*Technical Report*

UCAM-CL-TR-52
ISSN 1476-2986

Number 52

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

# The design of a ring communication network

Steven Temple

January 1984

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

*http://www.cl.cam.ac.uk/*

This technical report is based on a dissertation submitted January 1984 by the author for the degree of Doctor of Philosophy to the University of Cambridge, Corpus Christi College.

Technical reports published by the University of Cambridge Computer Laboratory are freely available via the Internet:

*http://www.cl.cam.ac.uk/TechReports/*

Series editor: Markus Kuhn

ISSN 1476-2986

# THE DESIGN OF A RING COMMUNICATION NETWORK

Steven Temple

Corpus Christi College

A dissertation submitted for the degree of

Doctor of Philosophy in the University of Cambridge

January 1984

# SUMMARY

This dissertation describes the design of a high speed local area network. Local networks have been in use now for over a decade and there is a proliferation of different systems, experimental ones which are not widely used and commercial ones installed in hundreds of locations. For a new network design to be of interest from the research point of view it must have a feature or features which set it apart from existing networks and make it an improvement over existing systems. In the case of the network described, the research was started to produce a network which was considerably faster than current designs, but which retained a high degree of generality.

As the research progressed other features were considered, such as ways to reduce the cost of the network and the ability to carry data traffic of many different types. The emphasis on high speed is still present but other aspects were considered and are discussed in the dissertation. The network has been named the Cambridge Fast Ring and the network hardware is currently being implemented as an integrated circuit at the University of Cambridge Computer Laboratory.

The aim of the dissertation is to describe the background to the design and the decisions which were made during during the design process, as well as the design itself. The dissertation starts with a survey of the uses of local networks and examines some established networks in detail. It then proceeds by examining the characteristics of a current network installation to assess what is required of the network in that and similar applications. The major design considerations for a high speed network controller are then discussed and a design is presented. Finally, the design of computer interfaces and protocols for the network is discussed.

# PREFACE

I wish to thank my supervisor, Dr. Andy Hopper, for stimulating my interest in local networks and for guidance and many useful discussions during the course of my research. I should also like to thank Professor Roger Needham for his encouragement and for allowing me to spend time working in the USA.

I am indebted to the following people who have read and commented on various parts of this dissertation: Andy Harter, Jane Hewitt, Andy Hopper, Mike Muller, Robin Williamson and Neil Wiseman.

Throughout my research I have been supported by a grant from the Science and Engineering Research Council for which I am grateful.

Except where explicitly stated in the text, this dissertation is the result of my own work and is not the outcome of any work done in collaboration. Furthermore, this dissertation is not substantially the same as any that I have submitted for a degree, diploma or other qualification at any other university. No part of this dissertation has already been or is being concurrently submitted for any other degree, diploma or other qualification.

# CONTENTS

# GLOSSARY

The following abbreviations are frequently used in this thesis. Where appropriate, they are explained in greater detail when they first appear in the text.

| | |
|---|---|
| **AFC** | Address Filtering Chip |
| **BSP** | Byte Stream Protocol |
| **B/W** | Bandwidth |
| **CFR** | Cambridge Fast Ring |
| **CFR/1** | Cambridge Fast Ring, Mark 1 |
| **CMDS** | Cambridge Model Distributed System |
| **CMOS** | Complementary Metal Oxide Semiconductor |
| **CR** | Cambridge Ring |
| **CRC** | Cyclic Redundancy Checksum |
| **CSMA/CD** | Carrier Sense Multiple Access with Collision Detection |
| **DMA** | Direct Memory Access |
| **ECL** | Emitter Coupled Logic |
| **FIFO** | First In, First Out memory |
| **IEEE** | Institute of Electrical and Electronic Engineers |
| **LAN** | Local Area Network |
| **LSI** | Large Scale Integration |
| **MOS** | Metal Oxide Semiconductor |
| **MSI** | Medium Scale Integration |
| **NCC** | Network Controller Chip |
| **NIP** | Network Interface Processor |
| **ppBW** | Point-to-Point Bandwidth |
| **PROM** | Programmable Read Only Memory |
| **PUP** | PARC Universal Packet |
| **RAM** | Random Access (read/write) Memory |
| **ROM** | Read Only Memory |
| **SSP** | Single Shot Protocol |
| **sysBW** | System Bandwidth |
| **TTL** | Transistor Transistor Logic |
| **ULA** | Uncommitted Logic Array |

**Chapter 1**

# INTRODUCTION

Practical research into the communication systems now known as Local Area Networks began in the late 1960's. The initial aim of the research was to provide a means of resource sharing in multi-processor environments. Local area networks evolved from larger packet switching networks such as the ARPANET [Roberts 73] and have the following characteristics:-

1) Connection of digital devices over a small geographical area
   - typically less than a few square miles

2) High data rate and low bit error rate
   - over 1MBit/S and better than 1 in $10^9$, respectively

3) Cheap transmission media such as twisted pair or coaxial cable

4) Simple addressing and routing of data formed into packets

Local area networks, also known as local networks or LANs, enable large numbers of digital devices (usually computers) to be connected together. The devices are able to communicate with each other at high data rates and with little delay. The area covered might be anything from a single room to a university campus stretching over several miles.

## 1.1 THESIS AIMS

The aim of this thesis is to describe the design of a new high speed local network, the Cambridge Fast Ring, which is being developed at the Computer Laboratory in the University of Cambridge. Having decided to undertake such a project, there are two approaches that may be adopted. One is to take an existing design and re-work it to function faster, the other is to start an entirely new design. The former method is likely to be easier and faster in implementation, whilst the second involves a great deal of research, but gives the opportunity to improve on current designs.

The design which is presented in this thesis falls between these two approaches. The new network borrows heavily from the Cambridge Ring, also developed in the Computer Laboratory, but has many new features added as a result of experience with

the Cambridge Ring and other networks. The thesis attempts to describe the ideas which resulted in the new design and the ways in which the design can be exploited in the construction of computer networks.

