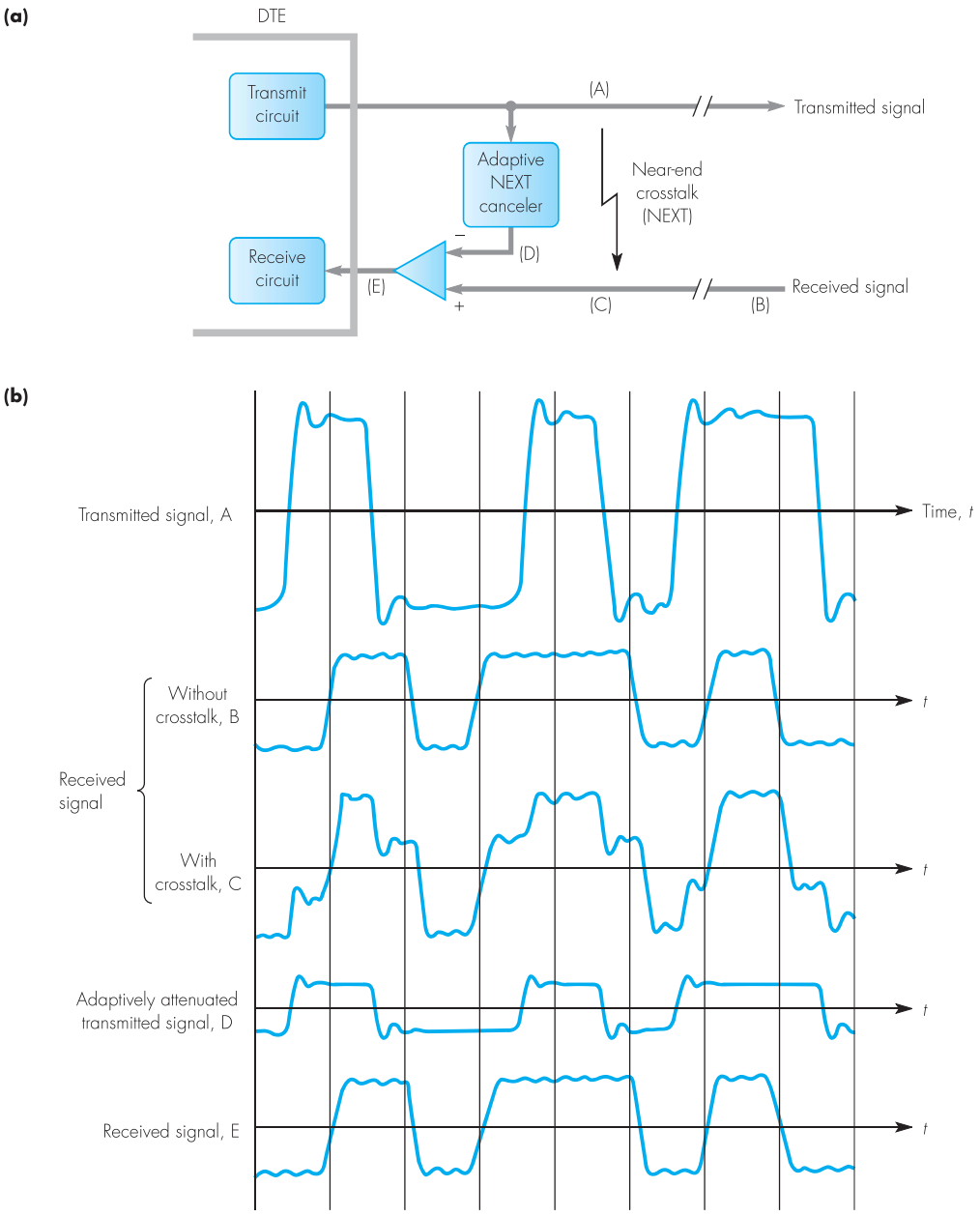


Figure 1.13 Adaptive NEXT cancelers: (a) circuit schematic; (b) example waveforms.

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

Figure 1.14 The ASCII character set.

- **information separators:** FS (file separator) and RS (record separator);
- **transmission control characters**: SOH (start of heading), STX (start of text), ETX (end of text), ACK (acknowledge), NAK (negative acknowledge), SYN (synchronous idle) and DLE (data link escape).

Within end systems – computers, etc. – all data is transferred between the various circuits and subsystems in a word or byte parallel mode. Consequently, since all transfers that are external to the system are carried out bit-serially, the transmission control circuit on the network interface card (NIC) must perform the following functions:

- parallel-to-serial conversion of each character or byte in preparation for its transmission on the line;
- serial-to-parallel conversion of each received character or byte in preparation for its storage and processing in the receiving end system;
- a means for the receiver to achieve bit, character, and frame synchronization;
- the generation of suitable error check digits for error detection and the detection of such errors at the receiver should they occur.

As we show in Figure 1.15(a), parallel-to-serial conversion is performed by a **parallel-in, serial-out (PISO) shift register**. This, as its name implies, allows a complete character or byte to be loaded in parallel and then shifted out bit-serially. Similarly, serial-to-parallel conversion is performed by a **serial-in, parallel-out (SIPO) shift register**, which executes the reverse function.

To achieve bit and character synchronization, we must set the receiving transmission control circuit (which is normally programmable) to operate with the same characteristics as the transmitter in terms of the number of bits per character and the bit rate being used.

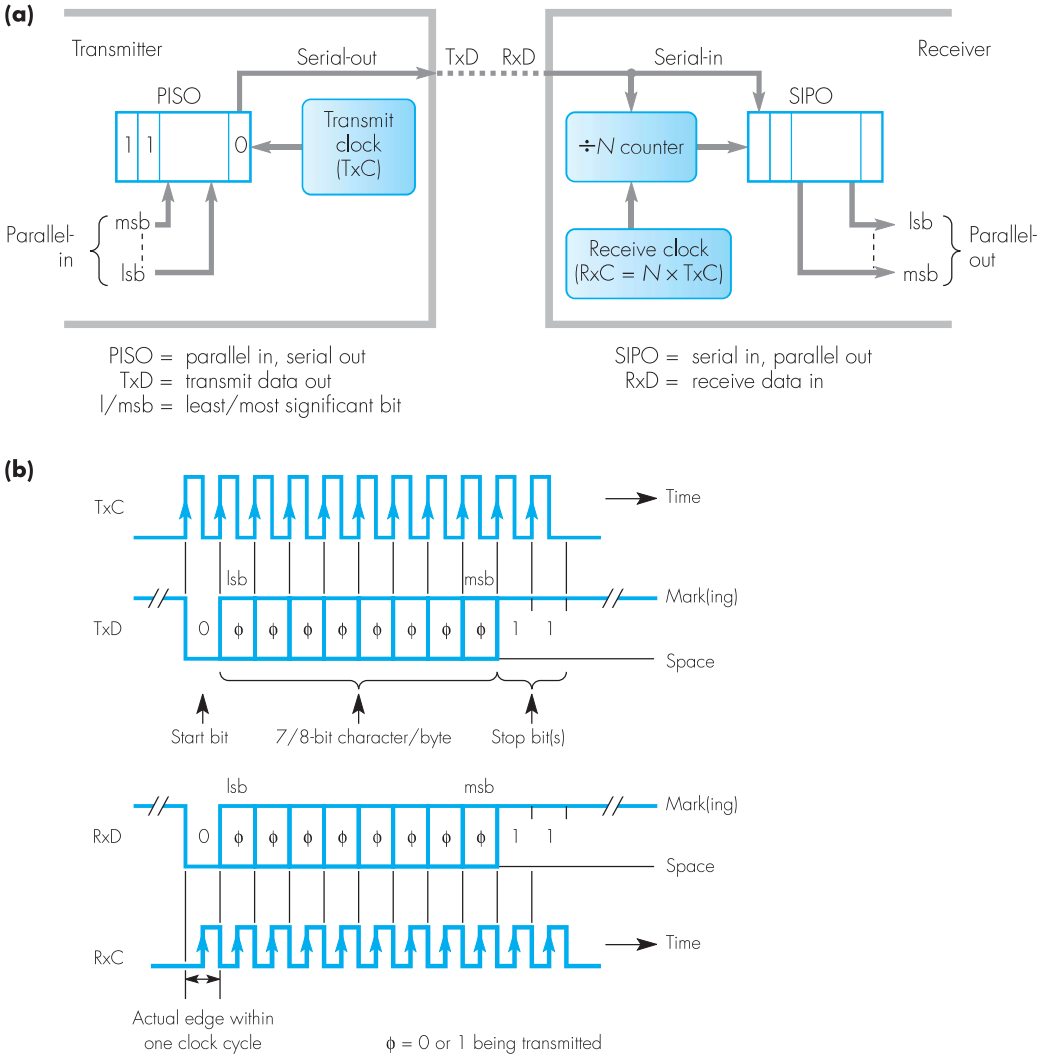


Figure 1.15 Asynchronous transmission: (a) principle of operation; (b) timing principles.

Bit synchronization

In asynchronous transmission, the receiver clock (which is used to sample and shift the incoming signal into the SIPO shift register) runs asynchronously with respect to the incoming signal. In order for the reception process to work reliably, we must devise a scheme whereby the local (asynchronous) receiver clock samples (and hence shifts into the SIPO shift register) the incoming signal as near to the center of the bit cell as possible.

To achieve this, each character/byte to be transmitted is encapsulated between a *start bit* and one or two *stop bits*. As we show in Figure 1.15(b), the start bit is a binary 0 (and known as a *space*) and the stop bit a binary 1 (and known as a *mark*). When transmitting a block comprising a string of characters/bytes, the start bit of each character/byte immediately follows the stop bit(s) of the previous character/byte. When transmitting random characters, however, the line stays at the stop/1 level until the next character is to be transmitted. Hence between blocks (or after each character), the line is said to be *marking* (time).

The use of a start and stop bit per character/byte of different polarities means that, irrespective of the binary contents of each character/byte, there is a guaranteed $1 \rightarrow 0$ transition at the start of each new character/byte. A local receiver clock of N times the transmitted bit rate ($N = 16$ is common) is used and each new bit is shifted into the SIPO shift register after N cycles of this clock. The first $1 \rightarrow 0$ transition associated with the start bit of each character is used to start the counting process. Each bit (including the start bit) is sampled at (approximately) the center of each bit cell. After the first transition is detected, the signal (the start bit) is sampled after $N/2$ clock cycles and then subsequently after N clock cycles for each bit in the character and also the stop bit(s). Three examples of different clock rate ratios are shown in Figure 1.16.

Remembering that the receiver clock (RxC) is running asynchronously with respect to the incoming signal (RxD), the relative positions of the two signals can be anywhere within a single cycle of the receiver clock. Those shown in the figure are just arbitrary positions. Nevertheless, we can deduce from these examples that the higher the clock rate ratio, the nearer the sampling instant will be to the nominal bit cell center. Because of this mode of operation, the maximum bit rate normally used with asynchronous transmission is 19.2 kbps.

Character synchronization

As we indicated above, the receiving transmission control circuit is programmed to operate with the same number of bits per character and the same number of stop bits as the transmitter. After the start bit has been detected and received, the receiver achieves character synchronization simply by counting the programmed number of bits. It then transfers the received character/byte into a local **buffer register** and signals to the controlling device (a microprocessor, for example) on the NIC that a new character/byte has been received. It then awaits the next line signal transition that indicates a new start bit (and hence character) is being received.

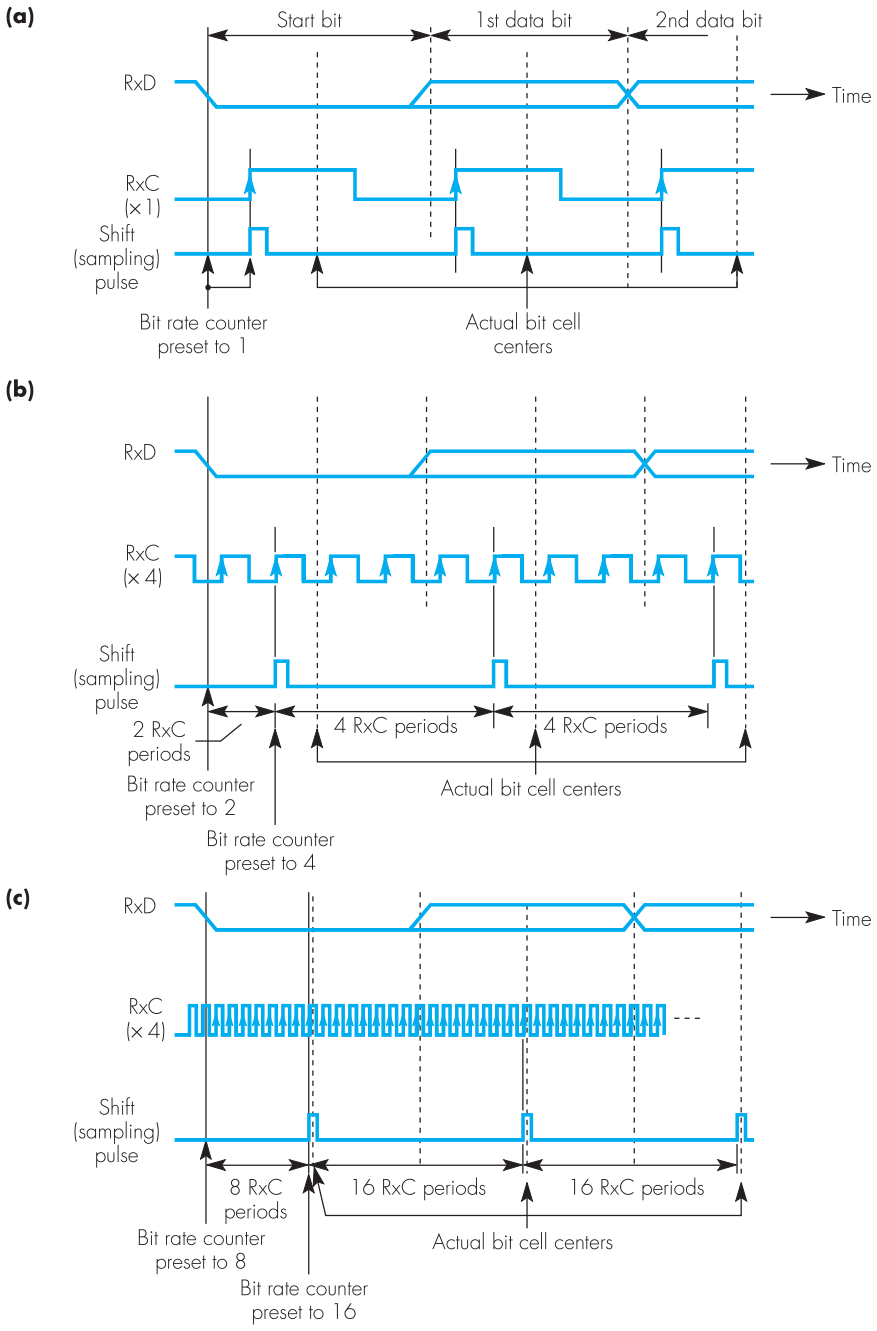


Figure 1.16 Examples of three different receiver clock rate ratios: (a)× 1; (b)× 4; (c)× 16.

Example 1.5

A block of data is to be transmitted across a serial data link. If a clock of 19.2 kHz is available at the receiver, deduce the suitable clock rate ratios and estimate the worst-case deviations from the nominal bit cell centers, expressed as a percentage of a bit period, for each of the following data transmission rates:

(i) 1200 bps

(ii) 2400 bps

(iii) 9600 bps

Answer:

It can readily be deduced from Figure 1.16 that the worst-case deviation from the nominal bit cell centers is approximately plus or minus one half of one cycle of the receiver clock.

Hence:

(i) At 1200 bps, the maximum RxC ratio can be × 16. The maximum deviation is thus ± 3.125%.

(ii) At 2400 bps, the maximum RxC ratio can be × 8. The maximum deviation is thus ± 6.25%.

(iii) At 9600 bps, the maximum RxC ratio can be × 2. The maximum deviation is thus ± 25%.

Clearly, the last case is unacceptable. With a low-quality line, especially one with excessive delay distortion, even the second may be unreliable. It is for this reason that a × 16 clock rate ratio is used whenever possible.

Frame synchronization

When messages comprising blocks of characters or bytes – normally referred to as **information frames** – are being transmitted, in addition to bit and character synchronization, the receiver must be able to determine the start and end of each frame. This is known as frame synchronization.

The simplest method of transmitting blocks of printable characters is to encapsulate the complete block between two special (nonprintable) transmission control characters: STX (start-of-text), which indicates the start of a new frame after an idle period, and ETX (end-of-text), which indicates the end of the frame. As the frame contents consist only of printable characters, the receiver can interpret the receipt of an STX character as signaling the start of a new frame and an ETX character as signaling the end of the frame. This is shown in Figure 1.17(a).

Although the scheme shown is satisfactory for the transmission of blocks of printable characters, when transmitting blocks that comprise strings of bytes (for example, the contents of a file containing compressed speech or video), the use of a single ETX character to indicate the end of a frame is not sufficient. In the case of a string of bytes, one of the bytes might be the same as an ETX character, which would cause the receiver to terminate the reception process abnormally.

To overcome this problem, when transmitting this type of data the two transmission control characters STX and ETX are each preceded by a third transmission control character known as **data link escape** (**DLE**). The modified format of a frame is then as shown in Figure 1.17(b).

