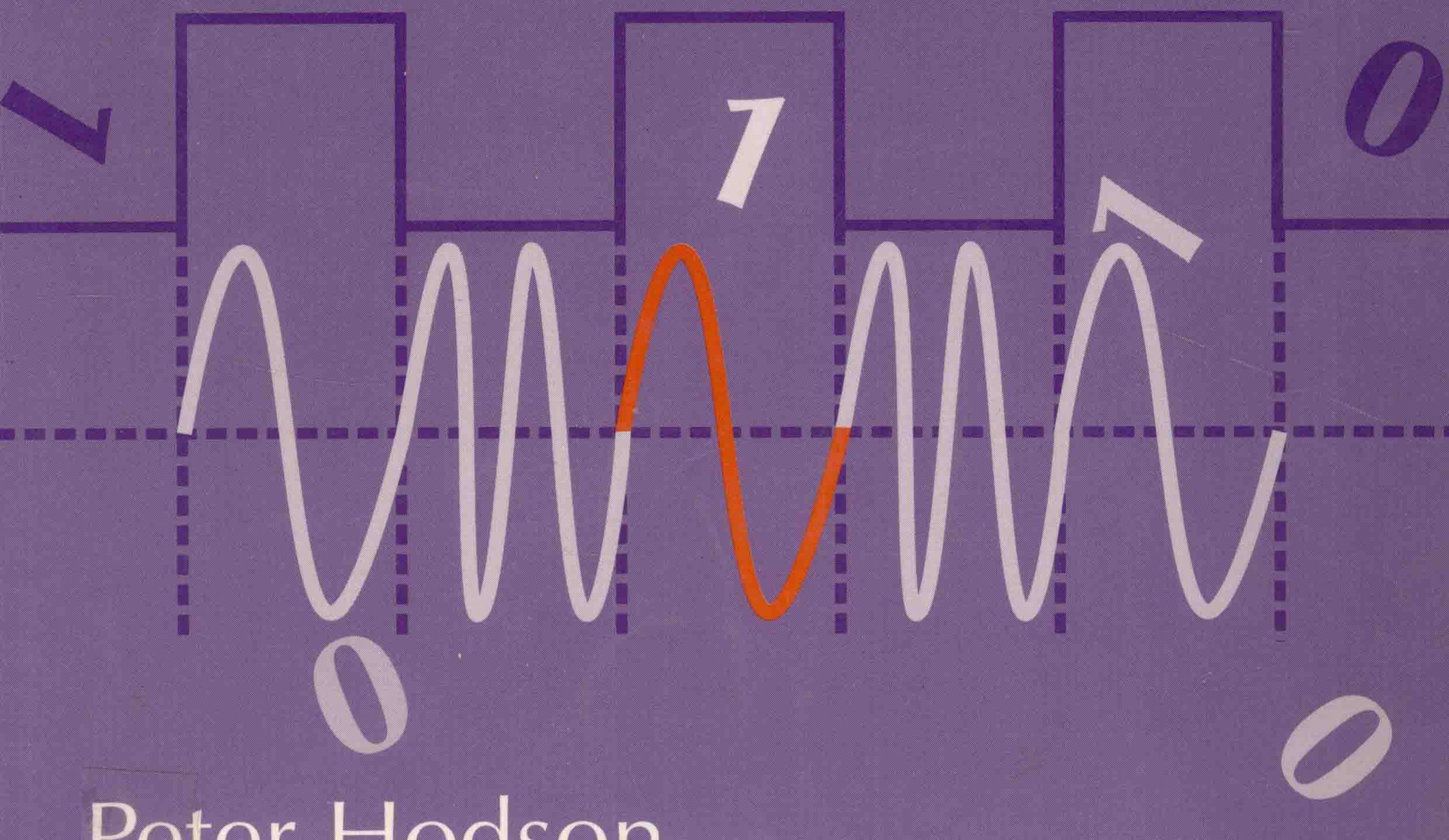
Local Area Networks

2nd edition

. . . . in a semester



Peter Hodson

Preface

The educational sector is experiencing significant changes in the way in which courses are delivered. Increasingly courses are being modularised to enable students from a wider range of educational backgrounds to study individual modules that were previously available only on particular programmes. Courses are also being run with significantly reduced lecturer contact time, and therefore texts are needed that are provocative and interactive.