### 1.1.1 Outline

The remainder of this chapter describes some of the uses to which local networks are currently put and speculates on possible uses for faster networks. Chapter 2 gives an outline of several local networks developed over the last decade and is intended to give some idea of the way these designs have evolved as a result of advances in technology and experience with earlier designs. Chapter 3 describes a particular implementation of a local network and the environment in which it is used. Some measurements of traffic patterns and network load made on this network are given.

Chapter 4 presents some of the major issues which were considered in the design of the Cambridge Fast Ring and in Chapter 5 an early version of this system is described. Problems with the implementation of this version gave the opportunity for a redesign and Chapter 6 describes the Cambridge Fast Ring in its present form.

Chapter 7 discusses the use of bridges. A bridge is a means of connecting Cambridge Fast Rings together to make more complex networks. Chapters 8 and 9 consider the design of protocols and network interfaces which are necessary to make a computer network from the Cambridge Fast Ring. Finally, Chapter 10 summarises the work done and gives some ideas for further, related work.

### 1.1.2 Acknowledgement

The designs for the Cambridge Fast Ring, presented in Chapters 5 and 6, are the result of the joint efforts of A. Hopper, R.M. Needham and the author. The author is responsible for all other discussion, for the study in Chapter 3 and for the designs described in Chapters 7, 8 and 9.

## 1.2 USES OF LOCAL AREA NETWORKS

Many applications have been found for local area networks and they are widely used in computer systems in commercial and academic environments. In industry they are used for process control and, more recently, have been applied in office automation applications.

### 1.2.1 Resource Sharing

The first local networks were designed with resource sharing in mind and today this is still one of their major uses. Traditional time shared computer systems consist of a processor unit, a printer, a disc store, a magnetic tape unit and a number of terminals. Many users may use the processor and peripherals on a time slicing basis. In order to use a second computer nearby, a user must get up and go to another terminal attached to that computer. This second computer will require a similar complement of peripherals and it is desirable to be able to share peripherals between computers, not least for economic reasons. Past methods of sharing one set of peripherals between many computers have been complex and costly, with little scope for expansion.

Connecting the two computers and a single set of peripherals to a local network is an effective strategy and has a number of advantages. Since each user's terminal is connected to the network, it may communicate with both processors. Also, only one printer and tape unit are required, assuming that they can cope with the load imposed on them, and the disc stores may be shared by all users. Functions such as an electronic mail mechanism common to all users of the systems are now possible. However, the security of each system is now possibly reduced and some means of authenticating network operations may be required.

The last ten years have seen a substantial fall in the cost of semiconductor products, particularly microprocessors and memories. The price of mechanical peripherals has remained much the same in absolute terms and risen considerably relative to semiconductor based products. The fall in processor prices has led to the emergence of the personal computer, a single user computer with a processing power comparable to the normal "slice" of a time shared system. There is a trend towards using these personal computers, which may incorporate a visual display unit and keyboard and yet be no larger than a conventional computer terminal.

The relative cost of peripherals is high and it is sensible to share them using a local network. The personal machines may have a small disc store but even so a large, reliable file store on the network is very useful for dependable storage and shared access to files. This file store is commonly called a **file server** and is made up of a number of disc stores, controlled and connected to the network by a dedicated computer. There may be several of these file servers on the network and more can be added should the need arise. Simpler peripherals, such as a line printer, require little

processing power in the computer which connects them to the network and a microprocessor is normally sufficient.

The cost of attaching devices to a network may well be a factor in deciding whether to employ one. If connection costs are high then the benefits must also be high to justify the purchase. The cost will be split between cabling, the network interface hardware and the software needed to enable communications. The cost of cabling will vary widely according to the installation and software costs are also difficult to assess. Current hardware costs are around one thousand pounds per connection for Ethernet or Cambridge Ring networks [Lee-R 83].

The cost of network hardware will fall when the necessary functions can be implemented by dedicated integrated circuits. Costs should then fall by a factor of ten or more for that particular part of the network. Software costs will remain the same unless the network hardware reduces the amount or complexity of software.

### 1.2.2 Distributed Computer Systems

The falling cost of processors and memory means that the CPU cycles of a mainframe can be obtained from a number of microcomputers at significantly lower cost. By connecting many of these computers to a network and giving one to each user, we achieve the power of the mainframe but with some additional advantages (and disadvantages). Reliability will be increased since failure of one of the processors is unlikely to affect the operation of the others. Failure of the network or some other essential part of the system is still a potential problem. However, systems in use for some years have shown the reliability of the local networks employed to be quite adequate for resource sharing and distributed computing applications [Binns 82].

The performance of a computer in this environment will be independent of the number of users of the system provided any shared resources are not being heavily loaded. Some mainframe applications, such as dedicated use for numerical calculation, may not be feasible because it is not possible to split the problem up for execution by many smaller processors.

There are a number of possible implementations of a distributed system. The Cambridge Model Distributed System (CMDS) is a working experimental system which presently supports over 20 users [Needham 82]. The CMDS approach is to connect to the network a collection of processors, not necessarily all of the same type, and

allocate these to users who request them. Users have a terminal in their office which is connected to the network and so to the processors. The processors are kept together elsewhere, which has the advantage that maintenance is easy and the noise and heat that they produce can be suppressed more effectively than if they were in the user's office.

A user may request one or more processors of given types or ability. Separate processors are required to administer the resources of the network. The processors which perform these network administration functions are small dedicated microcomputers known as **servers**. There are also file servers and printing servers for storing data and providing hard copy output.

The system is able to provide all of the facilities of a time shared system, such as electronic mail, tape archiving and the sharing of programs and data, whilst being rather more flexible in terms of expansion and accommodating user's needs. Failure of one of the processors will not affect overall operation and failure of a server can be remedied by keeping spares on the network and relocating the server function in a working spare. This may take several minutes to do but is considerably faster than repairing or replacing a faulty part of a mainframe. In practice the simplicity of the processors and servers makes them highly reliable. The CMDS has been highly successful and currently has over 30 processors, five times more than at its inception. It seems possible that such systems may replace mainframe computers in certain applications.