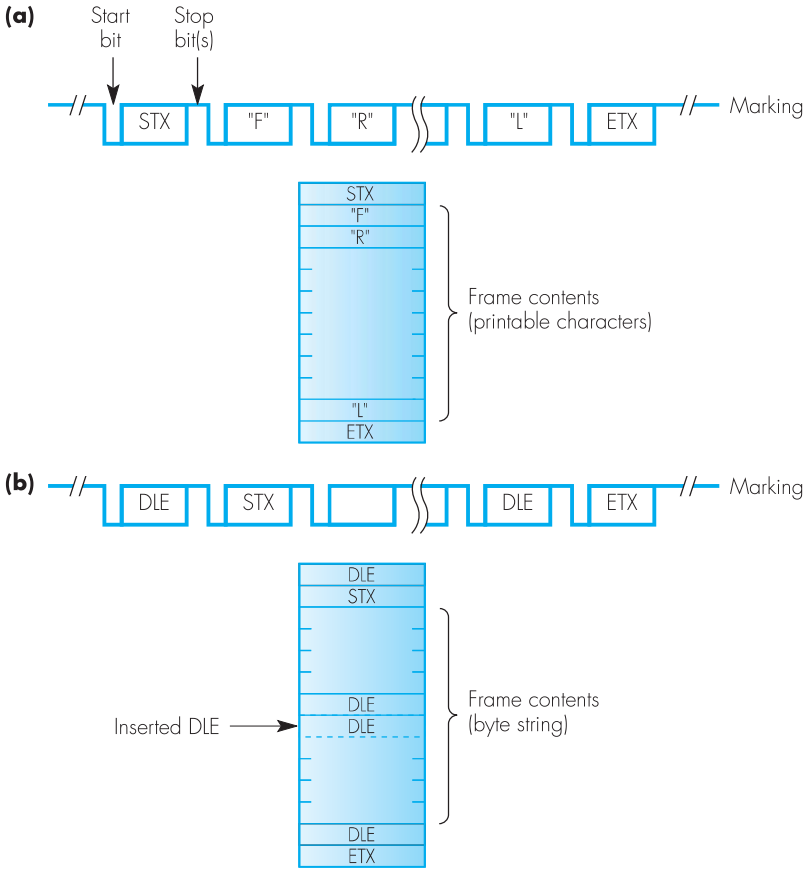


Figure 1.17 Frame synchronization with different frame contents: (a) printable characters; (b) string of bytes.

Remember that the transmitter knows the number of bytes in each frame to be transmitted. After transmitting the start-of-frame sequence (DLE-STX), the transmitter inspects each byte in the frame prior to transmission to determine if it is the same as the DLE character pattern. If it is, irrespective of the next byte, a second DLE character (byte) is transmitted before the next byte. This procedure is repeated until the appropriate number of bytes in the frame have been transmitted. The transmitter then signals the end of the frame by transmitting the unique DLE-STX sequence.

This procedure is known as **character** or **byte stuffing**. On receipt of each byte after the DLE-STX start-of-frame sequence, the receiver determines whether it is a DLE character (byte). If it is, the receiver then processes the next byte to determine whether that is another DLE or an ETX. If it is a DLE, the receiver discards it and awaits the next byte. If it is an ETX, this can reliably be taken as being the end of the frame.

Synchronous transmission

The use of an additional start bit and one or more stop bits per character or byte means that asynchronous transmission is relatively inefficient in its use of transmission capacity, especially when transmitting messages that comprise large blocks of characters or bytes. Also, the bit (clock) synchronization method used with asynchronous transmission becomes less reliable as the bit rate increases. This results, firstly, from the fact that the detection of the first start bit transition is only approximate and, secondly, although the receiver clock operates at N times the nominal transmit clock rate, because of tolerances there are small differences between the two that can cause the sampling instant to drift during the reception of a character or byte. We normally use synchronous transmission to overcome these problems. As with asynchronous transmission, however, we must adopt a suitable method to enable the receiver to achieve bit (clock) character (byte) and frame (block) synchronization. In practice, there are two synchronous transmission control schemes: character-oriented and bit-oriented. We shall discuss each separately but, since they both use the same bit synchronization methods, we shall discuss these methods first.

Bit synchronization

Although we often use the presence of a start bit and stop bit(s) with each character to discriminate between asynchronous and synchronization transmission, the fundamental difference between the two methods is that with asynchronous transmission the receiver clock runs asynchronously (unsynchronized) with respect to the incoming (received) signal, whereas with synchronous transmission the receiver clock operates in synchronism with the received signal.

As we have just indicated, start and stop bits are not used with synchronous transmission. Instead each frame is transmitted as a contiguous stream of binary digits. The receiver then obtains (and maintains) bit synchronization in one of two ways. Either the clock (timing) information is

embedded into the transmitted signal and subsequently extracted by the receiver, or the receiver has a local clock (as with asynchronous transmission) but this time it is kept in synchronism with the received signal by a device known as a **digital phase-lock-loop** (**DPLL**). As we shall see, the DPLL exploits the $1 \rightarrow 0$ or $0 \rightarrow 1$ bit transitions in the received signal to maintain bit (clock) synchronism over an acceptably long period. Hybrid schemes that exploit both methods are also used. The principles of operation of both schemes are shown in Figure 1.18.

Clock encoding and extraction The alternative methods of embedding timing (clock) information into a transmitted bitstream are shown in Figure 1.19. The scheme shown in part (a) is called **Manchester encoding** and that in part (b) **differential Manchester encoding**.

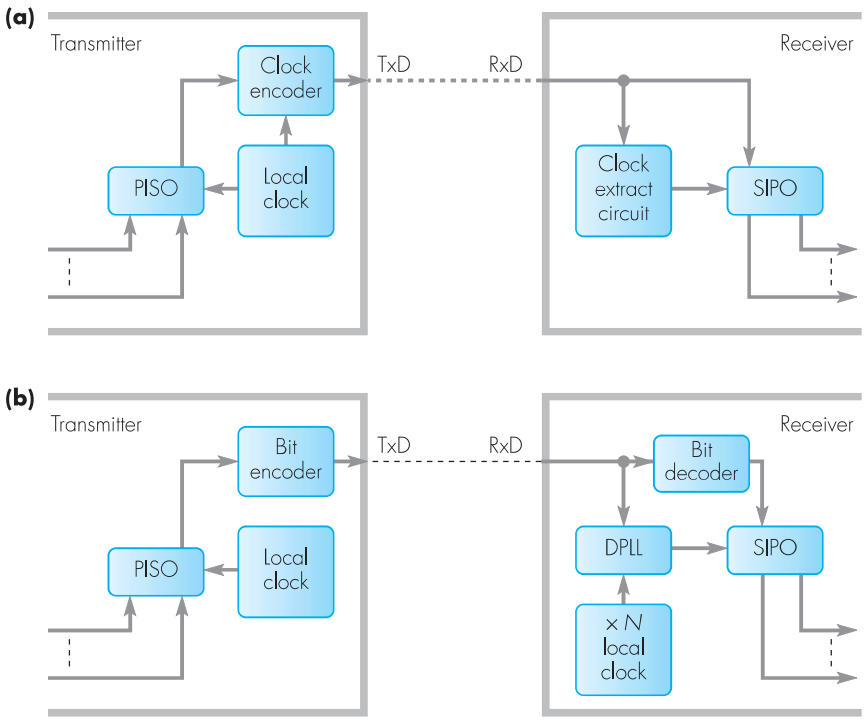


Figure 1.18 Alternative bit/clock synchronization methods with synchronous transmission: (a) clock encoding; (b) digital phase-lock-loop (DPLL).

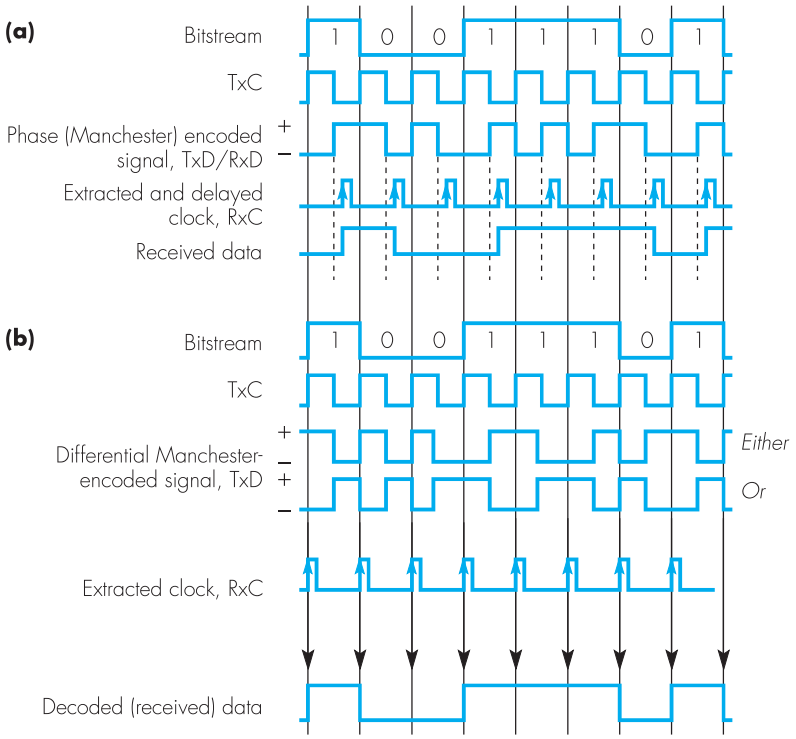


Figure 1.19 Synchronous transmission clock encoding methods: (a) Manchester; (b) differential Manchester.

As we can see, with Manchester encoding each bit is encoded as either a low-high signal (binary 1) or a high-low signal (binary 0), both occupying a single bit-cell period. Also, there is always a transition (high-low or low-high) at the center of each bit cell. It is this that is used by the clock extraction circuit to produce a clock pulse that is then delayed to the center of the second half of the bit cell. At this point the received (encoded) signal is either high (for binary 1) or low (for binary 0) and hence the correct bit is sampled and shifted into the SIPO shift register.

The scheme shown in Figure 1.19(b) is differential Manchester encoding. This differs from Manchester encoding in that although there is still a transition at the center of each bit cell, a transition at the start of the bit cell occurs only if the next bit to be encoded is a 0. This has the effect that the encoded output signal may take on one of two forms depending on the assumed start level (high or low). As we can see, however, one is simply an inverted version of the other and this is the origin of the term "differential". As we show later in Figure 2.12(c), and explain in the accompanying text, a

differential driver circuit produces a pair of differential signals and the differential receiver operates by determining the difference between these two signals. For example, if the two signals each vary between $+V$ and $-V$, then the difference would be $+2V$ and $-2V$. The extracted clock is generated at the start of each bit cell. At this point the received (encoded) signal either changes – for example, from $+2V$ to $-2V$ or from $-2V$ to $+2V$ in which case a binary 0 is shifted into the SIPO – or remains at the same level, in which case a binary 1 is shifted into the SIPO.

The two Manchester encoding schemes are **balanced codes** which means there is no mean (DC) value associated with them. This is so since a string of binary 1s (or 0s) will always have transitions associated with them rather than a constant (DC) level. This is an important feature since it means that the received signal can be **AC coupled** to the receiver electronics using a transformer. The receiver electronics can then operate using its own power supply since this is effectively isolated from the power supply of the transmitter.

Digital phase-lock-loop An alternative approach to encoding the clock in the transmitted bit stream is to utilize a stable clock source at the receiver which is kept in time synchronism with the incoming bit stream. However, as there are no start and stop bits with a synchronous transmission scheme, we must encode the information in such a way that there are always sufficient bit transitions ($1 \rightarrow 0$ or $0 \rightarrow 1$) in the transmitted waveform to enable the receiver clock to be resynchronized at frequent intervals. One approach is to pass the data to be transmitted through a **scrambler**, which randomizes the transmitted bitstream so removing contiguous strings of 1s or 0s. Alternatively, the data may be encoded in such a way that suitable transitions will always be present.

The bit pattern to be transmitted is first differentially encoded as shown in Figure 1.20(a). We refer to the resulting encoded signal as a **non-return-to-zero-inverted** (**NRZI**) waveform. With NRZI encoding the signal level (1 or 0) does not change for the transmission of a binary 1, whereas a binary 0 causes a change. This means that there will always be bit transitions in the incoming signal of an NRZI waveform, providing there are no contiguous streams of binary 1s. On the surface, this may seem no different from the normal NRZ waveform but, as we shall describe later, if a bit-oriented scheme with zero bit insertion is used, an active line will always have a binary 0 in the transmitted bitstream at least every five bit cells. Consequently, the resulting waveform will contain a guaranteed number of transitions, since long strings of 0s cause a transition every bit cell. This enables the receiver to adjust its clock so that it is in synchronism with the incoming bitstream.

The circuit used to maintain bit synchronism is known as a digital phase-lock-loop and is shown in Figure 1.20(b). A crystal-controlled oscillator (clock source), which can hold its frequency sufficiently constant to require only very small adjustments at irregular intervals, is connected to the DPLL. Typically, the frequency of the clock is 32 times the bit rate used on the data

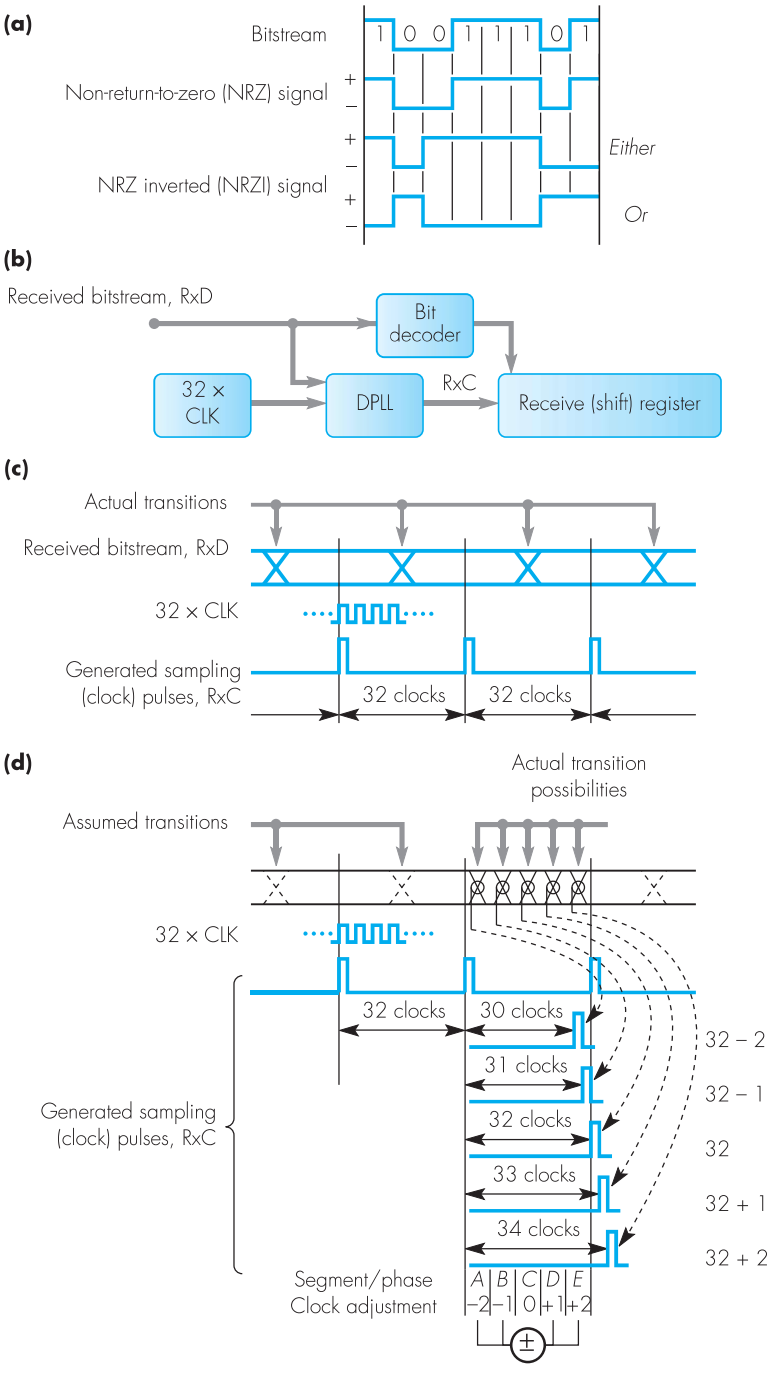


Figure 1.20 DPLL operation: (a) bit encoding; (b) circuit schematic; (c) in phase; (d) clock adjustment rules.

link and is used by the DPLL to derive the timing interval between successive samples of the received bitstream.

Assuming the incoming bitstream and the local clock are in synchronism, the state (1 or 0) of the incoming signal on the line will be sampled (and hence clocked into the SIPO shift register) at the center of each bit cell with exactly 32 clock periods between each sample. This is shown in Figure 1.20(c).

Now assume that the incoming bitstream and local clock drift out of synchronism because of small variations in the latter. The sampling instant is adjusted in discrete increments as shown in Figure 1.20(d). If there are no transitions on the line, the DPLL simply generates a sampling pulse every 32 clock periods after the previous one. Whenever a transition ($1 \rightarrow 0$ or $0 \rightarrow 1$) is detected, the time interval between the previously generated sampling pulse and the next is determined according to the position of the transition relative to where the DPLL thought it should occur. To achieve this, each bit period is divided into five segments, shown as *A*, *B*, *C*, *D*, and *E* in the figure. For example, a transition occurring during segment *A* indicates that the last sampling pulse was too close to the next transition and hence late. The time period to the next pulse is therefore shortened to 30 clock periods. Similarly, a transition occurring in segment *E* indicates that the previous sampling pulse was too early relative to the transition. The time period to the next pulse is therefore lengthened to 34 clock periods. Transitions in segments *B* and *D* are clearly nearer to the assumed transition and hence the relative adjustments are less (−1 and +1 respectively). Finally a transition in segment *C* is deemed to be close enough to the assumed transition to warrant no adjustment.

In this way, successive adjustments keep the generated sampling pulses close to the center of each bit cell. In practice, the widths of each segment (in terms of clock periods) are not equal. The outer segments (*A* and *E*), being further away from the nominal center, are made longer than the three inner segments. For the circuit shown, a typical division might be $A=E=10$, $B=D=5$, and $C=2$. We can readily deduce that in the worst case the DPLL requires 10 bit transitions to converge to the nominal bit center of a waveform: 5 bit periods of coarse adjustments (±2) and 5 bit periods of fine adjustments (±1). Hence when using a DPLL, it is usual before transmitting the first frame on a line, or following an idle period between frames, to transmit a number of characters/bytes to provide a minimum of 10 bit transitions. Two characters/bytes each composed of all 0s, for example, provide 16 transitions with NRZI encoding. This ensures that the DPLL generates sampling pulses at the nominal center of each bit cell by the time the opening character or byte of a frame is received. We must stress, however, that once in synchronism (lock) only minor adjustments normally take place during the reception of a frame.

We can deduce from Figure 1.20(a) that with NRZI encoding the maximum rate at which the encoded signal changes polarity is one half that of Manchester encoding. If the bit period is *T*, with NRZI encoding the maximum rate is $1/T$, whereas with Manchester encoding it is $2/T$. The maximum rate is known as the **modulation rate**. The highest fundamental frequency

component of each scheme is $1/T$ and $2/T$ respectively. This means that, for the same data rate, Manchester encoding requires twice the transmission bandwidth of an NRZI encoded signal, that is, the higher the modulation rate, the wider is the required bandwidth.