Aim

The aim of this book is to provide the course text for the Networking element required by most degree and BTEC Higher National courses in Computing.

It has been developed by the author over a number of years whilst presenting courses to both students and practising professionals. The first chapters cover simple concepts and principles of communication that may be revision for some students. The rest of the book deals with issues relating to LANs and their interconnection and internetworking.

Approach

The book provides a framework for understanding key concepts in local area networks. Additionally, the more practical issues of cabling structures and interconnections of networks are discussed. By covering key concepts and practical issues, the text provides a blend of material based on traditional concepts of having a solid *technical* content coupled with a course covering *real issues of relevance in practice*. In this way the book can be used as

- a core text to *support* courses which are delivered through traditional lectures
- a text allowing *self-learning* on part, or the whole, of courses where lecturer contact-time is limited.

Important issues are repeated in several chapters, allowing individual chapters to be freestanding whilst reinforcing the learning process when read as a whole.

Questions

The author has developed an extensive range of questions which have been successfully tried on students over several years. Questions are divided into two main levels: short *in-text questions* and longer, *end of chapter questions*.

Short Questions

The short questions occur at frequent intervals in the text and encourage students to apply their understanding of the preceding text without halting them for too long in their progression through the topic. Answers to these questions are usually given on the following left-hand page.

Further questions

The end of chapter questions are graded in difficulty from simple tutorial questions to partial or full examination style questions. Approximately half these questions have answers at the end of the book (indicated with an asterisk): the others have answers in the Lecturers' Supplement only.

Lecturers' Supplement

A supplement is available to lecturers that contains suggested answers to those chapter-end questions without answers in the book (approximately half) and larger versions of about 50 diagrams in the book for use as OHP masters. It is available free to lecturers adopting the book as a course text – you will be required to give details of the course title, student numbers and book supplier – or for a charge of £3 to any lecturer applying on college headed paper.

Changes incorporated into the Second Edition

The second edition includes introductory chapters on basic data communications. These provide an introduction to the terminology and concepts used throughout the remainder of the text. The opportunity to include the latest developments in LAN technology and technically update the text has also been taken.

Peter Hodson
March 1995

Object-Oriented Programming
with C++

D Parsons

Level: HND/Degree **Series:** In a Semester

The book assumes a prior knowledge of the basic principles of programming.

All the example programs and solutions to exercises are designed to run on any C++ compiler.

It is known to be used on HNC/D, A Level Computing and BSC Computer Studies.

Review comments

'A very welcome book. The material and content are suitable for an undergraduate course and the price is suitable for a student's pocket.' 'An excellent, reasonably priced text.' 'Clearest explanations I've seen of these difficult concepts.' 'The explanation of difficult concepts using diagrams is superb.'
Lecturers

1st edition • 328 pp • Sept 1994 • ISBN 1 85805 089 8

Human Computer Interface

S Hill

Level: HND/Degree **Series:** In a Semester

This book provides a practical introduction for the Human Computer Interaction and Interface design elements of most BTEC Higher and Degree courses in Computing, BIT and Software Engineering.

Contents: Issues in HCI, Standards and the law, People, Technology, Designing interfaces, Evaluating interfaces (the measurement of usability), Designing support materials, Case study.

1st edition • 200 pp • June 1995 • ISBN 1 85805 119 3

Operating Systems

C Ritchie

Level: HND/Degree **Series:** In a Semester

It is known to be used on the following courses: BTEC Higher National, first and second year Computing degree courses, Software Engineering and Mechanical Engineering degrees. It is on the reading list of IDPM.

The new edition, apart from a general update, incorporates a significant amount of detail on PC Windowing systems, including newer technologies such as DDE, OLE and ODBC.