### 1.2.3 Office Automation

Many of the traditional office activities, such as typing letters and documents, filing and accounting, are presently being performed by computers. Local networks can be used to advantage in this environment in many ways. A high quality printer can be shared by many word processing systems, internal memos can be sent using electronic mail and communication of information between departments is made easy. An interface to a public wide area network allows mail to be quickly sent all over the world from a person's desk.

Most large offices have an internal telephone system controlled by a Private Automatic Branch Exchange (PABX). This connects internal telephones and external lines, routing calls as requested. The number of lines may range from less than ten to over a thousand and typically a maximum of 10% of them may be in use at once. Local

networks generally have sufficient bandwidth to carry digitised telephone conversations and the functions of the PABX may be performed by a local network with a computer to coordinate operations. With voice input and output for computers probable in the next few years, this integration of function will be a natural choice. The SILK local network, manufactured by Hasler, is an office local network which integrates voice and data traffic [Jackson 81].

Digitised voice is commonly sent at a data rate of 64KBit/S. Current local networks have overall bandwidths of around 4MBit/S and so 30 two-way conversations would saturate such systems. If some other forms of traffic are required on the network, then either the network must be made to go faster or the number of conversations must be reduced. The nature of the other traffic may be such that it interferes with the voice traffic, for example by delaying it. In this case a means of segregating the two traffic types so that they do not interfere with each other is desirable. Alternatively having sufficient bandwidth on the network to cope with all demands will be satisfactory.

### 1.2.4 Other Applications

Local networks are used to advantage in many process control applications. An industrial site may range in size from 100m to 10km and communication is usually necessary between devices all over the site. Local networks are a cost effective way of integrating the many wiring paths which are required. The requirements of this application are rather different to previous examples, data rates tend to be lower but delay is often critical and reliability crucial. Failure to turn on a valve or to coordinate two related processes may be disastrous. The ability of local networks to combine the functions of remote control, site-wide communication and statistical and other data gathering operations, onto a single wire, makes them highly suited to this work. Some local networks can use fibre-optic links and these can be of use in electrically noisy environments.

An example of local network use in an industrial application is in controlling robots on a production line. The robots need complex sequences of commands and the actions of all the robots must be carefully coordinated. A local network is used to connect the robots to their controlling computer and this greatly reduces the wiring costs involved.

On a much smaller scale an Ethernet local network, implemented as a single integrated circuit, has been used to connect circuit boards together inside a photocopier. The low cost of the chip means that this approach is cheaper and more reliable than routing a loom of wires between the various boards.

### 1.2.5 Applications for a Faster Network

The ability of computers to present data in graphical form is used to advantage in many applications. Many computing environments could employ a network which could carry a mixture of voice, data and real-time graphical information. Many personal computers have high resolution, bit-mapped screens and performing rapid updates to these across a network requires a large bandwidth. A very fast network could support a small number of true video transmissions or a larger number of less demanding real-time image transfers such as run-length encoded pictures. A monochrome video signal can be transmitted in digital form at a rate of 9.6MBit/S and with data compression this might reduce to less than half. Networks to support video data will need to allow data to be transmitted at these rates and ideally the overall capacity should be such that several video transmissions may occur and still leave room for other data transmissions.

A number of present applications are limited by network bandwidth, particularly bulk data transfers from fast storage devices such as disc stores. Discs are now available with transfer rates of over 10MBit/S and implementation of a virtual storage system over a local network with discs such as these could not be contemplated with currently available networks.

Faster networks will allow existing applications to grow in size and speed and will also generate new applications. Parallel processing, in which computations are split up and assigned to many processors running simultaneously, demands very rapid transfers of data and results between processors. These systems presently use circuit switching techniques for routing data between processors, a high speed local network could equally well be employed.

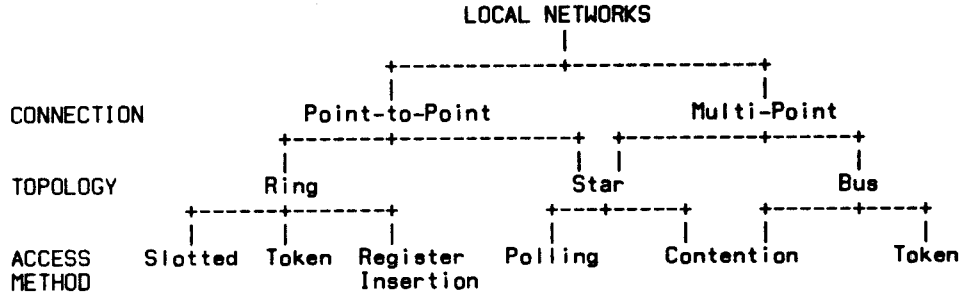
Many present network applications are limited in speed not by the network, but by the software which drives it. Protocols are necessary to enforce an orderly flow of data between machines on the network and these protocols are generally implemented as computer programs. The speed at which these programs run and the rate at which they can move data to and from the network are the limiting factors in many network

applications. Faster networks will only be useful if the protocols and data transfers can also be speeded up. This may be achieved by simplifying the protocol where possible and by speeding up its execution. For very fast networks the latter may have to be done by implementation in hardware.

Chapter 2

# LOCAL AREA NETWORKS

The first LAN to be constructed was probably that of Farmer and Newhall [Farmer 69] and since then over 100 LANs have been documented. When comparing LANs there are three main parameters to consider. First, the topology of the network is significant. To date most LANs have used simple topologies such as a ring or a tree. The second parameter is the protocol by which devices attached to the network gain access to it and communicate with each other. Access must be arbitrated since several devices may wish to use the transmission medium simultaneously. The final parameter is the transmission medium itself. This will have a characteristic speed and error rate and may be a fibre optic system or one of several wire systems. Some distinctive modulation and clocking scheme will be employed. The following diagram illustrates the way in which some common types of local networks are related to each other.




A multi-point connection scheme is one in which the signals sent by one station are seen by all others as they are transmitted. A station is the hardware which connects a device to the network and allows it to transmit and receive data. In a point-to-point system the signals are passed from station to station and only one station sees each bit at a given time. At present fibre-optic cables can only be used in point-to-point systems. Some of the topologies used for local networks are illustrated in Figure 2.0a.