The effect of this is that Manchester and differential Manchester encoding are both used extensively in applications such as LANs. As we shall expand upon in Chapter 3, LANs operate in a single office or building and hence use relatively short cable runs. This means that even though they operate at high bit rates – for example 10 Mbps and multiples of this – the attenuation and bandwidth of the transmission medium are not generally a problem. In contrast, as we shall expand upon in Chapter 2, in networks such as an ISDN twisted-pair cable is often used with relatively high bit rates and over distances of several kilometers. Hence encoding schemes such as NRZI are used with each bit represented by a full-width pulse. We shall describe a number of examples of each scheme in later chapters.

Character-oriented

As we indicated at the beginning of the subsection on synchronous transmission, there are two types of synchronous transmission control scheme: character-oriented and bit-oriented. Both use the same bit synchronization methods. The major difference between the two schemes is the method used to achieve character and frame synchronization.

Character-oriented transmission is used primarily for the transmission of blocks of characters, such as files of ASCII characters. Since there are no start or stop bits with synchronous transmission, an alternative way of achieving character synchronization must be used. To achieve this the transmitter adds two or more transmission control characters, known as **synchronous idle** or **SYN** characters, before each block of characters. These control characters have two functions. Firstly, they allow the receiver to obtain (or maintain) bit synchronization. Secondly, once this has been done, they allow the receiver to start to interpret the received bitstream on the correct character boundaries **– character synchronization**. The general scheme is shown in Figure 1.21.

Part (a) shows that frame synchronization (with character-oriented synchronous transmission) is achieved in just the same way as for asynchronous transmission by encapsulating the block of characters – the frame contents – between an STX-ETX pair of transmission control characters. The SYN control characters used to enable the receiver to achieve character synchronization precede the STX start-of-frame character. Once the receiver has obtained bit synchronization it enters what is known as the **hunt mode**. This is shown in Figure 1.21(b).

When the receiver enters the hunt mode, it starts to interpret the received bitstream in a window of eight bits as each new bit is received. In this way, as each bit is received, it checks whether the last eight bits were equal to the known SYN character. If they are not, it receives the next bit and repeats the check. If they are, then this indicates it has found the correct character boundary and hence the following characters are then read after each subsequent eight bits have been received.

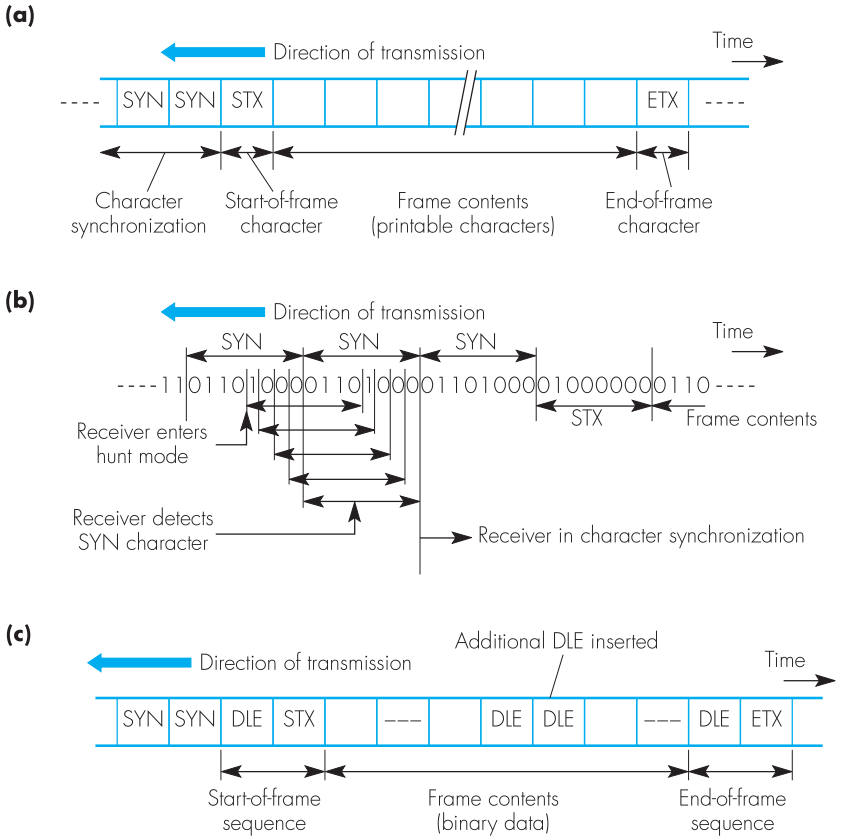


Figure 1.21 Character-oriented synchronous transmission: (a) frame format; (b) character synchronization; (c) data transparency (character stuffing).

Once in character synchronization (and hence reading each character on the correct bit boundary), the receiver starts to process each subsequently received character in search of the STX character indicating the start of the frame. On receipt of the STX character, the receiver proceeds to receive the frame contents and terminates this process when it detects the ETX character. On a point-to-point link, the transmitter normally then reverts to sending SYN characters to allow the receiver to maintain synchronism. Alternatively, the above procedure must be repeated each time a new frame is transmitted.

Finally, as we can see in Figure 1.21(c), when binary data is being transmitted, data transparency is achieved in the same way as described previously by preceding the STX and ETX characters by a DLE (data link escape) character and inserting (stuffing) an additional DLE character

whenever it detects a DLE in the frame contents. In this case, therefore, the SYN characters precede the first DLE character.

Bit-oriented

The need for a pair of characters at the start and end of each frame for frame synchronization, coupled with the additional DLE characters to achieve data transparency, means that a character-oriented transmission control scheme is relatively inefficient for the transmission of binary data. Moreover, the format of the transmission control characters varies for different character sets, so the scheme can be used only with a single type of character set, even though the frame contents may be pure binary data. To overcome these problems, a more universal scheme known as **bit-oriented transmission** is now the preferred control scheme as it can be used for the transmission of frames comprising either printable characters or binary data. The main features of the scheme are shown in Figure 1.22(a). It differs mainly in the way the start and end of each frame are signaled.

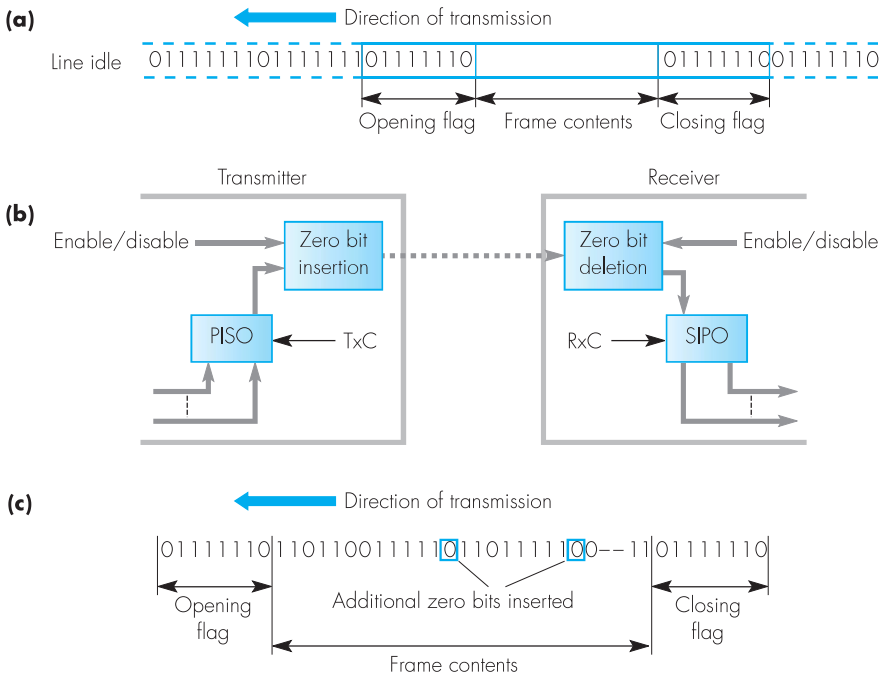


Figure 1.22 Bit-oriented synchronous transmission: (a) framing structure; (b) zero bit insertion circuit location; (c) example transmitted frame contents.

The start and end of a frame are both signaled by the same unique 8-bit pattern 01111110, known as the **flag byte** or **flag pattern**. We use the term "bit-oriented" because the received bitstream is searched by the receiver on a bit-by-bit basis both for the start-of-frame flag and, during reception of the frame contents, for the end-of-frame flag. Thus in principle the frame contents need not necessarily comprise multiples of eight bits.

To enable the receiver to obtain and maintain bit synchronism, the transmitter sends a string of **idle bytes** (each comprising 01111111) preceding the start-of-frame flag. Recall that with NRZI encoding the 0 in the idle byte enables the DPLL at the receiver to obtain and maintain clock synchronization. On receipt of the opening flag, the received frame contents are read and interpreted on 8-bit (byte) boundaries until the closing flag is detected. The reception process is then terminated.

To achieve data transparency with this scheme, we must ensure that the flag pattern is not present in the frame contents. We do this by using a technique known as **zero bit insertion** or **bit stuffing**. The circuit that performs this function is located at the output of the PISO register, as shown in Figure 1.22(b). It is enabled by the transmitter only during transmission of the frame contents. When enabled, the circuit detects whenever it has transmitted a sequence of five contiguous binary 1 digits, then automatically inserts an additional binary 0 digit. In this way, the flag pattern 01111110 can never be present in the frame contents between the opening and closing flags.

A similar circuit at the receiver located prior to the input of the SIPO shift receiver performs the reverse function. Whenever a zero is detected after five contiguous 1 digits, the circuit automatically removes (deletes) it from the frame contents. Normally the frame also contains additional error detection digits preceding the closing flag which are subjected to the same bit stuffing operation as the frame contents. An example stuffed bit pattern is shown in Figure 1.22(c).

1.4 Protocol basics

In Appendix B we describe the different methods that are used to detect the presence of transmission errors. Also, as we explained in Section 1.3, in most cases any blocks/frames that are received containing errors are simply discarded by the link layer. It is then left to the transport layer in each of the two communicating end systems to detect any missing blocks and, if necessary, to request that another copy of these is retransmitted. However, in a small number of cases the error recovery procedure is performed in the link layer; for example, in the link layer associated with a line leased from a PSTN. In this section we describe the basic principles associated with the error control procedure that is used and we then use this to explain how a protocol is specified.

1.4.1 Error control

The transmission control circuit associated with most NICs performs both the transmission control and error detection functions. The link layer protocol then builds on these basic functions to provide the required link layer service. In some networks – for example LANs and the Internet – frames received with errors are simply discarded. In others – for example wireless LANs – the receiving link protocol checks the received frame for possible transmission errors and then returns a short control message/frame either to acknowledge its correct receipt or to request that another copy of the frame is sent. This type of error control is known as **automatic repeat request** (**ARQ**) and the aim of all ARQ schemes is to provide a *reliable* link layer service. In this context, "reliable" means that the two peer link layer protocols will communicate with each other in order to deliver a sequence of blocks that is submitted to the sending link layer protocol. Thus

- the blocks will be delivered by the receiving link layer protocol in the same sequence as they were submitted and with no duplicate copies of any of the blocks;
- to a high probability, each block will be free of any bit errors.

The most basic type of ARQ scheme is **idle RQ** and so we shall start by explaining this. We then identify the limitations of idle RQ and explain how these are overcome in the **continuous RQ** scheme. In practice, there are two types of continuous RQ: selective repeat and go-back-N, both of which we describe. We then return to the idle RQ scheme to explain how the error control part of a protocol is specified.

1.4.2 Idle RQ

In order to discriminate between the sender (source) and receiver (destination) of data frames – more generally referred to as information or **I-frames** – the terms **primary** (**P**) and **secondary** (**S**) are used respectively. The idle RQ protocol operates in a half-duplex mode since the primary, after sending an I-frame, waits until it receives a response from the secondary as to whether the frame was correctly received or not. The primary then either sends the next frame, if the previous frame was correctly received, or retransmits a copy of the previous frame if it was not.

The secondary informs the primary of a correctly received frame by returning a (positive) **acknowledgment** or **ACK-frame**. Similarly, if the secondary receives an I-frame containing errors, then it returns a **negative acknowledgment** or **NAK-frame**. Three example frame sequences illustrating various aspects of this basic procedure are shown in Figure 1.23. The following points should be noted when interpreting the sequences:

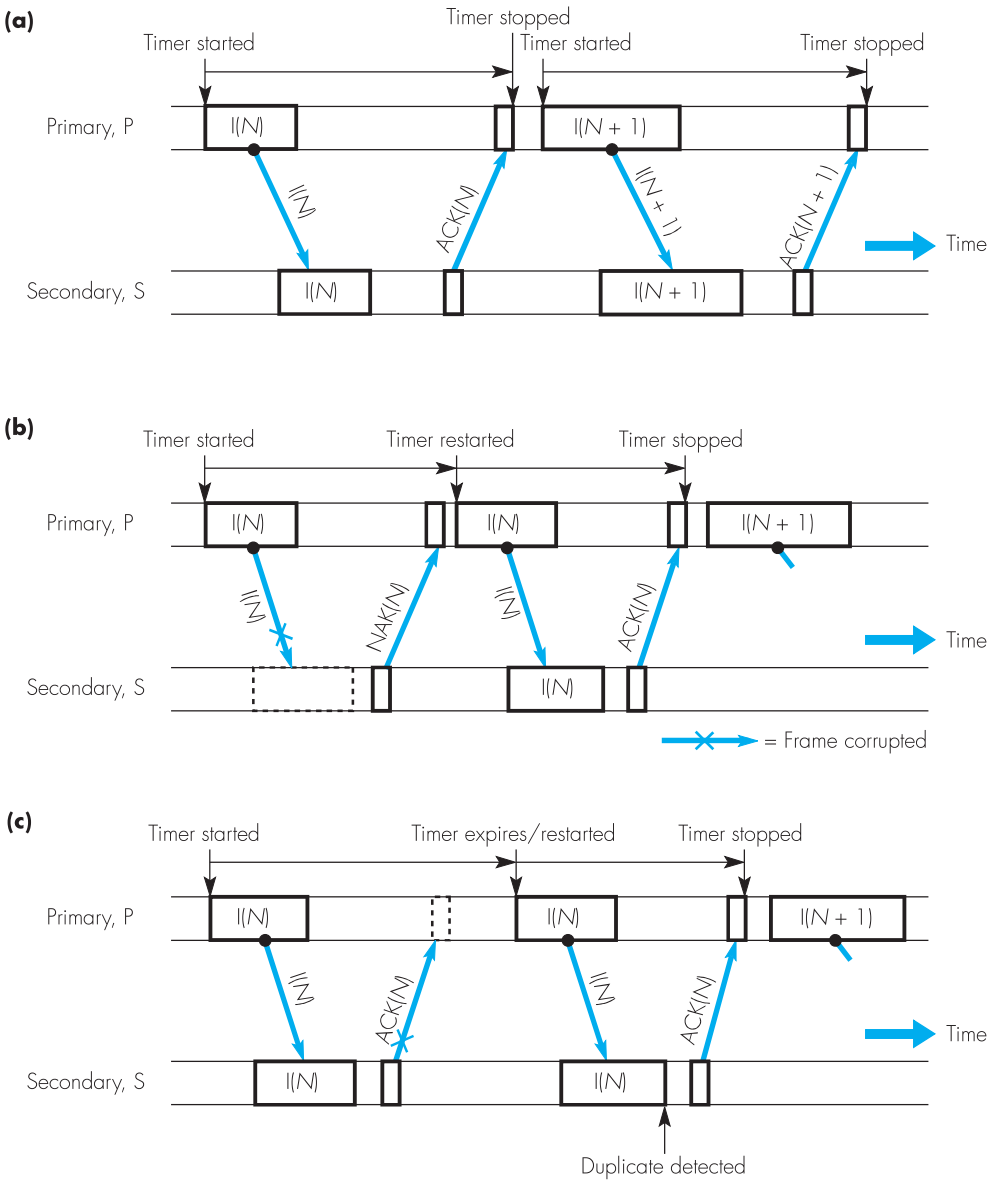


Figure 1.23 ARQ error control scheme: (a) error free; (b) corrupted I-frame; (c) corrupted ACK-frame.

- P can have only one I-frame outstanding (awaiting an ACK/NAK-frame) at a time.
- When P initiates the transmission of an I-frame it starts a timer.
- On receipt of an error-free I-frame, S returns an ACK-frame to P and, on receipt of this, P stops the timer for this frame and proceeds to send the next frame – part (a).
- On receipt of an I-frame containing transmission errors, S discards the frame and returns a NAK-frame to P which then sends another copy of the frame and restarts the timer – part (b).
- If P does not receive an ACK- (or NAK-) frame within the timeout interval, P retransmits the I-frame currently waiting acknowledgment – part (c). However, since in this example it is an ACK-frame that is corrupted, S detects that the next frame it receives is a duplicate copy of the previous error-free frame it received rather than a new frame. Hence S discards the duplicate and, to enable P to resynchronize, returns a second ACK-frame for it. This procedure repeats until either an error-free copy of the frame is received or a defined number of retries is reached in which case the network layer in P would be informed of this.

As we show in the figure, in order for S to determine when a duplicate is received, each frame transmitted by P contains a unique identifier known as the **send sequence number N(S)** (N, $N+1$, and so on) within it. Also, S retains a record of the sequence number contained within the last I-frame it received without errors and, if the two are the same, this indicates a duplicate. The sequence number in each ACK- and NAK-frame is known as the **receive sequence number N(R)** and, since P must wait for an ACK- or NAK-frame after sending each I-frame, the scheme is known also as **send-and-wait** or sometimes **stop-and-wait**. Alternatively, since only a single frame is being transferred at a time, only two sequence numbers are required. Normally, just a single bit is used and this alternates between 0 and 1. Hence the scheme is known also as the **alternating bit protocol** and an example application is in the Bluetooth wireless network that we describe later in Section 4.2 of Chapter 4.