Review comments

'This is one of the most exciting, useful and lucid books on the subject I have seen.' 'Excellent – I wish I had had it when I was a student.' 'Very readable book with difficult concepts presented in a clear, understandable way.' Lecturers

2nd edition • 240 pp • Feb 1995 • ISBN 1 85805 131 2

Programming in Visual Basic

P K MacBride

Level: HND/Degree **Series:** Complete Course Text

The book is suitable for use with versions 2.0 and 3.0. It is known to be used on C & G, GNVQ Advanced, ALC, BTEC National.

Contents: Designing and creating programs, Program flow, Interacting with the user, Testing and debugging, Graphics, Procedures, functions and forms, Arrays, Sequential files, Records and random access files, MDI Forms.

Review comments

'Definitely an answer to prayer!' 'So much easier to learn from than a manual!' 'Excellent book at a good price.' 'A must for all students of Visual Basic.' 'An excellent book at an affordable price.' Lecturers

1st edition • 208 pp • Sept 1994 • ISBN 1 85805 092 8

Prolog Programming for Students

D H Callear

Level: HND/Degree **Series:** Complete Course Text

The book assumes access to any suitable Prolog or PD Prolog interpreter, such as LPA Prolog or PD Prolog.

It is known to be used on the following courses: HND, Access, BA (Business Studies), Bsc (Software Engineering), BSc (Computing) and MSc (Information Systems).

Contents: Basics, Variables, Facts, Semantic nets, Rules, Backtracking, Input, output and files, Expert systems, Recursion, Expert system shells, Lists in Prolog, Quantitative expert systems, Natural language, The cut, Route finding, Structures and structure predicates, Operator definitions, Prolog grammar notation.

Review comments

'Very easy to follow for complete beginners to Prolog.' 'Very clearly and concisely written.'

Lecturers

1st edition • 256 pp • Oct 1994 • ISBN 1 85805 093 6

Pascal Programming

B J Holmes

Level: HND/Degree **Series:** Complete Course Text

The book is known to be used on the following courses: BCS Part 1, BSc Computer Studies, HNC/D Computer Studies, HNC Software Design, NDI, HEFC Info Tech, C&G 726/223.

Review comments

'Excellent detailed book with plenty of examples.' 'Far better than most books that are twice the price.' 'A useful and well structured book that my students find easy to read and refer to.'

Lecturers

2nd edition • 464 pp • 1990 • ISBN 1 870941 65 9

Structured Programming in Cobol

B J Holmes

Level: HND/Degree **Series:** Complete Course Text

It is known to be used on the following courses: Computing degree, BTEC Higher National Computer Studies, C&G 424/425/726, IDPM, HND Commercial Systems, Programming Methodology. It is on the reading list of the ACAB.

Review comments

'The author has brought together in a practical way the JSP philosophy with all the traditional areas of COBOL.' 'Wins on price and coverage.' Lecturers

'A number of texts are mentioned, but the most popular seems to be B J Holmes' Structured Programming in COBOL.' Report on COBOL on BTEC Higher National Courses, Manchester Metropolitan University

2nd edition • 528 pp • 1991 • ISBN 1 870941 82 9

Modula-2 Programming

B J Holmes

Level: HND/Degree **Series:** Complete Course Text

It is known to be used on the following courses: HNC/D Computing, BSc Computer Science, HNC Software Engineering, BSc Engineering.

Review comments

'A good introductory book to outline basic principles of Modula-2 programming.' 'Contains the most important aspects of Modula-2 ... very affordable price.' Lecturers

2nd edition • 464 pp • 1994 • ISBN 1 85805 081 2

Programming with ANSI C

B J Holmes

Level: HND/Degree **Series:** Complete Course Text

This book comprehensively covers the official standard version (American National Standards Institute) of the programming language C and assumes the reader has no prior experience of other computer languages.

Contents: Programming environment, Data, Input and output, Instruction sequence, Selection, Repetition, Functions, Macros and mathematics, One-dimensional arrays, Arrays and structures, Pointers, Program development, Recursion, Sorting and searching, Files, Lists and trees, Further topics.