## 2.1 PIERCE LOOP

Pierce postulated a loop network based on the slotted ring principle of operation [Pierce 72]. In this scheme a number of fixed length frames or **slots** (collections of bits) circulate round a loop of wire, the signals being regenerated at intervals. The

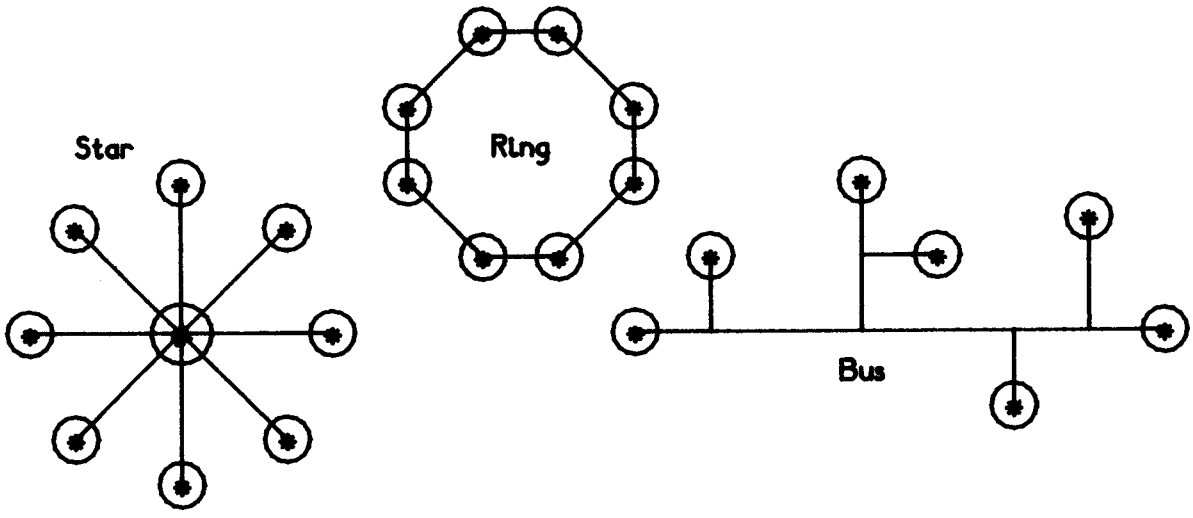



*Figure 2.0a Local network topologies*

slots are stored in the delay in the wire and in the nodes which are attached to the loop. The number of slots is thus determined by the length of wire in the loop and by the number of nodes. Each slot has a bit near the front to indicate whether it is full (carrying data) or empty. Devices connected to the loop are assigned addresses which are hard wired in the node and the slot contains two fields for addresses, one for the sender of the data (the source) and one for the recipient (the destination). The remainder of the slot is available to carry a message (data) between the source and destination.

Pierce envisaged a three level hierarchy of such loops, with a single national loop serving an entire country and connected to many regional loops. Each regional loop was then connected to many local loops, each of which served a small area of a few square miles. Such a system was never realised but a single loop was constructed by Kropfl [Kropfl 72] and subsequently used by Croker [Croker 72]. Only the local loops conform to our present notion of a local network. The format of the Pierce slot is shown in Figure 2.1a.

The sync field ensures that loop interfaces align to the bit sequence of the packet correctly and the control field contains the full marker as well as some other administrative information. When used in the hierarchical arrangement further addressing information is placed in the slot after the local addresses.

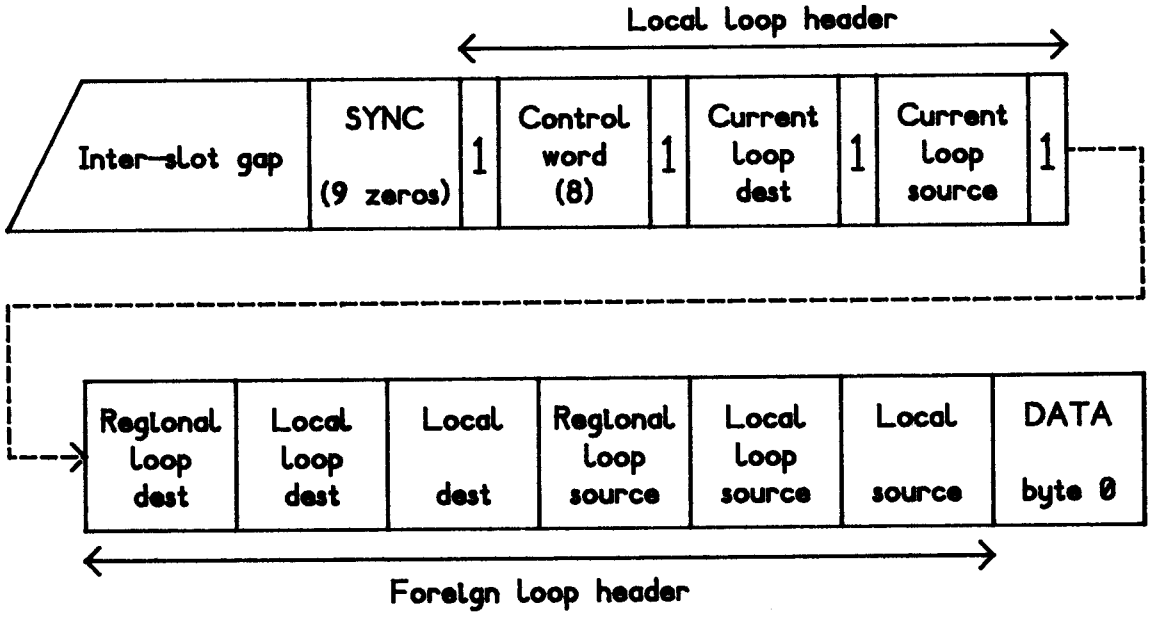



*Figure 2.1a Format of a slot on the Pierce loop*

A Pierce loop has three types of "station" attached to it. Each loop requires a single **A-station** to initialise and maintain the slot structure. This station also removes full slots which were not claimed by their destination. Host devices, such as computers, interface to the loop via **B-stations** and finally, **C-stations** connect local to regional and regional to national loops. No C-stations were ever implemented. The arrangement of local, regional and national loops is illustrated in Figure 2.1b.