Link utilization

Before considering the error procedures associated with the two types of continuous RQ scheme, we shall first quantify the efficiency of utilization of the available link capacity with the idle RQ scheme. The efficiency of utilization U is a ratio of two times, each measured from the point in time when the transmitter starts to send a frame. It is defined as:

$$U = \frac{T_{\text{ix}}}{T_{\text{t}}}$$

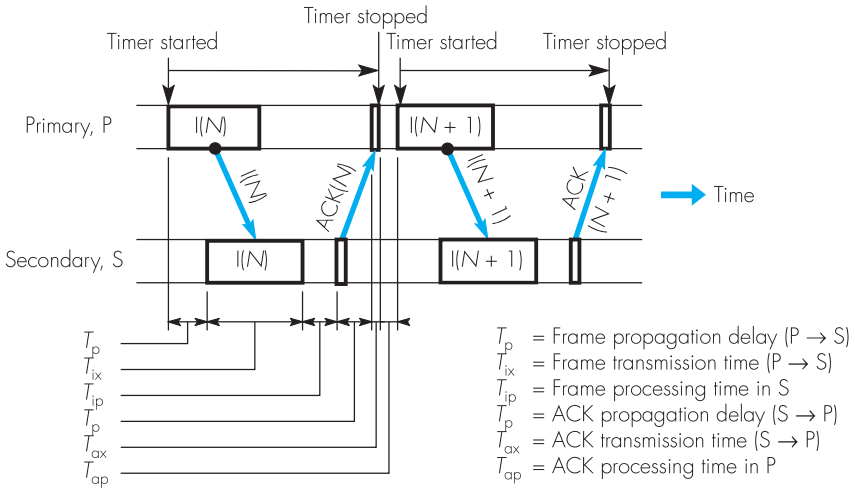


Figure 1.24 Idle RQ link utilization.

where T_{ix} is the time for the transmitter to transmit a frame and T_t equals T_{ix} plus any time the transmitter spends waiting for an acknowledgment.

To quantify the link utilization with idle RQ, a frame sequence diagram with the various component times identified is given in Figure 1.24. In practice, in most cases for which the idle RQ protocol is adequate, the time to process an I-frame T_{ip} and its associated ACK-frame T_{ap} are both short compared with their transmission times T_{ix} and T_{ax}. Also, since an ACK-frame is much shorter than an I-frame, T_{ax} is negligible compared with T_{ix}. Hence the minimum total time before the next frame can be transmitted is often approximated to $T_{ix} + 2T_p$ where T_p is the signal propagation delay of the link. An approximate expression for U is thus:

$$U = \frac{T_{ix}}{T_{ix} + 2T_p}$$

or:

$$U = \frac{1}{1 + 2T_p/T_{ix}}$$

As we described earlier in the subsection on Signal propagation delay, the ratio T_p/T_{ix} is often given the symbol a and hence:

$$U = \frac{1}{1 + 2a}$$

In Example 1.4 we saw that a can range from a small fraction for low bit rate links of modest length to a large integer value for long links and high bit rates. For these two extremes, U varies between a small fraction and near unity (100%).

Example 1.6

A series of 1000-bit frames is to be transmitted using an idle RQ protocol. Determine the link utilization for the following types of data link assuming a transmission bit rate of (a) 1 kbps and (b) 1 Mbps. Assume that the velocity of propagation of the first two links is 2×10^8 m s^{-1} and that of the third link 3×10^8 m s^{-1}. Also the bit error rate is negligible.

(i) A twisted-pair cable 1 km in length

(ii) A leased line 200 km in length

(iii) A satellite link of 50 000 km.

Answer:

The time taken to transmit a frame T_{ix} is given by:

$$T_{ix} = \frac{\text{Number of bits in frame, } N}{\text{Bit rate, } R, \text{ in bps}}$$

At 1 kbps:

$$T_{ix} = \frac{1000}{10^3} = 1\text{ s}$$

At 1 Mbps:

$$T_{ix} = \frac{1000}{10^6} = 10^{-3}\text{ s}$$

$$T_p = \frac{S}{V} \quad \text{and} \quad U = \frac{1}{1+2a}$$

(i) $T_p = \dfrac{10^3}{2\times10^8} = 5\times10^{-6}$ s

(a) $a = \dfrac{5\times10^{-6}}{1} = 5\times10^{-6}$ and hence $(1+2a) \cong 1$ and $U = 1$

(b) $a = \dfrac{5\times10^{-6}}{10^{-3}} = 5\times10^{-3}$ and hence $(1+2a) \cong 1$ and $U = 1$

(ii) $T_p = \dfrac{200\times10^3}{2\times10^8} = 1\times10^{-3}$ s

(a) $a = \dfrac{1\times10^{-3}}{1} = 1\times10^{-3}$ and hence $(1+2a) \cong 1$ and $U = 1$

(b) $a = \dfrac{1\times10^{-3}}{10^{-3}} = 1$ and hence $(1+2a) > 1$ and $U = \dfrac{1}{1+2} = 0.33$

(iii) $T_p = \dfrac{50\times10^6}{3\times10^8} = 0.167\,\text{s}$

(a) $a = \dfrac{0.167}{1} = 0.167$ and hence $(1+2a) > 1$ and $U = \dfrac{1}{1+0.334} = 0.75$

(b) $a = \dfrac{0.167}{10^{-3}} = 167$ and hence $(1+2a) > 1$ and $U = \dfrac{1}{1+334} = 0.003$

The results are summarized in Figure 1.25 from which we can make some interesting observations. Firstly, for relatively short links for which a is less than 1, the link utilization is (to a good approximation) 100% and is independent of the bit rate. This means that an idle RQ protocol is perfectly adequate for short links and modest bit rates. Examples are networks based on modems and an analog PSTN. Secondly, for longer terrestrial links, the link utilization is high for low bit rates (and hence low values of a) but falls off significantly as the bit rate (and hence a) increases. Thirdly, the link utilization is poor for satellite links, even at low bit rates. We can conclude that an idle RQ protocol is unsuitable for such applications and also for those that involve high bit rate terrestrial links which include all of the networks that are used for multimedia.

1.4.3 Continuous RQ

With a continuous RQ error control scheme, link utilization is much improved at the expense of increased buffer storage requirements. As we shall see, a duplex link is required for its implementation. An example illustrating the transmission of a sequence of I-frames and their returned ACK-frames is shown in Figure 1.26. You should note the following points when interpreting the operation of the scheme:

- P sends I-frames continuously without waiting for an ACK-frame to be returned.
- Since more than one I-frame is awaiting acknowledgment, P retains a copy of each I-frame transmitted in a **retransmission list** that operates on a FIFO queue discipline.
- S returns an ACK-frame for each correctly received I-frame.
- Each I-frame contains a unique identifier which is returned in the corresponding ACK-frame.
- On receipt of an ACK-frame, the corresponding I-frame is removed from the retransmission list by P.

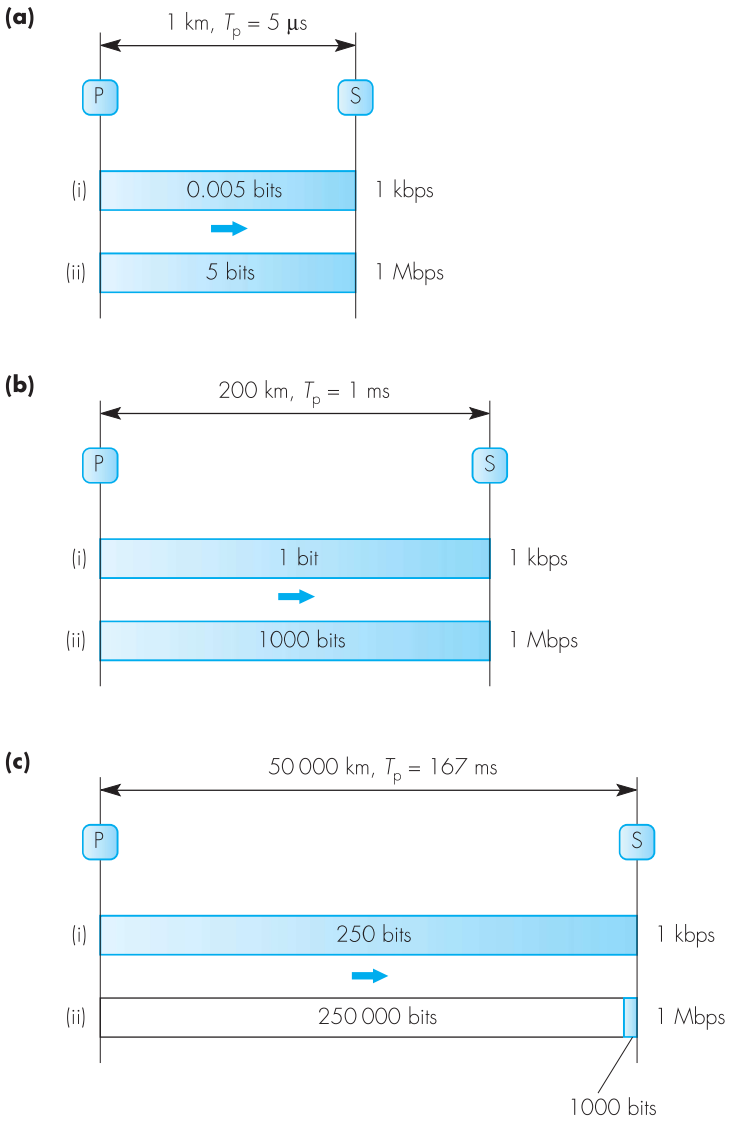


Figure 1.25 Effect of propagation delay as a function of data transmission rate; parts correspond to Example 1.6.

- Frames received free of errors are placed in the **link receive** list to await processing.
- On receipt of the next in-sequence I-frame expected, S delivers the information content within the frame to the upper network layer immediately it has processed the frame.

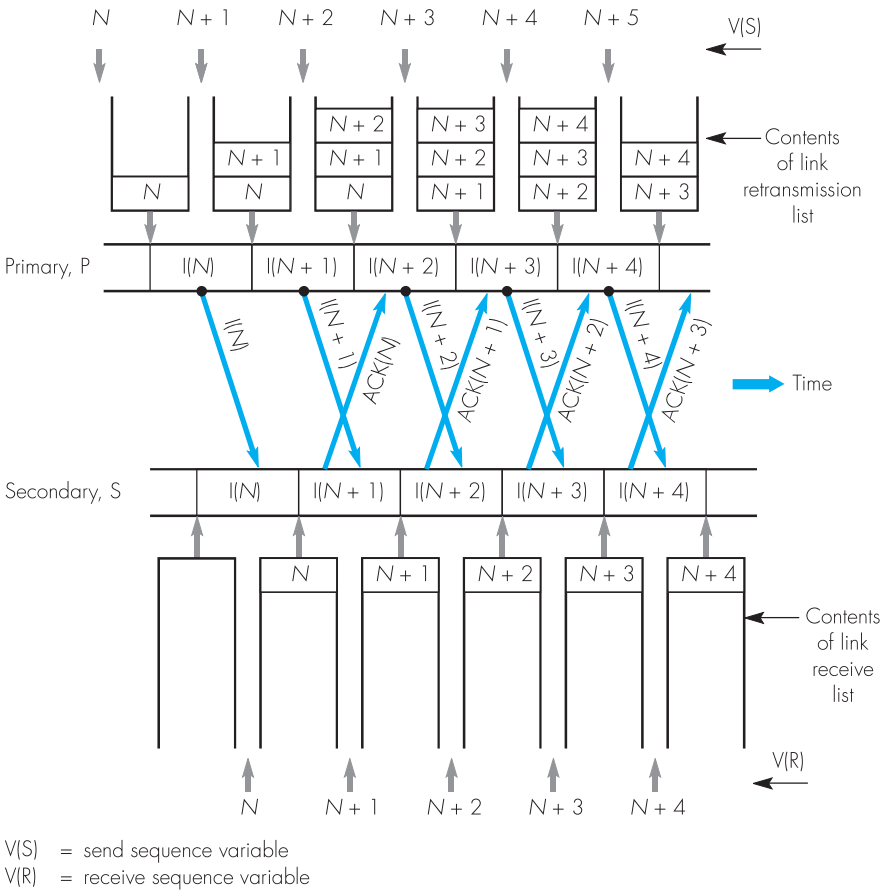


Figure 1.26 Continuous RQ frame sequence without transmission errors.

To implement the scheme, P must retain a send sequence variable V(S), which indicates the send sequence number N(S) to be allocated to the next I-frame to be transmitted. Also, S must maintain a receive sequence variable V(R), which indicates the next in-sequence I-frame it is waiting for.

The frame sequence shown in Figure 1.26 assumes that no transmission errors occur. When an error does occur, one of two retransmission strategies may be followed:

- S detects and requests the retransmission of just those frames in the sequence that are corrupted – selective repeat.
- S detects the receipt of an out-of-sequence I-frame and requests P to retransmit all outstanding unacknowledged I-frames from the last correctly received, and hence acknowledged, I-frame – go-back-N.

Note that with both continuous RQ schemes, corrupted frames are discarded and retransmission requests are triggered only after the next error-free frame is received. Hence, as with the idle RQ scheme, a timeout is applied to each frame transmitted to overcome the possibility of a corrupted frame being the last in a sequence of new frames.

Selective repeat

Two example frame sequence diagrams that illustrate the operation of the selective repeat retransmission control scheme are shown in Figure 1.27. The sequence shown in part (a) shows the effect of a corrupted I-frame being received by S. The following points should be noted when interpreting the sequence:

- An ACK-frame acknowledges all frames in the retransmission list up to and including the I-frame with the sequence number the ACK contains.
- Assume I-frame $N+1$ is corrupted.
- S returns an ACK-frame for I-frame N.
- When S receives I-frame $N+2$ it detects I-frame $N+1$ is missing from V(R) and hence returns a NAK-frame containing the identifier of the missing I-frame $N+1$.
- On receipt of NAK $N+1$, P interprets this as S is still awaiting I-frame $N+1$ and hence retransmits it.
- When P retransmits I-frame $N+1$ it enters the **retransmission state**.
- When P is in the retransmission state, it suspends sending any new frames and sets a timeout for the receipt of ACK $N+1$.
- If the timeout expires, another copy of I-frame $(N+1)$ is sent.
- On receipt of ACK $N+1$ P leaves the retransmission state and resumes sending new frames.
- When S returns a NAK-frame it enters the retransmission state.
- When S is in the retransmission state, the return of ACK-frames is suspended.
- On receipt of I-frame $N+1$, S leaves the retransmission state and resumes returning ACK-frames.
- ACK $N+1$ acknowledges all frames up to and including frame $N+4$.
- A timer is used with each NAK-frame to ensure that if it is corrupted (and hence NAK $N+1$ is not received), it is retransmitted.

The sequence shown in Figure 1.27(b) shows the effect of a corrupted ACK-frame. The following points should be noted:

- Assume ACK N is corrupted.
- On receipt of ACK-frame $N+1$, P detects that I-frame N is still awaiting acknowledgment and hence retransmits it.

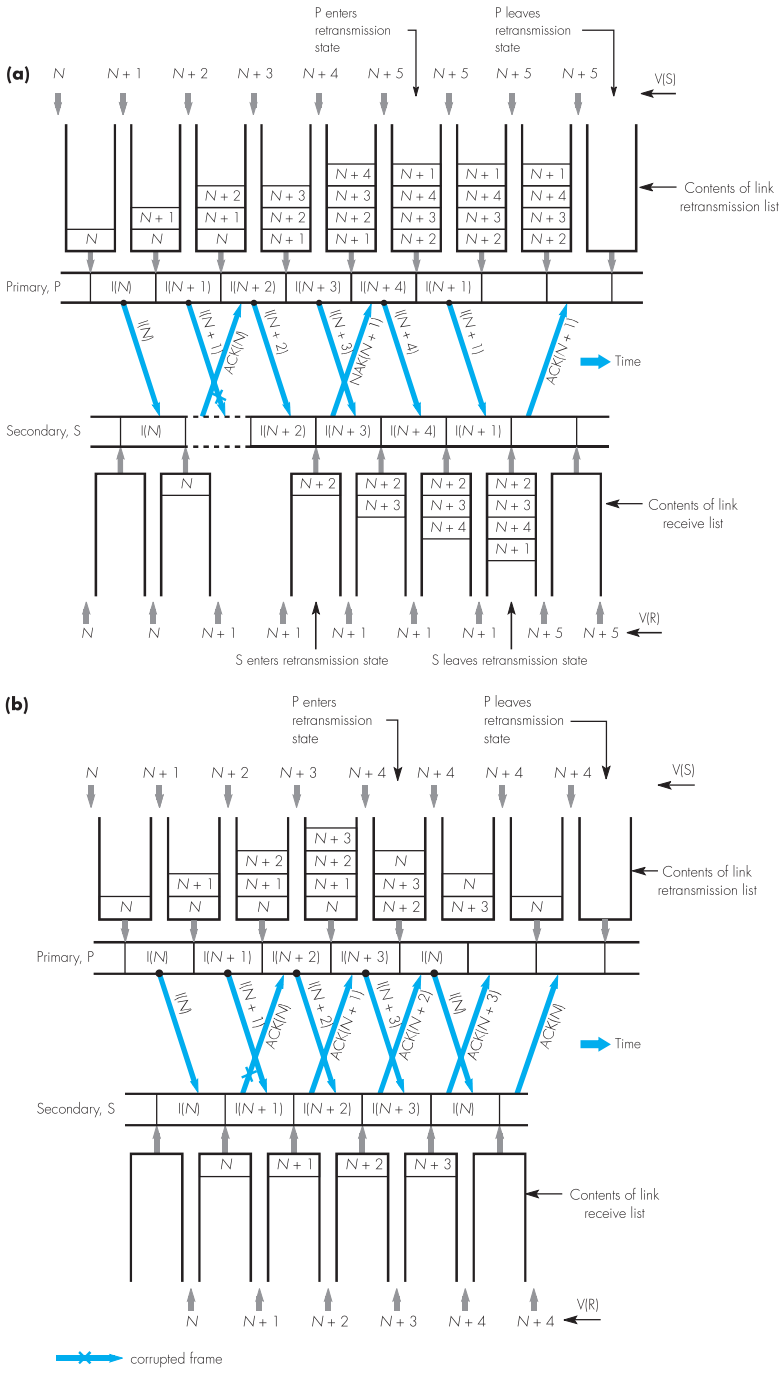


Figure 1.27 Selective repeat: (a) effect of corrupted I-frame; (b) effect of corrupted ACK-frame.

- On receipt of the retransmitted I-frame *N*, S determines from its received sequence variable that this has already been received correctly and is therefore a duplicate.
- S discards the frame but returns an ACK-frame to ensure P removes the frame from the retransmission list.

Go-back-N

Two example frame sequence diagrams that illustrate the operation of the go-back-N retransmission control scheme are shown in Figure 1.28. The sequence shown in part (a) shows the effect of a corrupted I-frame being received by S. The following points should be noted:

- Assume I-frame $N+1$ is corrupted.
- S receives I-frame $N+2$ out of sequence.
- On receipt of I-frame $N+2$, S returns NAK $N+1$ informing P to go back and start to retransmit from I-frame $N+1$.
- On receipt of NAK $N+1$, P enters the retransmission state. When in this state, it suspends sending new frames and commences to retransmit the frames waiting acknowledgment in the retransmission list.
- S discards frames until it receives I-frame $N+1$.
- On receipt of I-frame $N+1$, S resumes accepting frames and returning acknowledgments.
- A timeout is applied to NAK-frames by S and a second NAK is returned if the correct in-sequence I-frame is not received in the timeout interval.