Note: All the programs have been written in ANSI C and as a result may be run in either MS-DOS or UNIX environments.

1st edition • 512 pp (approx) • June 1995 • ISBN 1 85805 117 7

Contents

1 Basics of Data Communication

1.1 Introduction

Effective use of computer systems demands that the user has the ability to move data between devices reliably. In a very simple example we must be able to print from a PC to an attached printer without any errors occurring as the data is transferred. This can be considered as a two stage activity. Firstly the data must be correctly sent and received. Secondly the sending device must know that it has been correctly received. If, during this second stage, the sender is informed that it has not been correctly received and the transfer was unsuccessful, then an attempt to put matters right and recover from the error is undertaken. The whole concept of data communication is based on these basic principles of sending data, checking its correct receipt and confirming how successful the transfer was. The designs which have evolved to manage this approach are known as protocols i.e. who says what and when!

At the simplest level there may be two devices that are directly connected. At more advanced levels they may be connected either by a telephone line or a full-blooded network and be separated by thousands of miles.

In this first chapter we will explore some of the basic elements of achieving data transfer.

1.2 Data Transfer and Asynchronous Transmission

To transfer data between two devices there are a few basic points to be examined. Let us consider the physical connection:

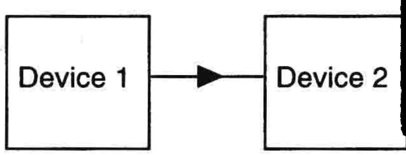


Figure 1.1 Simple Serial Connection

If this connection is a single strand of copper wire then exchange of data can only be by a serial sequence of bits flowing along the transmission media. Data can only flow in one direction at any instant in time along this single connection. For convenience the bits are frequently grouped together to represent a character, typically eight bits at a time (i.e. a letter of the alphabet, a number or a symbol such as a "!"). Each bit is usually represented on the copper wire as a voltage level. The normal arrangement is for a "zero" to be represented by a positive voltage and a "one" to be represented by a negative voltage. To ensure that each device is capable of sending to, or receiving from other devices, international standards have been established. This was intended to persuade designers not to create their own representations for their devices. This would have created havoc with possibly hundreds of different approaches and with little ability to connect devices which were supplied by different vendors. The interface standards, which include signalling conventions, that are most commonly found on devices are the

RS232c or V24 interface standards. The more recent X.21 standard has been in place for quite a while, but migration to this has been slow. Fortunately transition arrangements between X.21 and RS232c have been introduced to accommodate this slow migration.

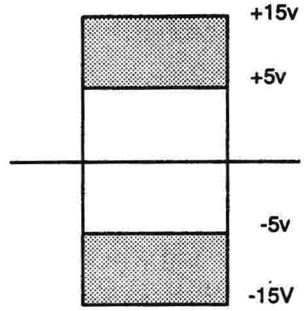


Figure 1.2 RS232c Voltage levels

Normally the voltage levels used in RS232c signalling are +12v or –12v, although voltage levels between +5v and +15v will be interpreted as a "zero" and equally a "one" will be interpreted by a signal in the –5v to –15v range. A voltage level in the shaded areas of figure 1.2 should be recognised as a signal representing a bit. Using this standard the bit stream 01010 may be represented by the signal levels in figure 1.3.

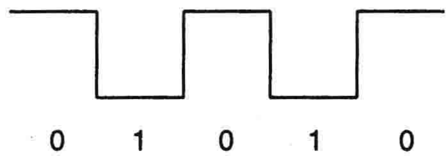


Figure 1.3 Typical bit stream.

Question 1.1

Draw the signal levels that would represent the bit stream 10100 using this standard.

Question 1.2

In answering question 1.1, were there any particular difficulties in representing the last two bits?