When a B-station has some data to transmit it waits for an empty slot to pass and places the data and destination address in the appropriate fields of the slot. The source address is wired into the B-station and is inserted automatically. The slot then travels around the loop being relayed and observed at each B-station. When a B-station recognises the destination address as its own and has sufficient buffer space, it copies the data and marks the slot empty. Each B-station must therefore buffer the slot at least as far as the destination address in order to be able to mark it empty. This means that in small loops most of the bits on the loop will be stored in the B-stations rather than on the wire.

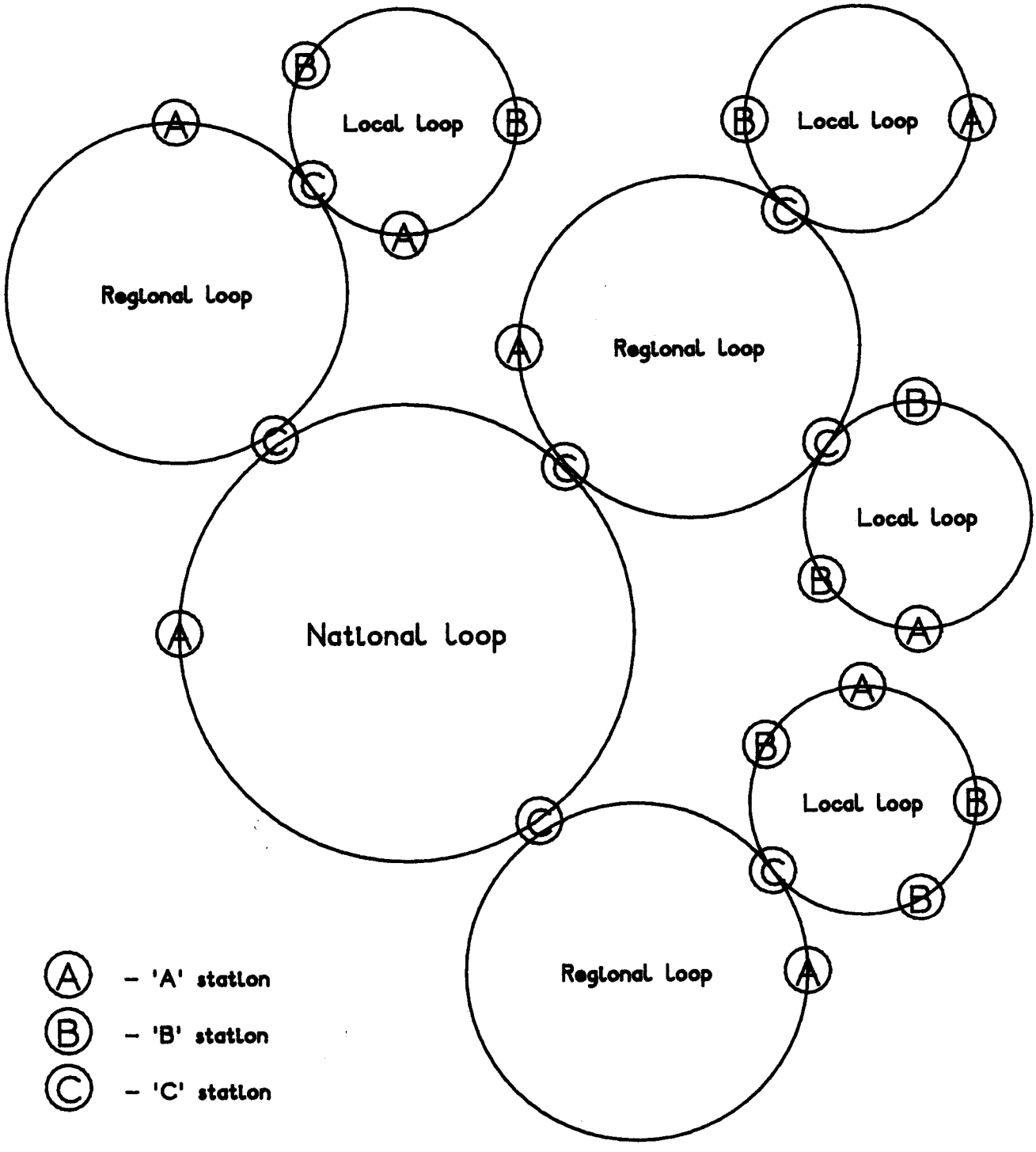



*Figure 2.1b* ***Local, regional and national Pierce loops***

When a full slot passes the A-station it is marked as such using another (monitor) bit in the control field. If a full slot enters the A-station with the monitor bit marked then this must be the second time it has done so and this implies that no B-station could receive the slot. This may have been because the address was invalid or the destination could not buffer the message. The A-station swaps the address fields of such slots so that the message is sent back to its source to indicate that something was wrong. The slot is also marked to show that this has occurred so that if the slot is not now received then the A-station will mark it empty when it next passes. This prevents full slots circulating indefinitely.

The Kropfl implementation of the Pierce loop used the Bell T1 modulation scheme for the data on the loop and operated with a clocking speed of 1.544MHz. A characteristic of this scheme is that each byte of data sent is accompanied by an extra bit (a one) to ensure that clock recovery can occur. The slots were 522 bits long and contained 432 bits (54 bytes) of user data. The addresses used in the loop were 8 bits long thus allowing up to 256 B-stations per loop. Pierce's addressing scheme for regional and national loops allowed 8 bits each for local and regional loop numbers thus allowing a maximum of $2^{24}$ hosts on the network.