The frame sequence shown in Figure 1.28(b) shows the effect of a corrupted ACK-frame. Note that:

- S receives each transmitted I-frame correctly.
- Assume ACK-frames N and $N+1$ are both corrupted.
- On receipt of ACK-frame $N+2$, P detects that there are two outstanding I-frames in the retransmission list (N and $N+1$).
- Since it is an ACK-frame rather than a NAK-frame, P assumes that the two ACK-frames for I-frames N and $N+1$ have both been corrupted and hence accepts ACK-frame $N+2$ as an acknowledgment for the outstanding frames.

In order to discriminate between the NAK-frames used in the two schemes, in the selective repeat scheme a NAK is known as a **selective reject** and in the go-back-N scheme a **reject**.

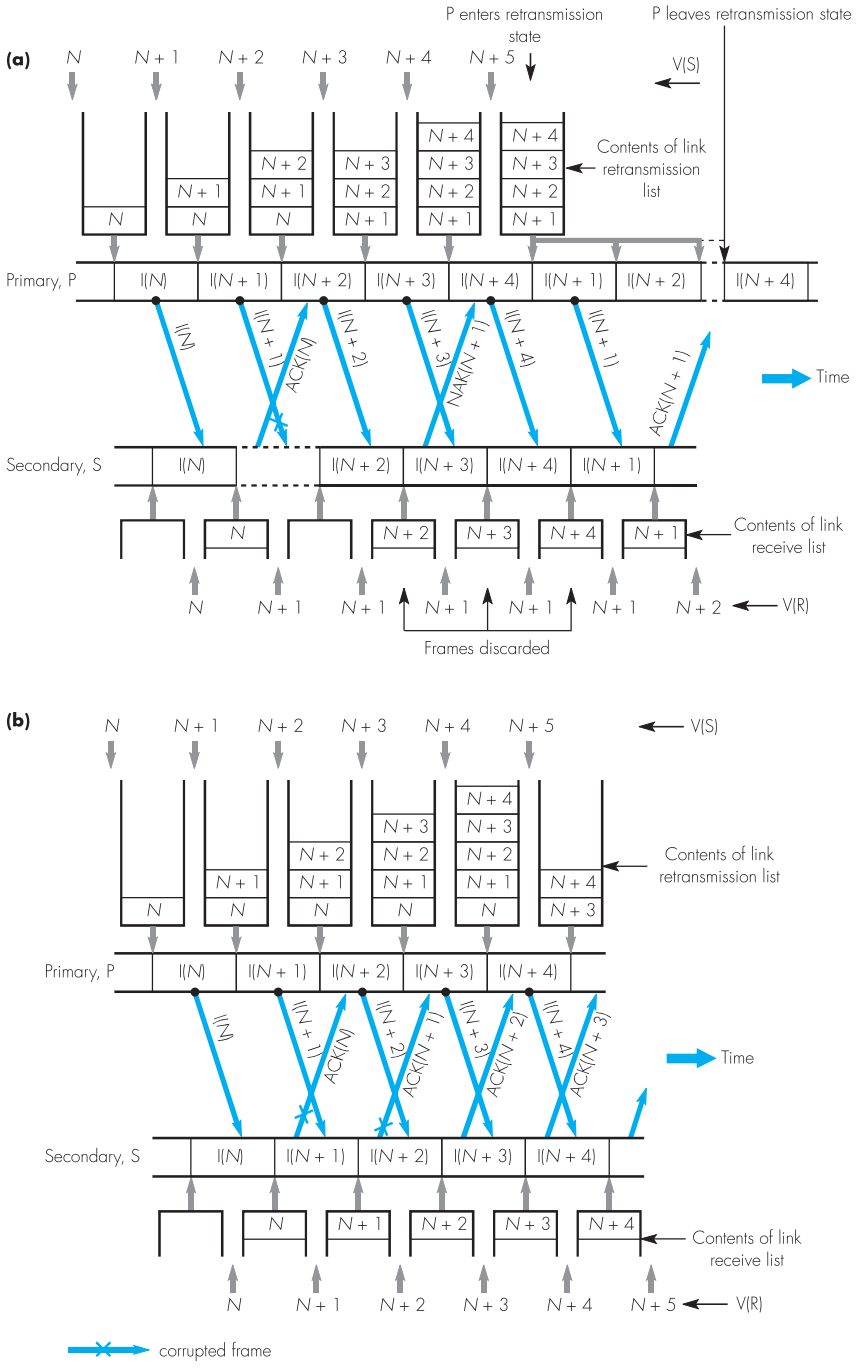


Figure 1.28 Go-back-N retransmission strategy: (a) corrupted I-frame; (b) corrupted ACK-frame.

1.4.4 Flow control

Error control is only one component of a data link protocol. Another important and related component is flow control. As the name implies, it is concerned with controlling the rate of transmission of frames on a link so that the receiver always has sufficient buffer storage resources to accept them prior to processing.

To control the flow of frames across a link, a mechanism known as a **sliding window** is used. The approach is similar to the idle RQ control scheme in that it essentially sets a limit on the number of I-frames that P may send before receiving an acknowledgment. P monitors the number of outstanding (unacknowledged) I-frames currently held in the retransmission list. If the destination side of the link is unable to pass on the frames sent to it, S stops returning acknowledgment frames, the retransmission list at P builds up and this in turn can be interpreted as a signal for P to stop transmitting further frames until acknowledgments start to flow again.

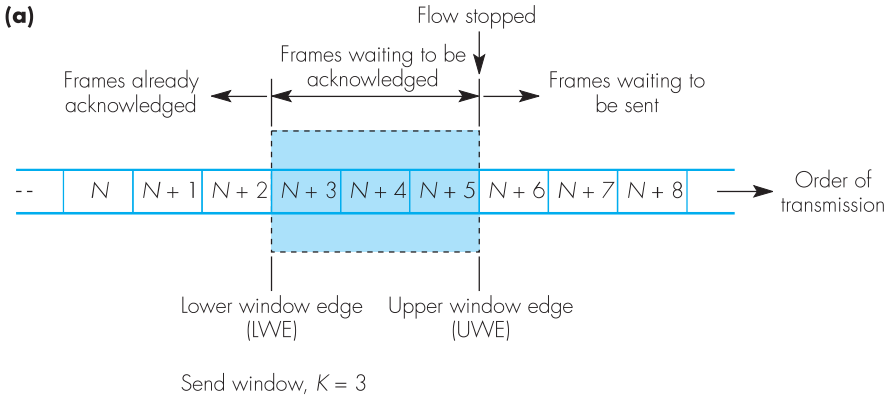
To implement this scheme, a maximum limit is set on the number of I-frames that can be awaiting acknowledgment and hence are outstanding in the retransmission list. This limit is the **send window**, ***K*** for the link. If this is set to 1, the retransmission control scheme reverts to idle RQ with a consequent drop in transmission efficiency. The limit is normally selected so that, providing the destination is able to pass on or absorb all frames it receives, the send window does not impair the flow of I-frames across the link. Factors such as the maximum frame size, available buffer storage, link propagation delay, and transmission bit rate must all be considered when selecting the send window.

The operation of the scheme is shown in Figure 1.29. As each I-frame is transmitted, the **upper window edge** (**UWE**) is incremented by unity. Similarly, as each I-frame is acknowledged, the **lower window edge** (**LWE**) is incremented by unity. The acceptance of any new message blocks, and hence the flow of I-frames, is stopped if the difference between UWE and LWE becomes equal to the send window *K*. Assuming error-free transmission, *K* is a fixed window that moves (slides) over the complete set of frames being transmitted. The technique is thus known as "sliding window".

The maximum number of frame buffers required at S is known as the **receive window**. We can deduce from the earlier frame sequence diagrams that with the idle RQ and go-back-N schemes only one buffer is required. With selective repeat, however, *K* frames are required to ensure frames are delivered in the correct sequence.

1.4.5 Sequence numbers

Until now, we have assumed that the sequence number inserted into each frame by P is simply the previous sequence number plus one and that the range of numbers available is infinite. Defining a maximum limit on the number of I-frames being transferred across a link not only limits the size of the link retransmission and receive lists, but also makes it possible to limit the



(b)

Protocol	Send window	Receive window
Idle RQ	1	1
Selective repeat	*K*	*K*
Go-back-N	*K*	1

Figure 1.29 Flow control principle: (a) sliding window example; (b) send and receive window limits.

range of sequence numbers required to identify each transmitted frame uniquely. The number of identifiers is a function of both the retransmission control scheme and the size of the send and receive windows.

For example, with an idle RQ control scheme, the send and receive windows are both 1 and hence only two identifiers are required to allow S to determine whether a particular I-frame received is a new frame or a duplicate. Typically, the two identifiers are 0 and 1; the send sequence variable is incremented modulo 2 by P.

With a go-back-N control scheme and a send window of *K*, the number of identifiers must be at least $K+1$. We can deduce this by considering the case when P sends a full window of *K* frames but all the ACK-frames relating to them are corrupted. If only *K* identifiers were used, S would not be able to determine whether the next frame received is a new frame – as it expects – or a duplicate of a previous frame.

With a selective repeat scheme and a send and receive window of *K*, the number of identifiers must not be less than $2K$. Again, we can deduce this by considering the case when P sends a full window of *K* frames and all subsequent acknowledgments are corrupted. S must be able to determine whether any of the next *K* frames are new frames. The only way of ensuring that S can deduce this is to assign a completely new set of *K* identifiers to the next window of I-frames transmitted, which requires at least $2K$ identifiers. The limits for each scheme are summarized in Figure 1.30(a).

(a)

Protocol	Maximum number of frame identifiers
Idle RQ	2
Selective repeat	$2K + 1$
Go-back-N	$K + 1$

(b)

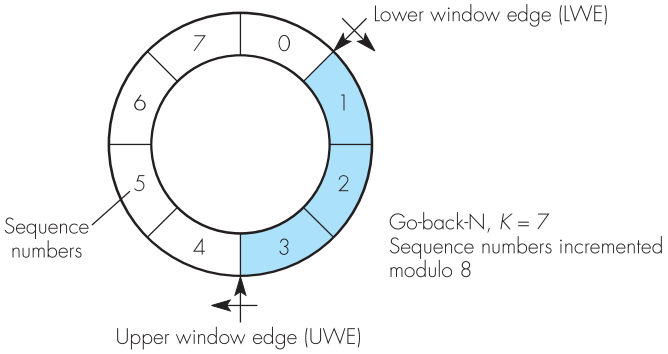


Figure 1.30 Sequence numbers: (a) maximum number for each protocol; (b) example assuming eight sequence numbers.

In practice, since the identifier of a frame is in binary form, a set number of binary digits must be reserved for its use. For example, with a send window of, say, 7 and a go-back-N control scheme, three binary digits are required for the send and receive sequence numbers yielding eight possible identifiers: 0 through 7. The send and receive sequence variables are then incremented modulo 8 by P and S respectively. This is illustrated in Figure 1.30(b).

1.4.6 Layered architecture

The frame sequence diagrams that we showed earlier provide a qualitative description of the error control (and flow control) components of a link layer protocol that is based on an idle RQ scheme – Figure 1.23 – and a continuous RQ scheme – Figures 1.26–1.28. In practice, however, it is not possible to describe fully the operation of all aspects of a protocol using just this method. Normally, therefore, the operation of a protocol is specified in a more formal way and, in order to gain an insight into how this is done, we shall specify the error control component of the idle RQ error control scheme.

Before we do this, we revisit the subject of layering, which involves decoupling each protocol layer one from another and defining a formal interface between them. Assuming an idle RQ scheme, a suitable layered architecture for the link layer is as shown in Figure 1.31. As we have described, the service provided to the network layer in the source is to transfer in a reliable way a series of blocks of information to the network layer in the destination. Also, depending on the BER probability of the line, a maximum block size will be

specified that ensures a good percentage of I-frames transmitted will be free of errors.

As we show in Figure 1.31, in order to decouple the network and link layers in each system, we introduce a queue between them. Each queue is simply a data structure that implements a first-in, first-out (FIFO) queuing discipline. Elements are added to the tail of the queue and are removed from the head.

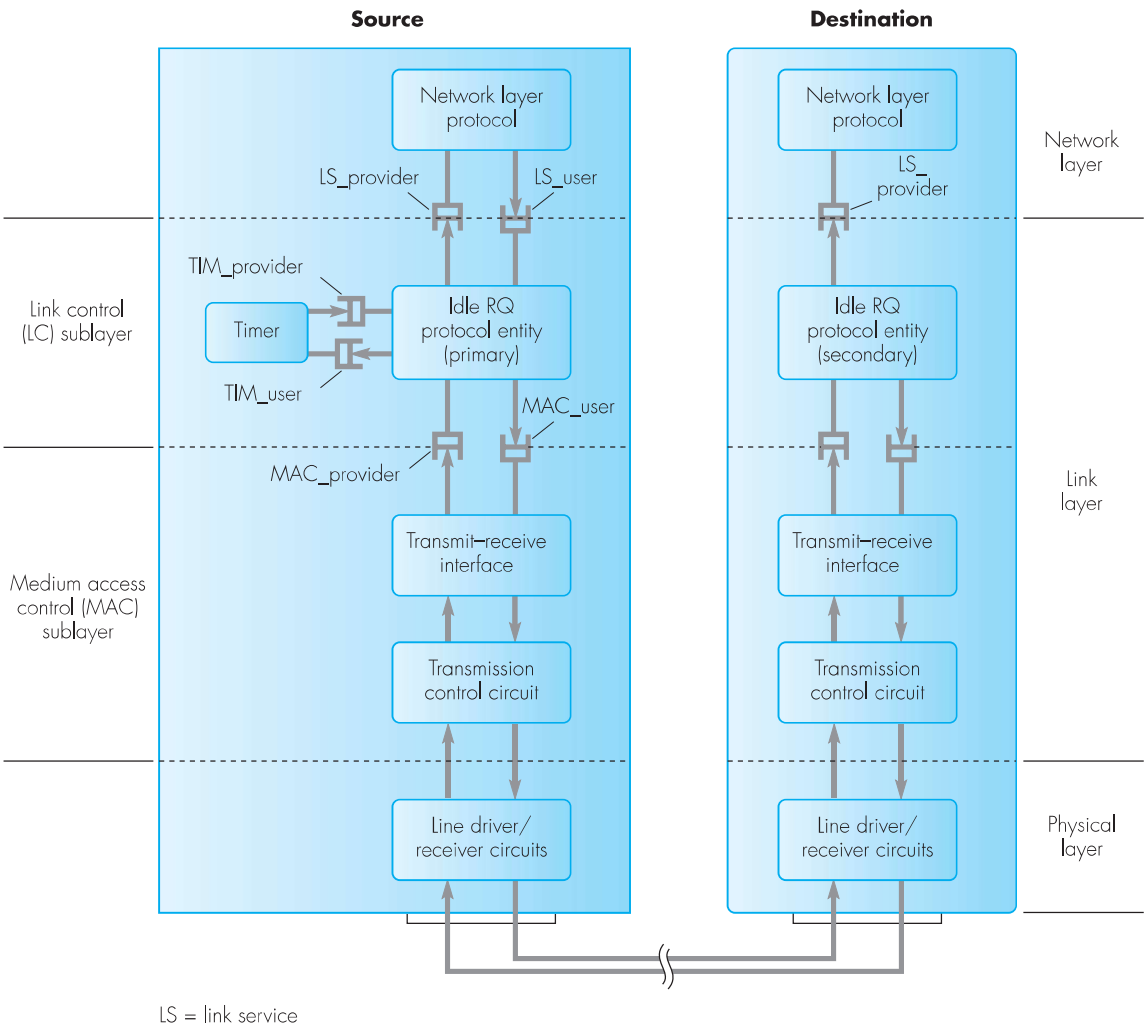


Figure 1.31 Example layered architecture showing the layer and sublayer interfaces associated with the idle RQ protocol.

Normally, the user service primitive(s) associated with a layer is/are passed between layers using a data structure known as an **event control block** (**ECB**). This has the primitive type in the first field and an array containing the user data in a second field. For example, whenever the network layer protocol wishes to send a message block – the contents of a data structure – it first obtains a free ECB, sets the primitive type field to L_DATA.request, writes the address pointer of the data structure in the user data field, and inserts the ECB at the tail of the link layer (LS_user) input queue ready for reading by the idle RQ primary.

When the idle RQ protocol software is next run, it detects the presence of an entry (ECB) in the LS_user queue, reads the entry from the head of the queue, and proceeds to create an I-frame containing the send sequence number and the contents of the user data field. It then initiates the transmission of the frame to the secondary protocol using the services provided by the transmission control circuit. Normally, as we show in the figure, associated with the latter is a piece of low-level software that, in practice, is part of the **basic input-output software** (**BIOS**) of the computer. Assuming bit-oriented transmission, as the frame contents are being output, the transmission control circuit generates the CRC for the frame and adds the start and end flags. As we can deduce from this, therefore, the link layer comprises two sublayers: the **link control** (**LC**) – which is concerned with the implementation of the error and flow control procedures that are being used and is independent of the type of transmission control mode – and the **medium access control** (**MAC**) **sublayer** – which is concerned with the transmission of preformatted blocks using a particular transmission control mode which may vary for different networks. The physical layer comprises suitable bit/clock encoding circuits, line driver and receiver circuits, and the plug and socket pin definitions.

At the destination, assuming the received frame is error free, the MAC sublayer passes the frame contents to the LC sublayer using a MAC_DATA. indication primitive in an ECB and the MAC_provider queue. The LC sublayer then uses the send sequence number at the head of the frame to confirm it is not a duplicate and passes the frame contents – the message block – up to the network layer in an ECB using the LS_provider output queue with the primitive type set to L_DATA.indication. It then creates and returns an ACK-frame to P using a MAC_DATA.request primitive in an ECB and the MAC_user queue.

When the destination network layer protocol is next run, it detects and reads the ECB from the LS_provider queue and proceeds to process the contents of the message block it contains according to the defined network layer protocol. At the sending side, assuming the ACK-frame is received free of errors, the MAC sublayer passes the frame to the LC sublayer primary, which frees the memory buffer containing the acknowledged I-frame and checks the LS_user input queue for another waiting ECB. If there is one, the procedure is repeated until all queued blocks have been transferred. Note

that the LS_user queue and the retransmission list are quite separate. The first is used to hold new message blocks waiting to be transmitted and the second to hold frames – containing blocks – that have already been sent and are waiting to be acknowledged.