Each bit needs to be represented for a given length of time at the required voltage level on the copper wire. The length or duration of the signal has to be sufficient for the receiving end to recognise the incoming signal and record it accurately. For example, if each change of signal occurred after 10 ms, such that each bit was sent in this length of time, then in a 1 second period we could send 100 of these signals, i.e. we could send 100 bits in 1 second. This is usually stated as 100 bps or 100 bit/s (i.e. bits per second).

Question 1.3

If the data rate were 1600 bps, what would be the duration of each signal level?

It is evident that the faster you want to send data between two devices, the shorter the duration of each bit signalled on the transmission medium. A problem now arises at the receiving end of any data transfer. How does this receiver know when data is about to be sent and how will it know the rate at which the sending device will be transmitting? To recognise the incoming data and recover or record the data accurately, the receiver needs to read or sample the signal on the cable at a rate determined by the speed at which the data is being transmitted.

Let us first look at the situation where everything is idle i.e. the quiescent state. To alert the receiver that a character (eight bits) of data is about to be transmitted, a start bit is sent at the front of it. Conventionally, we also send a stop bit at the end of the character. These start and stop bits also act as delimiters on the character so that the boundary points of any character being transmitted will be understood by the receiver. Thus, the stop bit is an important check for the receiver that the end boundary point of the character frame has been correctly reached. Hence, a typical transfer of a character in asynchronous mode would look like:-

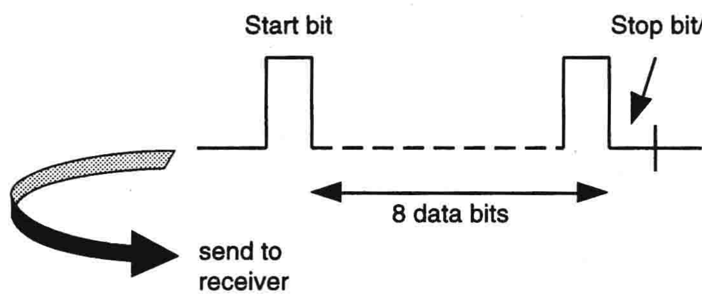


Figure 1.4 Asynchronous Character Representation.

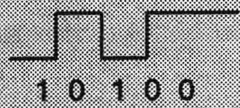
Question 1.4

Draw the asynchronous representation of the character frame for the eight bits 01011010.

Some systems may need more than one stop bit and a character will be terminated by 1.5 or 2 stop bits. This is usually for slower devices, typically with slow mechanical components, and is becoming less common. If we send two stop bits to a receiving device that only requires one, it won't matter since the second stop bit will just appear to be the start of an idle signal level.

The form of transmission discussed above is known as asynchronous transmission. The receiver doesn't know when the next character is going to arrive and awaits the arrival of the next start bit. There is no information within the character about the speed at which it is being sent, so there is no "timing" information (i.e. there is nothing to tell the receiver how quickly it needs to sample the incoming signal in order to recover the transmitted data). To clarify how important it is that the receiver should sample the transmission at the correct rate we will consider a simple example. A device is transmitting the character 01011010 at 100 bps (i.e. the duration or time cell of each bit is 10 ms) and the receiver tries to recover the data by sampling every 20 ms. Let's see what happens:

Answer 1.1



Answer 1.2

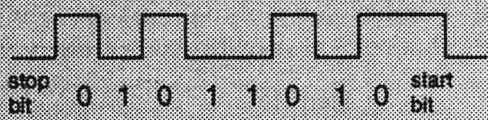
Perhaps the only difficulty is knowing where the boundary point between the two zeros is located, or how long to make the signal level. Of course if all binary values are drawn for a fixed length of time, this is not really a problem.

Answer 1.3

At 1600 bps each signal level or binary value lasts for 1/1600 sec or 625 µs.

Answer 1.4

The diagram below is drawn on a time axis. You have probably drawn the diagram in the reverse order, which is perfectly correct, as long as you know the order in which the bits are sent.