In the very long loops which Pierce proposed, the slotted approach had significant advantages over other network architectures in that many messages could be in flight simultaneously. Most LANs only allow one message to be in transit at once, which can result in a significant loss of bandwidth in larger configurations. Another significant point about loops in general is that they are vulnerable to cable breaks and failure of any circuitry in the loop signal path. Pierce suggested making duplicate loops with automatic switching of stations to the other loop when the primary loop failed. Slotted loops are also dependent on the correct functioning of the A-station, Pierce had no suggestions of what to do in the case of A-station malfunction.

## 2.2 ETHERNET

The Ethernet, developed at Xerox PARC by Boggs and Metcalfe [Metcalfe 76] is a LAN based on rather different principles to the Pierce loop. Its topology is that of a branching non-rooted tree and data is transmitted in packets of variable length. The transmission medium is a coaxial cable which, unlike a loop, is passive. This means that there are no signals present on the cable when the network is idle. There is no requirement for a network controller such as Pierce's A-station and access to the

network is controlled by detecting the presence of someone else's message on the cable and delaying transmission until the network is idle. Hosts connect to the network via an Ethernet **interface** which incorporates a **transceiver** which "taps" the cable. Data is sent bit serially from the interface via the transceiver and so onto the cable. Packets sent in this way are broadcast all over the cable and heard by all active interfaces. Destination and source addresses (8 bits each) are present at the front of each packet and generally only one interface will recognise the destination address and receive the packet. The packet format is depicted in Figure 2.2a and an Ethernet configuration is shown in Figure 2.2b.

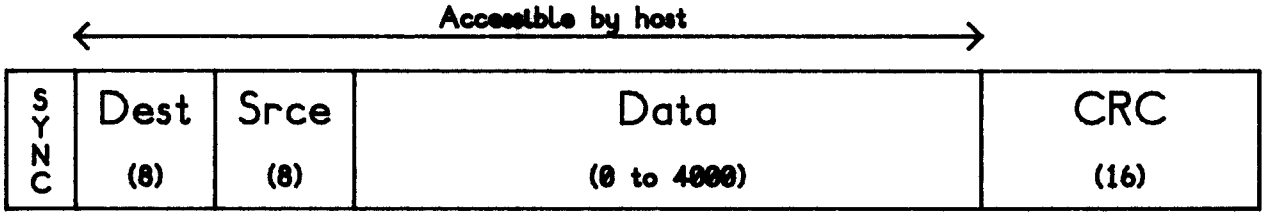



*Figure 2.2a Format of an Ethernet packet*

If an interface wishes to transmit, it first monitors the cable to see if anyone else is transmitting and if so waits until the transmission ceases. If no one else appears to be using the cable then the interface transmits and also listens to detect if some other interface started transmitting and thereby generated a collision on the cable. If a collision occurs both interfaces will detect it and immediately stop transmitting. Both will retransmit after an interval determined by observing the level of traffic prevailing and adding a random component. The interval is proportional to the frequency of collisions and this tends to reduce the load on the network when it is being heavily used.

This access method is called Carrier Sense Multiple Access with Collision Detection (CSMA/CD). The time during which a collision is possible is small in relation to the packet size, being the propagation delay of signals between the two furthest spaced interfaces. Thus if the first few bits of a packet are transmitted successfully then the whole packet will be.