We can conclude that the adoption of a layered architecture means that each layer performs its own well-defined function in relation to the overall communications task. Each layer provides a defined service to the layer immediately above it. The service primitives associated with the service are each implemented by the layer protocol communicating with a peer layer protocol in the remote system. Associated with the protocol are protocol data units (PDUs) – for example, I-frame, ACK-frame, and so on in the case of the link layer – and these are physically transferred using the services provided by the layer immediately below it.

1.4.7 Protocol specification

Irrespective of the specification method that is used, we model a protocol as a **finite state machine** or **automaton**. This means that the protocol – or, more accurately, the **protocol entity** – can be in just one of a finite number of defined **states** at any instant. For example, it might be idle waiting for a message to send, or waiting to receive an acknowledgment. Transitions between states take place as a result of an incoming event, for example, a message becomes ready to send, or an ACK-frame is received. As a result of an incoming event, an associated **outgoing event** is normally generated, for example, on receipt of a message, format and send the created I-frame on the link, or on receipt of a NAK-frame, retransmit the waiting I-frame.

Some incoming events may lead to a number of possible outgoing events. The particular outgoing event selected is determined by the computed state of one or more **predicates** (boolean variables). As an example, predicate P1 may be true if the N(R) in a received ACK-frame is the same as the N(S) in the I-frame waiting to be acknowledged. Hence, if P1 is true, then free the memory buffer in which the I-frame is being held; if it is false, initiate retransmission of the frame.

An incoming event, in addition to generating an outgoing event (and possibly a change of state), may also have one or more associated **local** or **specific actions**. Examples include *start a timer* and *increment the send sequence variable.*

We shall now expand upon all of these aspects of the specification of a protocol by considering the specification of the error control procedure associated with the idle RQ protocol. To simplify the description, we shall consider only a unidirectional flow of I-frames – from the source to the destination. In most applications, however, a two-way flow is needed and both sides require a primary and a secondary.

All finite state machines – and hence protocol entities – operate in an atomic way. This means that once an incoming event has started to be

processed, all processing functions associated with the event, including the generation of any outgoing event(s), local (specific) actions, and a possible change in state, are carried out in their entirety (that is, in an indivisible way) before another incoming event is accepted.

To ensure this happens, the various incoming (and outgoing) event interfaces are decoupled from the protocol entity itself by means of queues. As we showed earlier in Figure 1.31, there is an additional pair of queues between the protocol entity and the transmit–receive procedure that controls the particular transmission control circuit being used. Similarly, there is a pair of queues between the protocol entity and the timer procedure. Normally, the latter is run at regular (tick) intervals by means of an **interrupt** and, if a timer is currently running, its current value is decremented by the tick value. If the value goes to zero, a **timer expired** message is returned to the protocol entity via the appropriate queue.

The role of the transmit–receive procedure is simply to transmit a preformatted frame passed to it or to receive a frame from the link and queue the frame for processing by the protocol entity. This procedure may also be run as a result of an interrupt, but this time from the transmission control circuit. Also, although in principle only a single input and output queue is necessary to interface the primary and secondary to their respective network layers, in practice a pair of queues is necessary at each interface in order to handle the duplex flows of primitives.

To simplify the specification procedure, we give each of the various incoming events, outgoing events, predicates, specific actions, and states associated with each protocol entity an abbreviated name. Prior to specifying the protocol, the various abbreviated names are listed and all subsequent references are made using these names. For the error control component of the idle RQ protocol, the list of abbreviated names for the primary is as shown in Figure 1.32.

Since each protocol entity is essentially a sequential system, we must retain information that may vary as different incoming events are received. This information is held in a number of **state variables**. Examples, for the primary, are the send sequence variable V(S) – Vs in the specification – which holds the sequence number to be allocated to the next I-frame to be transmitted; the PresentState variable, which holds the present state of the protocol entity; and RetxCount, which is a count of the number of erroneous frames received. Typically, if either RetxCount or ErrorCount reaches its maximum limit then the frame is discarded, an error message is sent to the network layer above and the protocol (entity) reinitializes.

The three most common methods that are used for specifying a communication protocol are **state transition diagrams**, **extended event–state tables**, and high-level structured programs. In many instances, we define a protocol as a combination of these coupled with time sequence diagrams to illustrate the user service primitives associated with the protocol.

The formal specification of the primary is shown in Figure 1.33. In part (a) a state transition diagram is used, in part (b) an extended event–state table, and in part (c) structured pseudocode.

Incoming events

Name	*Interface*	*Meaning*
LDATAreq	LS_user	L_DATA.request service primitive received
ACKRCVD	MAC_provider	ACK-frame received from S
TEXP	TIM_provider	Wait-ACK timer expires
NAKRCVD	MAC_provider	NAK-frame received from S

States

Name	*Meaning*
IDLE	Idle, no message transfer in progress
WTACK	Waiting an acknowledgment

Outgoing events

Name	*Interface*	*Meaning*
TxFrame	MAC_user	Format and transmit an I-frame
RetxFrame	MAC_user	Retransmit I-frame waiting acknowledgment
LERRORind	LS_provider	Error message: frame discarded for reason specified

Predicates

Name	*Meaning*
P0	N(S) in waiting I-frame = N(R) in ACK-frame
P1	CRC in ACK/NAK-frame correct

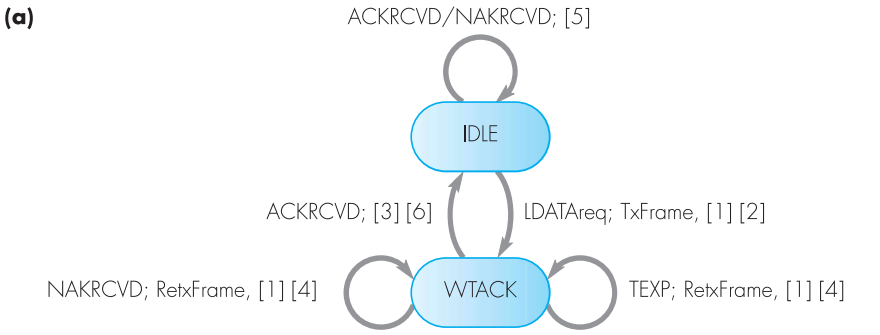
Specific actions

[1] = Start_timer using TIM_user queue
[2] = Increment Vs
[3] = Stop_timer using TIM_user queue
[4] = Increment RetxCount
[5] = Increment ErrorCount
[6] = Reset RetxCount to zero

State variables

Vs	=	Send sequence variable
PresentState	=	Present state of protocol entity
ErrorCount	=	Number of erroneous frames received
RetxCount	=	Number of retransmissions for this frame

Figure 1.32 Abbreviated names used in the specification of the idle RQ primary.



(*continued overleaf*)

Figure 1.33 Specification of idle RQ primary in the form of: (a) a state transition diagram; (b) an extended event–state table; (c) pseudocode.

(b)

Present state \ Incoming event	LDATAreq	ACKRCVD	TEXP	NAKRCVD
IDLE	1	0	0	0
WTACK	4	2	3	3

0 = [5], IDLE (error condition)
1 = TxFrame, [1] [2], WTACK
2 = P0 and P1: [3] [6], IDLE
P0 and NOT P1: RetxFrame, [1] [4], WTACK
NOT P0 and NOT P1: [5], IDLE
3 = RetxFrame, [1] [4], WTACK
4 = NoAction, WTACK

(c)

```
program IdleRQ_Primary;
const   MaxErrCount;
        MaxRetxCount;
type    Events = (LDATAreq, ACKRCVD, TEXP, NAKRCVD);
        States = (IDLE, WTACK);
var     EventStateTable = array [Events, States] of 0..4;
        PresentState : States;
        Vs, ErrorCount, RetxCount : integer;
        EventType : Events;
procedure       Initialize;     }   Initializes state variables and contents of EventStateTable
procedure       TxFrame;
procedure       RetxFrame;      |   Outgoing event procedures
procedure       LERRORind;
procedure       Start_timer;    }   Specific action procedures
procedure       Stop_timer;
function  P0 : boolean;   | Predicate functions
function  P1 : boolean;

begin   Initialize;
        repeat  Wait receipt of an incoming event
                EventType := type of event
                case EventStateTable [EventType, PresentState] of
                   0: begin     ErrorCount := ErrorCount + 1; PresentState = IDLE;
                              if(ErrorCount = MaxErrCount) then LERRORind end;
                   1: begin     TxFrame; Start_timer; Vs := Vs + 1; PresentState := WTACK end;
                   2: begin     if(P0 and P1) then begin Stop_timer; RetxCount := 0; PresentState := IDLE
end;
                                else if (P0 and NOTP1) then begin       RetxFrame; Start_timer;
                                                                        RetxCount := RetxCount + 1;
                                                                        PresentState := WTACK end;
                                else if (NOTP0 and NOTP1) then begin PresentState := IDLE; ErrorCount := ErrorCount +
1 end;
                                        if (ErrorCount = MaxErrorCount) then begin LERRORind; Initialize; end;
                              end;
                   3: begin RetxFrame; Start_timer; RetxCount := RetxCount + 1; PresentState := WTACK;
                              if (RetxCount = MaxRetxCount) then begin LERRORind; Initialize; end;
                              end;
                   4: begin NoAction end;
        until Forever;
end.
```

Figure 1.33 Continued.

Using the state transition diagram method, the possible states of the protocol entity are shown in ovals with the particular states written within them. **Directional arrows** (also known as **arcs**) indicate the possible transitions between the states, with the incoming event causing the transition and any resulting outgoing event and specific actions written alongside. If, for example, an L_DATA.request (LDATAreq) is received from LS_user interface, then the frame is formatted and output to the MAC_user interface (TxFrame), a timer is started for the frame [1], the send sequence variable incremented [2], and the WTACK state entered. Similarly, if an ACK-frame is received with an N(R) equal to the N(S) in the waiting frame and the CRC is correct, then the timer is stopped [3] and the transitions can be interpreted in a similar way.

Although state transition diagrams are useful for showing the correct operation of a protocol, because of space limitations it is not always practicable to show all possible incoming event possibilities including error conditions. Hence most state transition diagrams are incomplete specifications. Moreover, with all but the simplest of protocols, we need many such diagrams to define even the correct operation of a protocol. It is for these reasons that we use the extended event–state table and the structured program code methods.

Using the extended event–state table method – as we see in part (b) of the figure – we can show all the possible incoming events and protocol (present) states in the form of a table. For each state, the table entry defines the outgoing event, any specific action(s), and the new state for all possible incoming events. Also, if predicates are involved, it defines the alternative set of action(s). Clearly, the extended event–state table is a far more rigorous method since it allows for all possible incoming-event, present-state combinations. A basic event–state table has only one possible action and next-state for each incoming-event/present-state combination. It is the presence of predicates – and hence possible alternative actions/next states – that gives rise to the use of the term "extended" event–state table.

When we are interpreting the actions to be followed if predicates are involved, we must note that these are shown in order. Hence the action to be followed if the primary is in the WTACK state and an ACK-frame is received (ACKRCVD), is first to determine if P0 and P1 are both true. If they are, then carry out specific action [3] and [6] and enter the IDLE state. Else, determine if {P0 and NOTP1} is true, and so on. If neither condition is true then an error is suspected and the actions are shown.

A feature of the extended event–state table is that it lends itself more readily to implementation in program code than a state transition diagram. We can see this by considering the pseudocode specification of the idle RQ primary in Figure 1.33(c). In the figure this is shown as a program but in practice it is implemented in the form of a procedure or function so it can be included with the other protocol layers in a single program. However, this does not affect the basic operation of the program shown.

When each program (layer) is first run, the Initialize procedure is invoked. This performs such functions as initializing all state variables to their initial values and the contents of the EventStateTable array to those in the extended event–state table. The program then enters an infinite loop waiting for an incoming event to arrive at one of its input queues.

The incoming event that causes the program to run is first assigned to EventType. The current contents of PresentState and EventType are then used as indices to the EventStateTable array to determine the integer – 0, 1, 2, 3, or 4 – that defines the processing actions associated with that event. For example, if the accessed integer is 2, this results in the predicate functions P0 and P1 being invoked and, depending on their computed state (true or false), the invocation of the appropriate outgoing event procedure, coupled with any specific action procedures(s) as defined in the specification; for example, starting or resetting the timer, and updating PresentState.

We have simplified the pseudocode to highlight the structure of each program and hence the implementation methodology. No code is shown for the various outgoing event procedures nor for the predicate functions. In practice, these must be implemented in an unambiguous way using the necessary steps listed in the specification.

1.4.8 User service primitives

As we explained in the last section, to initiate the transfer of a block of information across the transmission line/link, the source network layer uses an ECB with a L_DATA.request primitive and the block of information within it. Similarly, the destination link layer (protocol), on receipt of an error-free I-frame containing the block of information, also uses an ECB to pass the block to the network layer with a L_DATA.indication primitive within it.

Example 1.7

Use the frame sequence diagram shown earlier in Figure 1.23 and the list of abbreviated names given in Figure 1.34(a) to specify the operation of the idle RQ secondary using (i) a state transition diagram, (ii) an extended event–state table, (iii) pseudocode.

Answer:

The specification of the idle RQ secondary in each form is given in Figure 1.34(b), (c), and (d) respectively. Note that just two state variables are needed for the secondary: the receive sequence variable – shown as V_r in the specification – which holds the sequence number of the last correctly received I-frame, and ErrorCount which keeps a record of the number of erroneous I-frames received. Again, if ErrorCount reaches a defined maximum limit an error message – LERRORind – is output to the network layer in an ECB.

(a) **Incoming events**

Name	*Interface*	*Meaning*
IRCVD	MAC_provider	I-frame received from P

States

Name	*Meaning*
WTIFM	Waiting a new I-frame from P

Outgoing events

Name	*Interface*	*Meaning*
LDATAind	LS_provider	Pass contents of received I-frame to user AP with L_DATA.indication primitive
TxACK(X)	MAC_user	Format and transmit an ACK-frame with N(R) = X
TxNAK(X)	MAC_user	Format and transmit a NAK-frame with N(R) = X
LERRORind	LS_provider	Issue error message for reason specified

Predicates

Name	*Meaning*
P0	N(S) in I-frame = Vr
P1	CRC in I-frame correct
P2	N(S) in I-frame = Vr − 1

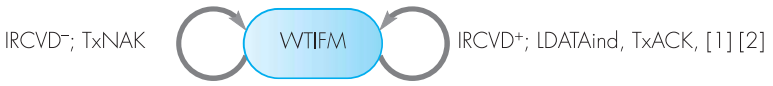
Specific actions

[1] = Increment Vr
[2] = Increment ErrorCount

State variables

Vr = Receive sequence variable
ErrorCount = No. of erroneous frames received

(b)



(c)

Present state \ Incoming event	IRCVD
WTIFM	1

1 = NOT P1: TxNAK, [2]
P1 and P2: TxACK
P0 and P1: LDATAind, TxACK, [1]

(*continued overleaf*)

Figure 1.34 Specification of idle RQ secondary: (a) abbreviated names; (b) state transition diagram; (c) extended event–state table; (d) pseudocode.

(d)

```
program IdleRQ_Secondary;
const.  MaxErrorCount;
type    Events = IRCVD;
        States = WTIFM;
var     EventStateTable = array [Events, States] of 1;
        EventType : Events;
        PresentState : States;
        Vr, X, ErrorCount : integer;
procedure       Initialize;     }       Initializes state variables and contents of EventStateTable
procedure       LDATAind;
procedure       TxACK(X);       |       Outgoing event procedures
procedure       TxNAK(X);
procedure       LERRORind;
function  P0 : boolean;
function  P1 : boolean;         |       Predicate functions
function  P2 : boolean;

begin   Initialize;
        repeat  Wait receipt of incoming event; EventType := type of event;
                case EventStateTable[EventType, PresentState] of
                   1 : X := N(S) from I-frame;
                      if (NOTP1) then TxNAK(X);
                      else if(P1 and P2) then TxACK(X);
                      else if(P0 and P1) then begin LDATAind; TxACK(X); Vr := Vr + 1; end;
                      else  begin                     ErrorCount := ErrorCount + 1; if (ErrorCount =
MaxErrorCount) then
                                  begin LERRORind; Initialize; end;
                           end;
        until Forever;
end.
```

Figure 1.34 Continued.

Finally, for the error and flow control schemes we have outlined in the previous sections to function correctly, we have assumed that both communicating link protocols have been initialized so that they are ready to exchange information. For example, both sides of the link must start with the same send and receive sequence variables before any information frames are transmitted. In general, this is known as the initialization or **link setup** phase and, after all data has been exchanged across a link, there is a **link disconnection** phase. Since the link setup and disconnection phases are not concerned with the actual transfer of user data, they are collectively referred to as **link management**. The two link management functions are also initiated by the network layer (protocol) using an ECB and the set of primitives that we show in Figure 1.35(a). Since the primitives shown are in the same sequence as they are issued, this form of representing the various user service primitives associated with a protocol is known as a **time sequence diagram**. Note that to avoid the diagram becoming too cluttered, we have left off the two error indication primitives.