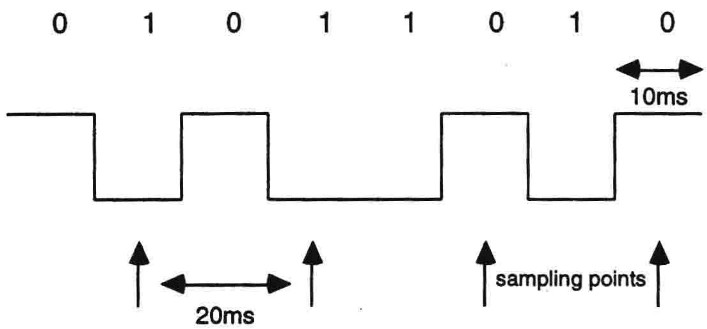


Figure 1.5 Sampling the transmission incorrectly

A receiver incorrectly sampling the transmission as shown in figure 1.5 would read the data sent as 1100. Clearly this is not what we want. The receiver needs to sample the line at the same rate as the sender is transmitting it. To achieve this, the transmission interfaces of each device have internal clocks each of which, within a reasonable level of accuracy, are both "clocking" at the same rate. Devices sending data to each other know the data rate between themselves because the data communications designer will have established the speeds at one of the standard rates. So the arrival of the start bit at the receiver starts its interface clock and instructs the receiver's interface to sample the incoming signal. Provided the sender and receiver clocks are running at the same rate, the data should be successfully received. We hope that the accuracy of the clocks is such that they maintain synchronisation and the receiver continues to clock near the mid-point of a time cell. If they are not accurate it is possible that the receiver clock will drift out of synchronisation and will try to sample at some point off the middle of the time cell and mis-read the signal.

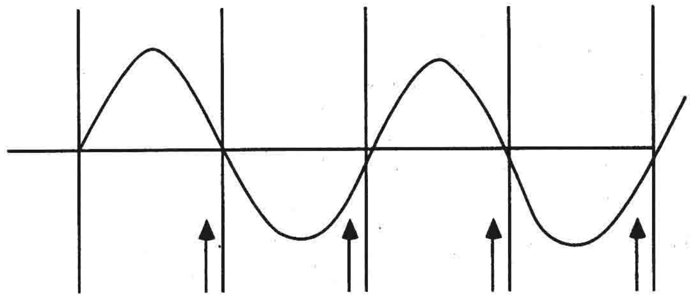


Figure 1.6 Delayed sampling

For relatively short lengths of data (in this case a character) the accuracy between the clocks is normally adequate enough for such drift not to be an issue. The faster the data rate and the greater the data length, the more acute this issue becomes. As a check on the protocol, the receiver insists on seeing a stop bit in the tenth (and possibly eleventh) bit position. If that isn't present, there has been an error. In a typical character transfer, successive characters will have a short gap between them. It is possible for this inter-character time gap to be nothing or very small, especially when the sender is transmitting at a maximum rate.

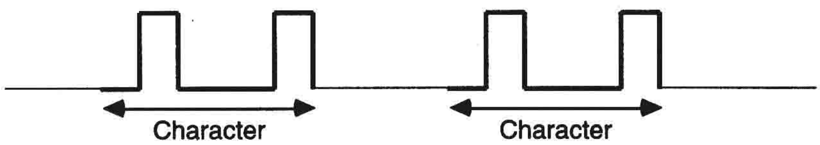


Figure 1.7 Character string

Consider a case where there is no "lost" time between each character sent, i.e. a start bit for one character follows immediately behind the stop bit of the previous character. If 200 characters are sent down a transmission line at a rate of 1200 bps we can now calculate the time it takes to transfer the data.

200 characters = (200 × 10) bits to transmit (assuming 1 stop bit and 1 start bit).

$$\text{Time to transmit} = \frac{\text{Number of bits sent}}{\text{Speed of transmission}}$$

$$\text{Time to transmit} = \frac{200 \times 10}{1200} \text{ seconds} = 1.66 \text{ seconds}$$

Question 1.5

If we assume that each character could be sent without a delay between them, how long would it take to asynchronously send 100 characters down a line at 2400 bps?