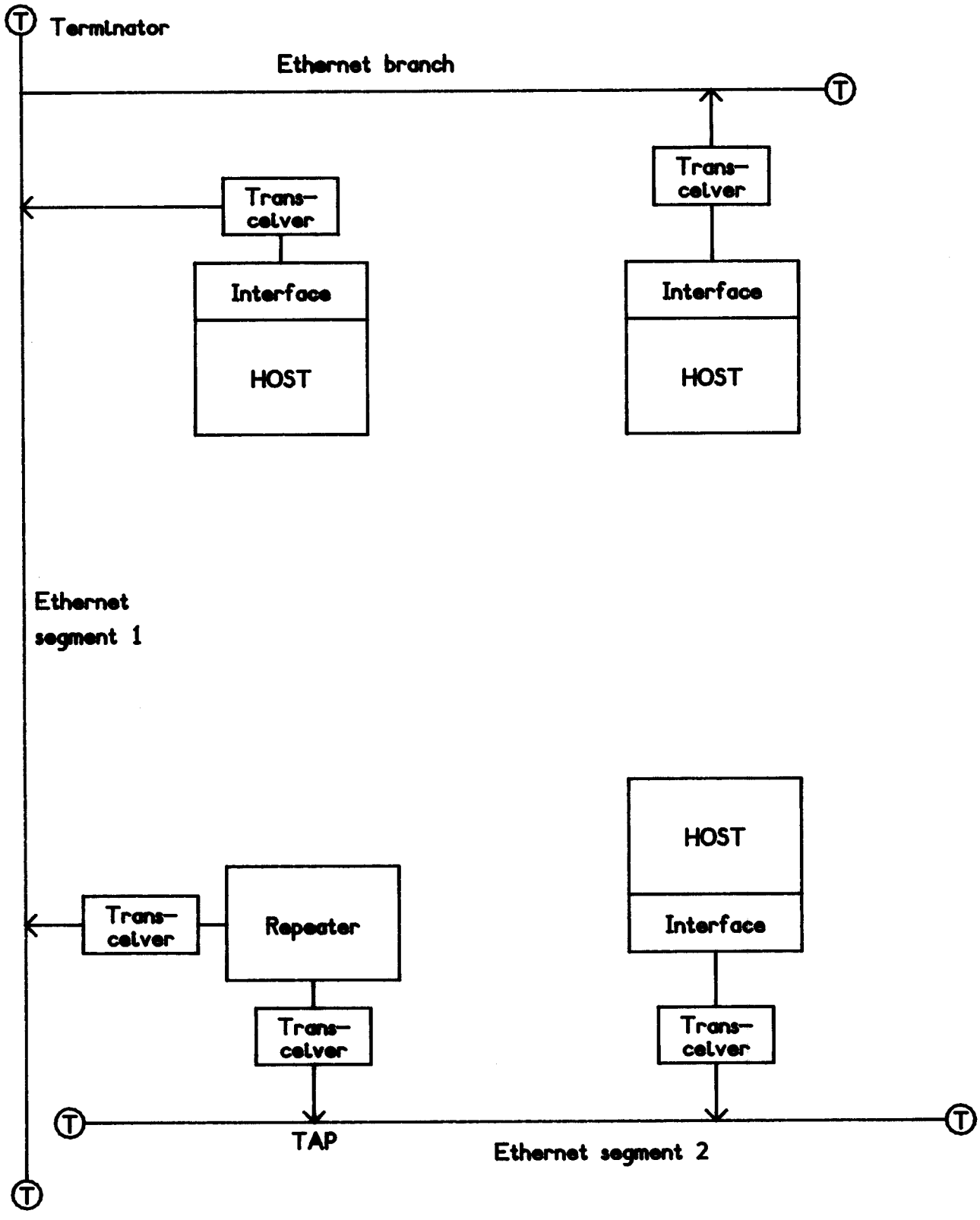



*Figure 2.2b A typical Ethernet configuration*

The original Ethernet supported up to 256 interfaces on a network with a maximum span of 1km. It was clocked at 3MHz and packets could be up to 4000 bits long. This limit on the maximum length prevents one host hogging the network. The access method is such that it is not possible to guarantee access to the network in any finite time. A probability of access can be calculated, however, and in normal operation no problems arise because of this. It does mean that the use of Ethernets is restricted in certain real-time applications.

In common with many LANs the Ethernet has a broadcast facility. Using this a single packet can be transmitted which may be received by all hosts on the network. The Ethernet does this by having a special destination address for the purpose and the facility has been used to advantage in some of the higher level Ethernet protocols.

Unlike a loop, an Ethernet cannot be extended indefinitely since each transceiver must be capable of driving the entire network, rather than just the next loop link. Ethernets may be extended in terms of size by means of packet repeaters which link two Ethernet segments and amplify signals passing through them in either direction. This makes the network active however and reduces the performance somewhat. Extending the addressing range to accommodate more hosts can be accomplished in much the same manner as proposed by Pierce and this has been successfully done for the Ethernet by means of packet gateways and a suitable addressing scheme [Boggs 80].

Recently a new version of Ethernet, supported by a number of computer manufacturers, has emerged. This runs at 10MHz and has 48 bit addresses which are thought to be enough for world-wide use. This is an attempt to bring Ethernet in line with a proposed standard for local area networks.

## 2.3 CAMBRIDGE RING

The Cambridge Ring (CR) is a slotted ring similar to the Pierce loop. It was developed by Wilkes, Wheeler and Hopper at the Computer Laboratory in Cambridge in the mid 1970's [Wilkes 79]. The slots of the CR are called **minipackets** and are somewhat smaller than either the Ethernet packets or Pierce's slots. The minipacket is 38 bits long and in addition to control bits contains two 8 bit address fields and 16 bits of data as shown in Figure 2.3a.

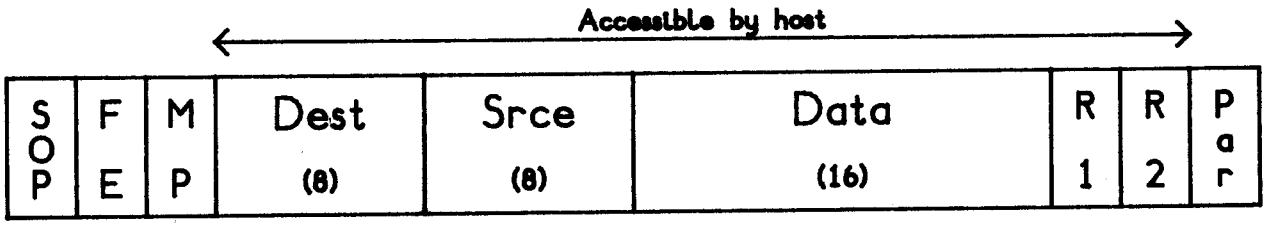



*Figure 2.3a Format of a Cambridge Ring minipacket*

The minipackets form a head to tail train on the ring with the excess length of the ring containing a **gap** of zero bits. The CR has a **monitor station** which is comparable in function to Pierce's A-station and hosts interface to the ring via **stations** and **repeaters**. Repeaters regenerate the signals on the ring and also provide a serial interface to a station (Figure 2.3b).

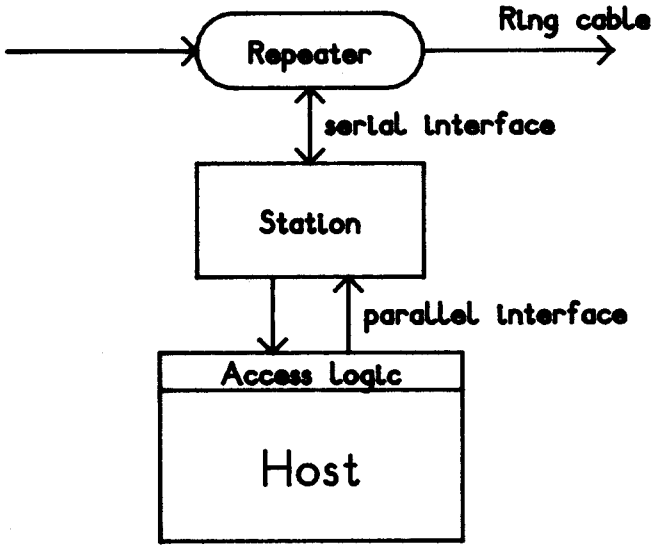



*Figure 2.3b Connection of a host to the Cambridge Ring*

The transmission medium used on the CR is a dual twisted pair and a phase modulation scheme is used. This allows an overall clocking rate of 10MHz with each twisted pair carrying data at half of this speed. Fibre-optic links have also been used successfully in a Cambridge Ring.