On receipt of an **L_CONNECT.request** primitive, the link protocol entity at the source initializes all state variables and then creates a **link SETUP** frame (known as a **protocol data unit**(PDU)). This is sent to the correspondent (peer) link protocol entity in the destination using the selected transmission mode. On receipt of the SETUP frame, the destination initializes

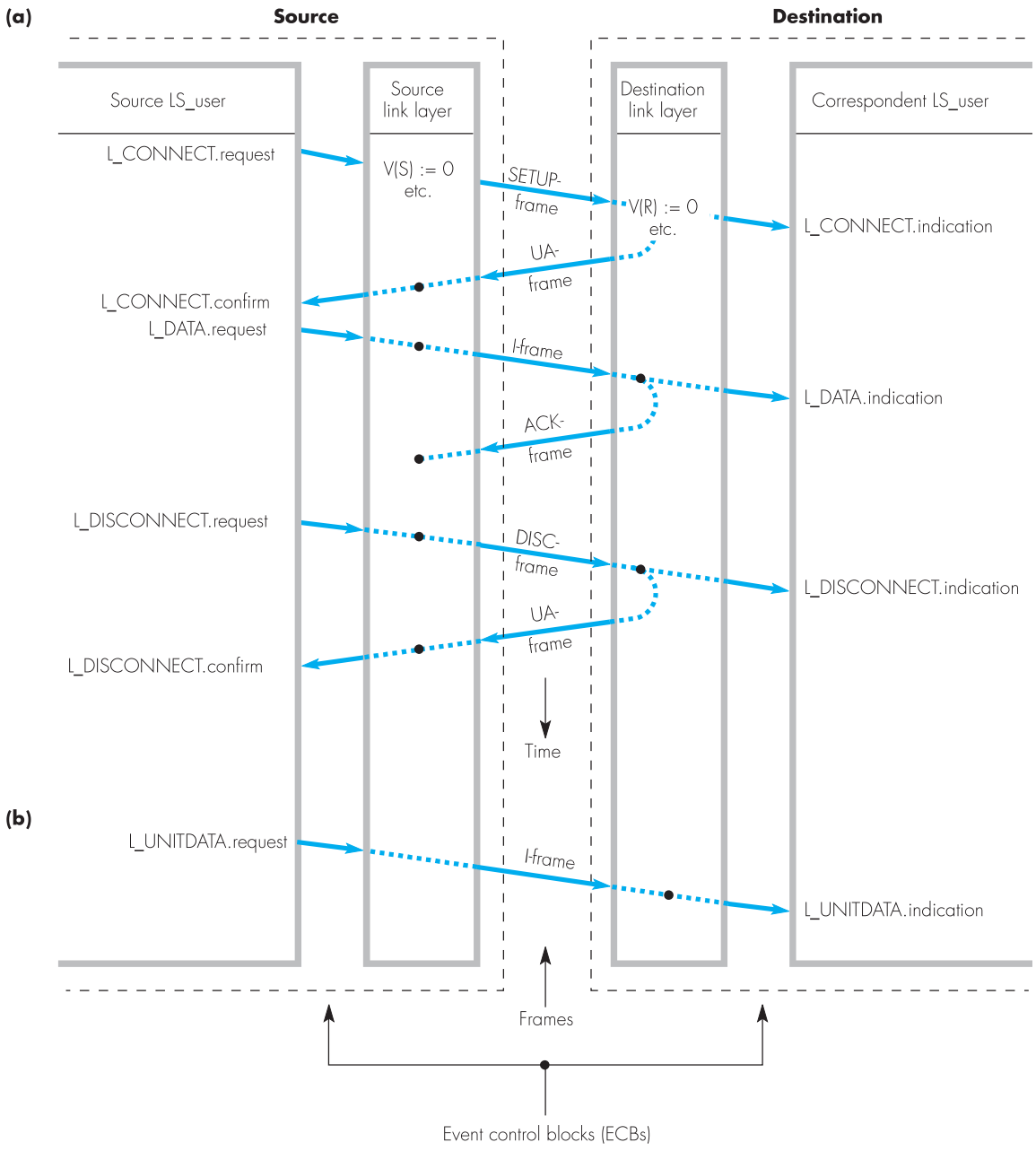


Figure 1.35 Time sequence diagram showing the link layer service primitives: (a) connection-oriented (reliable) mode; (b) connectionless (best-effort) mode.

its own state variables and proceeds by sending an **L_CONNECT.indication** primitive to the correspondent LS_user and an acknowledgment frame back to the source.

Since this acknowledgment does not relate to an I-frame, it does not contain a sequence number. It is known, therefore, as an **unnumbered acknowledgment** or **UA-frame**. On receipt of this UA-frame, the source protocol entity issues the **L_CONNECT.confirm** primitive to the LS_user and the link is now ready for the transfer of data using the L_DATA service. Finally, after all data has been transferred, the setup link is released using the L_DISCONNECT service, which is also a confirmed service. The corresponding frame, known as a **disconnect** or **DISC frame**, is acknowledged using a UA-frame.

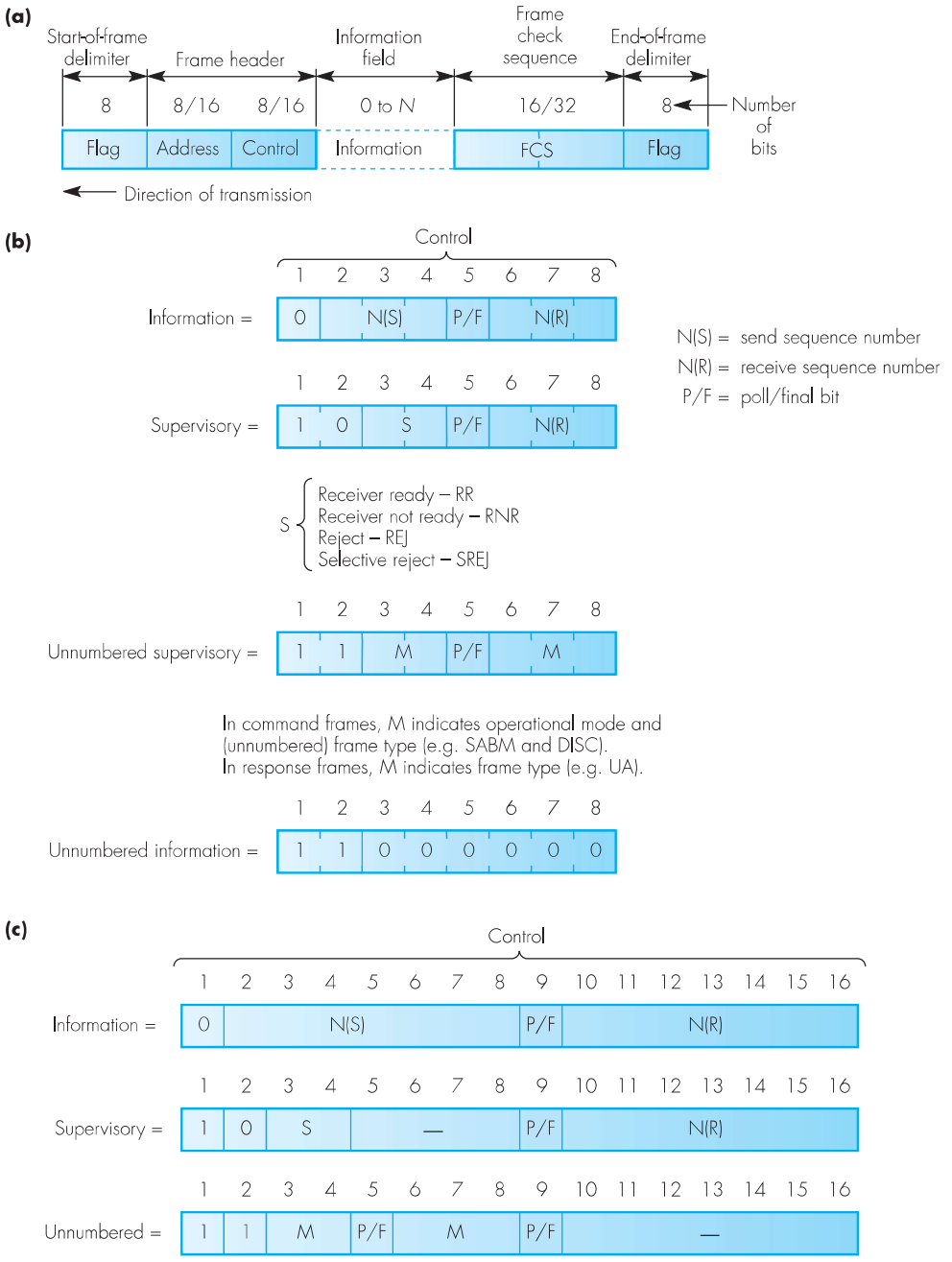
This mode of operation of the link layer is known as the connection-oriented mode and, as we have explained, it provides a reliable service. As we indicated at the start of Section 1.4, however, in many applications this mode of operation is not used and instead the simpler best-effort service is used in which the link layer protocol at the destination simply discards any frames received with errors. The two user service primitives associated with this mode are shown in Figure 1.35(b).

Since it is not necessary to set up a logical connection prior to sending blocks of information, this mode of operation is known as the connectionless mode and, in order to discriminate between the data transfer service associated with the two modes, as we show in the figure, the two service primitives used are **L_UNITDATA.request** and **L_UNITDATA.indication**. Also, since in the connectionless mode no retransmissions are used, no sequence numbers are required. Note, however, that the differences between the two modes occur only at the link control layer.

1.4.9 The HDLC protocol

To conclude this section, we describe selected aspects of a practical example of a link layer protocol known as the **high-level data link control** (**HDLC**) protocol. This is an international standard that has been defined for use with a number of different network configurations and types. These include duplex point-to-point links as used over the access circuits associated with an ISDN, and half-duplex multidrop/broadcast links as used in some LANs. Hence there is the original HDLC protocol and a number of variants of it, each of which uses slightly different fields in the frame header and also a different link control layer. Examples include the **link access procedure D-channel** (**LAPD**) which is used with an ISDN and the **logical link control** (**LLC**) which is used with LANs.

In HDLC, the frames sent by the primary to the secondary are known as **commands**, and those from the secondary to the primary as **responses**. Also, when the link control layer is operating in a connection-oriented (reliable) mode, all error and flow control frames are known as **supervisory frames** and the various frames that are used to set up and disconnect a logical link are known as **unnumbered frames**. A standard format is used for all frames, however, and this is shown in Figure 1.36(a).



Note: With the indicated direction of transmission, all control field types are transmitted bit 8/16 first.

Figure 1.36 HDLC frame format and types: (a) standard/extended format; (b) standard control field bit definitions; (c) extended control field bit definitions.

As we can see, HDLC is based on a bit-oriented transmission control scheme with flags to indicate the start and end of each frame together with zero bit insertion and deletion to ensure the flag pattern (01111110) does not occur within the bitstream between the flags. The frame check sequence (FCS) is a 16-bit CRC that is computed using the generator polynomial:

$$x^{16} + x^{12} + x^5 + 1$$

The CRC is first generated using the procedure we describe in Appendix B but an additional step is taken to make the check more robust. This involves adding sixteen 1s to the tail of the dividend (instead of zeros) and inverting the remainder. This has the effect that the remainder computed by the receiver is not all zeros but the bit pattern 0001 1101 0000 1111.

The various control field bit definitions are shown in Figure 1.36(b). The S-field in supervisory frames and the M-field in unnumbered frames are used to define the specific frame type. The send and receive sequence numbers – N(S) and N(R) – are used in conjunction with the error and flow control procedures.

The **P/F bit** is known as the **poll/final bit**. A frame of any type is called a **command frame** if it is sent by the primary station and a **response frame** if it is sent by a secondary station. The P/F bit is called the poll bit when used in a command frame and, if set, indicates that the receiver must acknowledge this frame. The receiver acknowledges this frame by returning an appropriate response frame with the P/F bit set; it is then known as the final bit.

The use of three bits for each of N(S) and N(R) means that sequence numbers can range from 0 through 7. This, in turn, means that a maximum send window of 7 can be selected. Although this is large enough for many applications, those involving very long links (satellite links, for example) or very high bit rates require a larger send window if a high link utilization is to be achieved. The extended format uses seven bits (0 through 127), thereby increasing the maximum send window to 127. This is shown in Figure 1.36(c).

The address field identifies the secondary station that sent the frame, and is not needed with point-to-point links. However, with multipoint links, the address field can be either eight bits – normal mode – or multiples of eight bits – extended mode. In the latter case, bit 1 of the least significant address octet(s) is/are set to 0 and bit 1 in the last octet is set to 1. The remaining bits form the address. In both modes, an address of all 1s is used as an all-stations broadcast address.

Unnumbered frames are used both to set up a logical link between the primary and secondary and to inform the secondary of the mode of operation to be used. For example, the set asynchronous balanced mode (SABM) command frame – indicated by the M-bits in the control field – is used to set up a logical link in both directions when a duplex point-to-point

link is being used. Other examples are the DISC-frame (which is used to disconnect the link) and the UA-frame, which is a response frame and is sent to acknowledge receipt of the other (command) frames in this class. Also, when operating in the connectionless (best-effort) mode, no acknowledgment information is required. Hence all information is transmitted in **unnumbered information** (**UI**) frames with the control field set to 11000000.

The four supervisory frames are used to implement a continuous RQ error control scheme and have the following functions:

- receiver ready (RR): this has the same function as the ACK-frame in Figures 1.27 and 1.28;
- reject (REJ): this has the same function as the NAK-frame in the go-back-N scheme;
- selective reject (SREJ): this has the same function as the NAK-frame in the selective repeat scheme;
- receiver not ready (RNR): this can be used by the secondary to ask the primary to suspend sending any new I-frames.

Each RR (ACK) frame contains a receive sequence number – N(R) – which acknowledges correct receipt of all I-frames awaiting acknowledgment up to and including that with an N(S) equal to N(R) – 1. This is slightly different from what we used earlier in the frame sequence diagrams in Figures 1.27 and 1.28 in which N(R) acknowledged I-frames up to N(S). This is because in HDLC the receive sequence variable V(R) is incremented before the ACK-frame is returned rather than after as we used in the earlier figures.

A summary of the various service primitives and frame types (protocol data units) associated with HDLC is given in Figure 1.37(a). In practice, there are more unnumbered frames associated with HDLC than are shown in the figure but, as mentioned earlier, the aim is simply to highlight selected aspects of HDLC operation. To reinforce understanding further, a (simplified) state transition diagram for HDLC is given in Figure 1.37(b). The first entry alongside each arc is the incoming event causing the transition (if any); the second entry is the resulting action. Note that a state transition diagram shows only the correct operation of the protocol entity; normally it is accompanied by a more complete definition in the form of an extended event–state table and/or pseudocode.

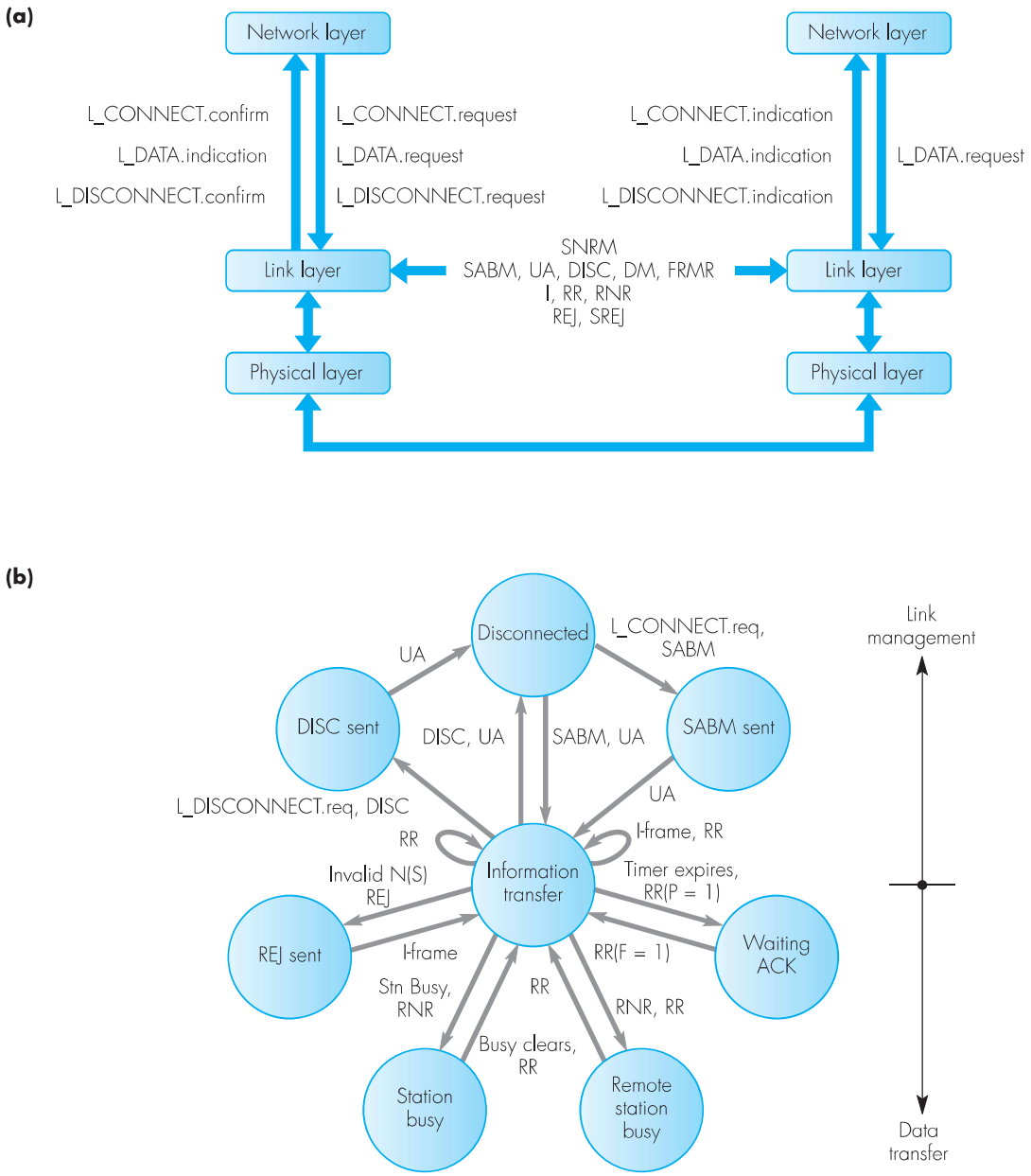


Figure 1.37 HDLC summary: (a) service primitives; (b) state transition diagram (ABM).

1.5 Protocol stacks

As we showed in Figure 1.37 of the previous section, the HDLC link layer protocol built on the underlying data transmission services of the physical layer to provide either a reliable or a best-effort service to transfer frames containing user data over a data link. The figure also showed how the HDLC protocol provided these services through a set of user service primitives. As we can deduce from this, therefore, the link layer protocol offers a defined set of user service primitives – part (a) – and implements these services by exchanging a sequence of frame types – called **protocol data units (PDUs)** – with the same link layer protocol in a remote system using the services provided by the underlying physical layer. Also, both the type and sequence of the PDUs that are exchanged are determined by the link layer protocol – part (b). This general arrangement is known as **layering** and is shown in diagrammatic form in Figure 1.38(a).

As we shall see, there are a number of protocol layers involved in data transfers over the Internet. Each layer performs a well-defined function in the context of the overall communication subsystem. The protocol used at each layer is chosen to meet the needs of a particular application/network combination. The two peer communicating protocol entities within each layer operate according to the specified protocol to implement the required function of the layer. This is achieved by adding appropriate **protocol control information (PCI)** to the head of the data being transferred. In the case of the HDLC protocol, for example, the PCI includes send and receive sequence numbers. The complete PDU is then transferred to the peer protocol entity in the remote system using the services provided by the layer below. The latter simply treats the PDU as user data, adds its own PCI to this to create a new PDU and proceeds to transfer the new PDU to the same peer layer in the remote system again using the services provided by the layer below.

1.5.1 The Internet protocol stack

As its name implies, the link layer protocol is concerned only with the transfer of user data – a PDU of the layer protocol immediately above it in practice – over a point-to-point physical link. Hence when communicating over a network, this is the physical link that connects the source end system/station to the nearest access point of the network; for example, an access gateway in the case of the Internet. A separate instance of the protocol is then needed to transfer the PDU over each physical link connecting the switches/routers that make up the network and also over the link connecting the remote access gateway to the destination end system/station. As we can deduce from

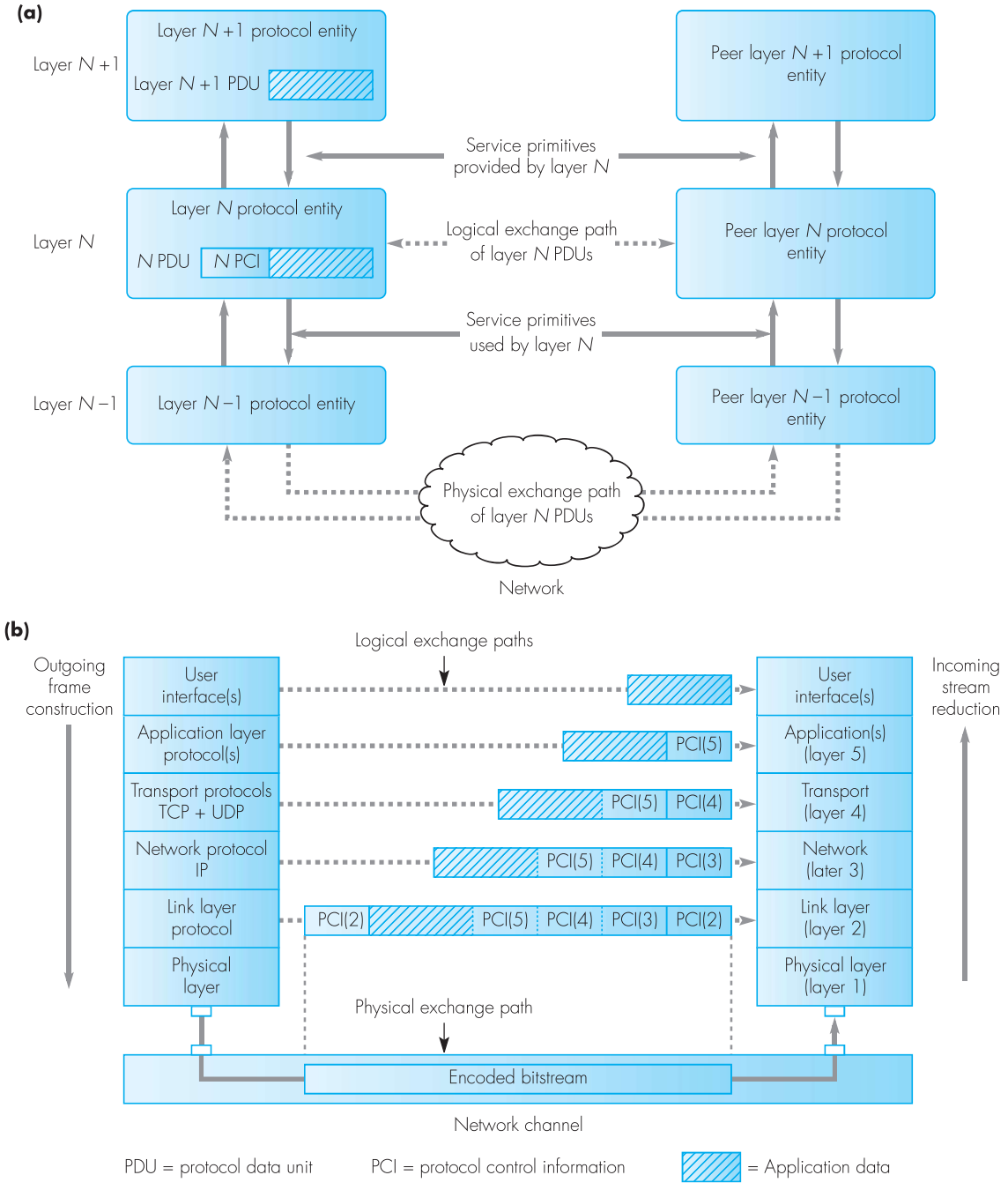


Figure 1.38 Protocol stacks: (a) layer interactions and terminology; (b) the Internet protocol stack.

this, therefore, an additional protocol is needed to route the PDU over the total network. This is the role of the **network layer** and, in the case of the Internet, the protocol is called the **Internet protocol** (**IP**).

The IP provides a connectionless best-effort service that involves the formatting of the PDU to be transferred into **packets**, each of which has the unique network-wide address of both the source and destination end systems in the header of each packet/PDU. As with the link and physical layers below it, the network protocol – IP – is not concerned with the content of the packet being transferred but simply with how the total packet – header/PCI plus the PDU/data being transferred – is routed over the total network.

As we indicated in the introduction to this chapter, the Internet supports a wide range of applications. These include e-mail, Web access, telephony and so on. E-mail and Web access require a reliable connection-oriented service while for telephony a best-effort service is more appropriate. Clearly, therefore, since the IP only provides a best-effort service, an additional protocol/layer is needed to support both types of application. This is the role of the **transport layer**, which provides applications with a network-independent data interchange service.

To support both service types, there are two transport protocols in the transport layer: the **transmission control protocol** (**TCP**), which provides a reliable service, and the **user datagram protocol** (**UDP**), which provides a best-effort service. The PDUs relating to both protocols are all transferred over the network/Internet using the IP network layer protocol. In addition, since there are multiple applications, the transport layer is also responsible for directing the flow of application PDUs to the related application in the **application layer**.

The application layer provides the user, through a suitable interface, with access to the range of Internet applications. Associated with each application is a specific **application protocol** that provides the user with the corresponding service. The complete set of protocols is then called the **protocol stack** of the Internet or, more usually, the **TCP/IP protocol stack**. This is shown in diagrammatic form in Figure 1.38(b). As you can deduce from this, the name is derived from the two protocols that form the core of the Internet protocol stack.

Summary

A summary of the topics discussed in this chapter is given in Figure 1.39.

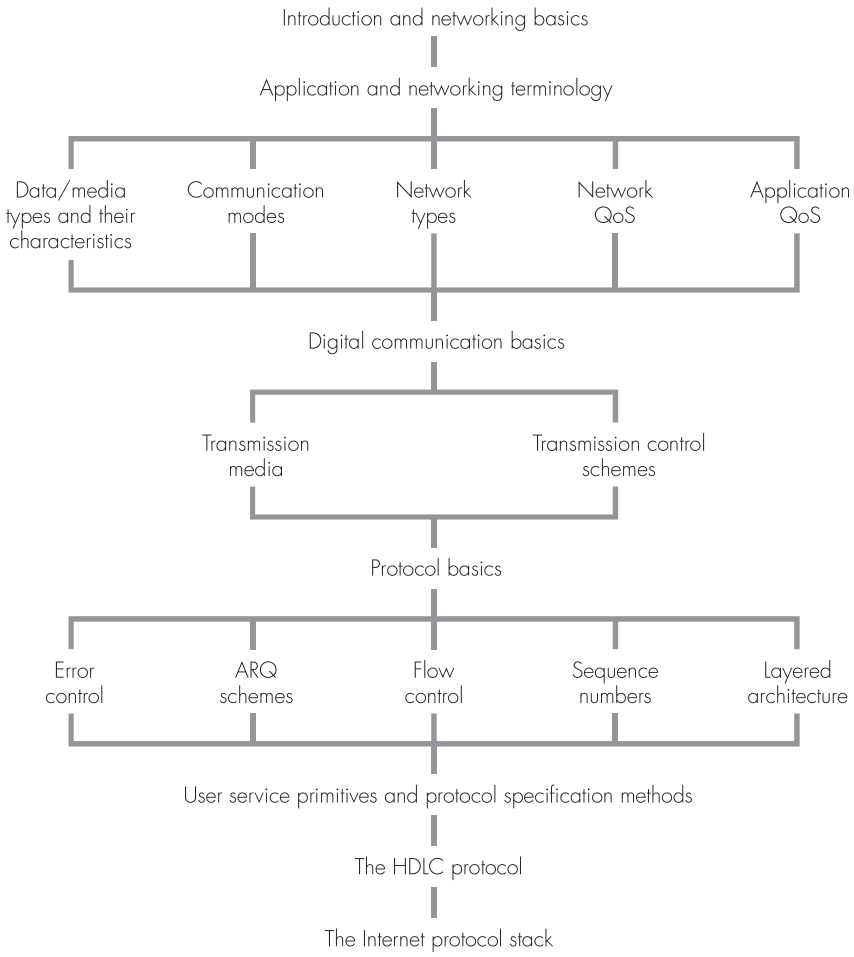


Figure 1.39 Introduction and networking basics summary.

Exercises

Section 1.2

1.1 State the basic form of representation of:
(i) text,
(ii) an image,
(iii) audio,
(iv) video.
State the form of representation that is used when all are integrated together and give your reason.

1.2 State the meaning of the term "bits per second" in relation to digitized audio and video.

1.3 What is the meaning of the term "compression" and why is compression used?

1.4 Describe the meaning of the following terms relating to the different types of data used in data communications:
(i) block-mode,
(ii) continuous,
(iii) streaming,
(iv) constant bit rate,
(v) variable bit rate.

1.5 With the aid of diagrams, explain the meaning of the following operational modes of a communication channel:
(i) simplex,
(ii) half-duplex,
(iii) duplex,
(iv) broadcast,
(v) multicast,
(vi) asymmetric and symmetric.

1.6 With the aid of a diagram explain the principle of operation of a circuit-mode network. Include in your explanation the need for signaling (messages) and the overheads of the connection setup delay associated with this.

1.7 With the aid of diagrams, explain the principle of operation of a connection-oriented packet-mode network. Include in your explanation the need for a virtual connection/circuit, a virtual circuit identifier, and a routing table.

1.8 In relation to a connectionless packet-mode network, explain the meaning of the following terms:
(i) best-effort service,
(ii) store-and-forward delay,
(iii) mean packet transfer delay,
(iv) jitter.

1.9 Identify and explain the meaning of the key QoS parameters associated with the following network types:
(i) circuit-switched,
(ii) packet-switched.

1.10 Define the BER probability of a transmission line/channel. How does this influence the maximum block size to be used with the line/channel?

1.11 Define the transmission delay of a transmission line/channel. First identify the individual sources of delay that contribute to this.

1.12 Identify and explain the meaning of the key parameters associated with application QoS.

1.13 With the aid of a diagram, explain the meaning of the following terms relating to packet-switched networks:
(i) packetization delay,
(ii) mean packet transfer delay,
(iii) jitter.
Hence describe how the effects on a constant bit rate stream of packetization delay and jitter can be overcome by buffering.

1.14 A Web page of 10 Mbytes is being retrieved from a Web server. Assuming negligible delays within the server and transmission network, quantify the time to transfer the page over the following types of access circuit:
(i) a PSTN modem operating at 56 kbps,
(ii) an aggregated ISDN basic rate access line of 128 kbps,
(iii) a primary rate ISDN access line of 1.5 Mbps,

(iv) a high-speed modem operating at 1.5 Mbps,
(v) a cable modem operating at 27 Mbps.

1.15 Discuss the term "application service classes". Include in your discussion how packets belonging to different classes are treated within the network.

Section 1.3

1.16 With the aid of the diagrams shown in Figure 1.8(a) and (b), explain the basic principles of the following transmission modes:
(i) baseband,
(ii) modulated.

1.17 With the aid of the waveform set shown in Figure 1.9, explain why the receiver samples the incoming signal as near to the center of each bit cell period as possible and, in some instances, bit errors occur.

Section 1.3.1

1.18 Explain why twisted-pair cable is preferred to non-twisted-pair cable. What are the added benefits of using shielded twisted-pair cable?

1.19 With the aid of diagrams, explain the differences between the following transmission modes used with optical fiber:
(i) multimode stepped index,
(ii) multimode graded index,
(iii) monomode,
(iv) wave-division multiplexing.

1.20 State the meaning of the following relating to satellite systems:
(i) microwave beam,
(ii) transponder,
(iii) geostationary.

1.21 With the aid of diagrams, explain the operation of a satellite system that is used in
(i) TV broadcast applications, and
(ii) data communication applications.

1.22 The maximum distance between two terrestrial microwave dishes, d, is given by the expression:

$$d = 7.14\sqrt{Kh}$$

where h is the height of the dishes above ground and K is a factor that allows for the curvature of the earth. Assuming $K = 4/3$, determine d for selected values of h.

1.23 Explain the terms "signal propagation delay" and "transmission delay". Assuming the velocity of propagation of an electrical signal is equal to the speed of light, determine the ratio of the signal propagation delay to the transmission delay, a, for the following types of data link and 1000 bits of data:
(i) 100 m of UTP wire and a transmission rate of 1 Mbps,
(ii) 2.5 km of coaxial cable and a transmission rate of 10 Mbps,
(iii) a satellite link and a transmission rate of 512 kbps.

1.24 With the aid of sketches, explain the effect on a transmitted binary signal of the following:
(i) attenuation,
(ii) limited bandwidth,
(iii) delay distortion,
(iv) line and system noise.

1.25 With the aid of a diagram and associated waveform set, explain the meaning of the term "adaptive NEXT canceler" and how such circuits can improve the data transmission rate of a line.

Section 1.3.2

1.26 Explain the difference between asynchronous and synchronous transmission.

Assuming asynchronous transmission, one start bit, two stop bits, one parity bit, and two bits per signaling element, derive the useful information transfer rate in bps for each of the following signaling (baud) rates:
(i) 300,
(ii) 600,
(iii) 1200,
(iv) 4800.

1.27 With the aid of a diagram, explain the clock (bit) and character synchronization methods

used with an asynchronous transmission control scheme. Use for example purposes a receiver clock rate ratio of ×1 and ×4 of the transmitter clock.

1.28 With the aid of the diagrams shown in Figure 1.17, explain how frame synchronization is achieved with an asynchronous transmission control scheme, assuming the data being transmitted is
(i) printable characters,
(ii) binary bytes.

1.29 With the aid of the diagrams shown in Figure 1.18 relating to bit/clock synchronization with synchronous transmission, explain how synchronization is achieved using
(i) clock encoding,
(ii) DPLL.

1.30 With the aid of the waveform sets shown in Figure 1.19, explain
(i) the Manchester and
(ii) the differential Manchester clock encoding methods.
Why do both methods yield balanced codes?

1.31 (i) Explain under what circumstances data encoding and a DPLL circuit may be used to achieve clock synchronization. Also, with the aid of a diagram, explain the operation of the DPLL circuit.
(ii) Assuming the receiver is initially out of synchronization, derive the minimum number of bit transitions required for a DPLL circuit to converge to the nominal bit center of a transmitted waveform. How may this be achieved in practice?

1.32 Assuming a synchronous transmission control scheme, explain how character and frame synchronization are achieved:
(i) with character-oriented transmission,
(ii) with bit-oriented transmission.

1.33 Explain what is meant by the term "data transparency" and how it may be achieved using:
(i) character stuffing,
(ii) zero bit insertion.

Section 1.4

1.34 With the aid of frame sequence diagrams and assuming an idle RQ error control procedure with explicit retransmission, describe the following:
(i) the factors influencing the minimum time delay between the transmission of two consecutive information frames,
(ii) how the loss of a corrupted information frame is overcome,
(iii) how the loss of a corrupted acknowledgment frame is overcome.

1.35 A series of information frames with a mean length of 100 bits is to be transmitted across the following data links using an idle RQ protocol. If the velocity of propagation of the links is 2×10^8 m s^{-1}, determine the link efficiency (utilization) for each type of link:
(i) a 10 km link with a BER of 10^{-4} and a data transmission rate of 9600 bps,
(ii) a 500 m link with a BER of 10^{-6} and a data transmission rate of 10 Mbps.

1.36 With the aid of frame sequence diagrams, describe the difference between an idle RQ and a continuous RQ error control procedure. For clarity, assume that no frames are corrupted during transmission.

1.37 With the aid of frame sequence diagrams, describe how the following are overcome with a selective repeat error control scheme:
(i) a corrupted information frame,
(ii) a corrupted ACK frame.

1.38 Repeat Exercise 1.37 for a go-back-N scheme.

1.39 Discriminate between the send window and receive window for a link and how they are related with:
(i) a selective repeat retransmission scheme,
(ii) a go-back-N control scheme.

1.40 With the aid of frame sequence diagrams, illustrate the effect of a send window flow control limit being reached. Assume a send window of 2 and a go-back-N error control procedure.

1.41 Assuming a send window of K, deduce the minimum range of sequence numbers (frame

identifiers) required with each of the following error control schemes:
(i) idle RQ,
(ii) selective repeat,
(iii) go-back-N.

Clearly identify the condition when the maximum number of identifiers is in use.

1.42 With the aid of Figure 1.31, explain the meaning of the following terms:
(i) layered architecture,
(ii) interlayer queues,
(iii) local event,
(iv) user services,
(v) used services.

1.43 Using the abbreviated names listed in Figure 1.32, show how the idle RQ primary can be specified in the form of:
(i) a state transition diagram,
(ii) an extended event–state table,
(iii) a high-level pseudocode program.

1.44 With the aid of a time sequence diagram, show a typical set of link layer service primitives, assuming the link layer provides
(i) a reliable service, and
(ii) a best-effort service.

1.45 In relation to the HDLC frame format shown in Figure 1.36, explain the meaning and use of the following terms:
(i) supervisory frames,
(ii) unnumbered frames,
(iii) poll/final bit,
(iv) command and response frames,
(v) extended control field bit definitions,
(vi) piggyback acknowledgment,
(vii) unnumbered information frame.

Section 1.5

1.46 With the aid of a diagram, explain the meaning of the following terms:
(i) protocol data unit,
(ii) protocol control information,
(iii) layering,
(iv) protocol stack.

1.47 In relation to the Internet protocol stack, with the aid of a diagram explain briefly the role of the following protocol layers:
(i) physical layer,
(ii) data link control layer,
(iii) network layer,
(iv) transport layer,
(v) application layer.