Stations contain minipacket buffers and control logic for interfacing to a host computer. Like the Pierce slot, each CR minipacket has a full/empty bit and a bit to indicate whether it has passed the monitor station while full. The minipacket also has control bits at the rear, two of which (the response bits) are marked by the destination station. When a minipacket is transmitted, it travels round the ring to its destination where it may or may not be received. Unlike the Pierce loop the slot is not marked empty at this point, but continues round the ring back to the source where it is marked empty. This has two major effects. First, hogging of the ring is prevented, since the transmitter is obliged to wait for the return of a minipacket before transmitting another. Second, the response bits at the back of the minipacket allow the destination to convey information back to the source without the cost of an explicit transmission. The price paid for this is that some bandwidth is wasted on the return path.

Another notable feature of the CR is the parity bit at the back of each minipacket. This is the parity of all previous bits in the minipacket and is tested and corrected at each active station. If it is found to be incorrect the station which detected the error transmits an error minipacket to an error logging station. Since the error minipacket will contain the address of the sending station, the source of the error can be localised to a single link. This has proved to be of value in detecting broken links and links with an abnormally high error rate. The monitor station co-operates in this continuous testing by placing random bits in empty packets which would otherwise contain zeros. The monitor station also checks for certain types of bit errors which could be fatal to the correct operation of the ring. An example of this is a corruption of the full/empty bit, if an empty minipacket is erroneously marked full it could circulate indefinitely, unable to be used by anybody.

A coding plug is used at each station to supply the station address and also the number of slots on the ring. Knowing the number of slots means that a source station can empty a returning minipacket without having to wait to see the source address field. The delay through a repeater and station can thus be kept small, being only 3 bits in present implementations.

Although clocked at 10MHz, the CR slots contain much control and addressing information and just 4MBit/S is available to carry user data. Only part of this bandwidth is available on a point-to-point basis and one station sending data to another can only achieve 1.6MBit/S on a ring with a single slot. Because a station

must wait for a minipacket to pass round the ring before it can transmit again, the bandwidth at a station falls as the delay in the ring increases. This is made up of the wire delay and the delay in each repeater. Adding more stations and repeaters to a ring will therefore reduce the bandwidth available at all stations.

The small size of the minipacket means that some form of protocol is necessary to send messages of more than two bytes from one host to another. A protocol which is widely used on the Cambridge Ring to perform this function is the Basic Block Protocol [Walker 78]. This allows the transmission of up to 2048 data bytes along with a port number. The port number is used, for example, to address the data to a specific process or function within the destination host. The Basic Block is sent as a sequence of minipackets (Figure 2.3c), the first being a header containing the block length and the second containing the port number.

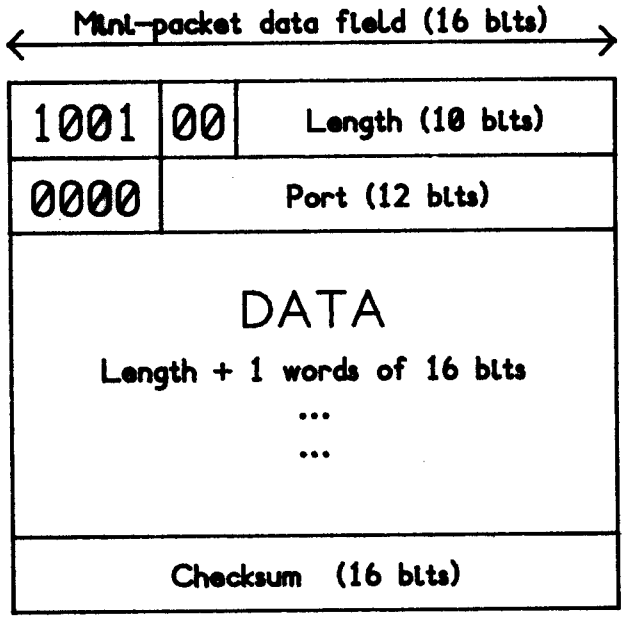



*Figure 2.3c Structure of a Basic Block*

Subsequent minipackets contain data up to the specified length and finally a minipacket containing the checksum of the data fields of all minipackets in the block is sent. The checksum provides a very strong check that all minipackets were delivered correctly, in particular that lost minipackets are detected. The very low error rate of the ring, typically 1 bit in $10^{11}$ according to [Dallas 80], means that very few minipackets are delivered in a corrupt state.

In order to allow a host to concentrate entirely on minipackets from a given source, for example while receiving a Basic Block, each station contains a **Source Select Register** (SSR) which may be written by the host. When the SSR contains zero the station will reject all minipackets sent to it by marking the response bits **unselected**. When the register contains 255 (all ones) the station will accept minipackets from any address and when it contains a valid station address (1 through 254) it will accept minipackets from that station and reject all others. In an environment where the Basic Block Protocol is used the normal state of the SSR is such that the station will accept minipackets from anyone. When a valid header minipacket arrives the host will set the SSR to the source of the header and receive the rest of the block. If at any point the destination host decides that it does not want to receive the block, it will set its SSR to zero and the source will find its minipackets being rejected and cease transmitting.

The two response bits can be used to encode four states of the destination. One of these is unselected and another is **accepted**, meaning that the minipacket was successfully received. The third state is **busy** which means that the receive buffer of the destination was full and implies that the destination is slow in processing minipackets and that the current minipacket should be retransmitted. Thus a low level flow control system is effected by the busy response. The fourth state of the response bits is **ignored**. In this case the response bits return in the same state as that in which they were transmitted, implying that the destination is switched off or non-existent.

Because the repeaters must be powered at all times for the ring to function, they must be powered independently of the host machines. Stations normally share the host's power supply and are isolated from the repeaters by a relay which is kept closed by the host supply. Power for the repeaters is carried in the ring cables and supplied by a number of co-operating supplies. Failure of one supply is not fatal, provided the remaining supplies can power all repeaters on the ring.

## 2.1 MACROLAN

Macrolan is a recent development by ICL, its main aim being the efficient linking of peripherals and mainframe computers. It is a high speed network with a data rate of 50MBit/S [Stevens 83]. The topology is an interconnection of stars; hosts connect to **stations** which then connect via a bidirectional link to a star switching unit or **port**

unit. Port units may also connect to other port units in the same way that they connect to stations (Figure 2.4a).

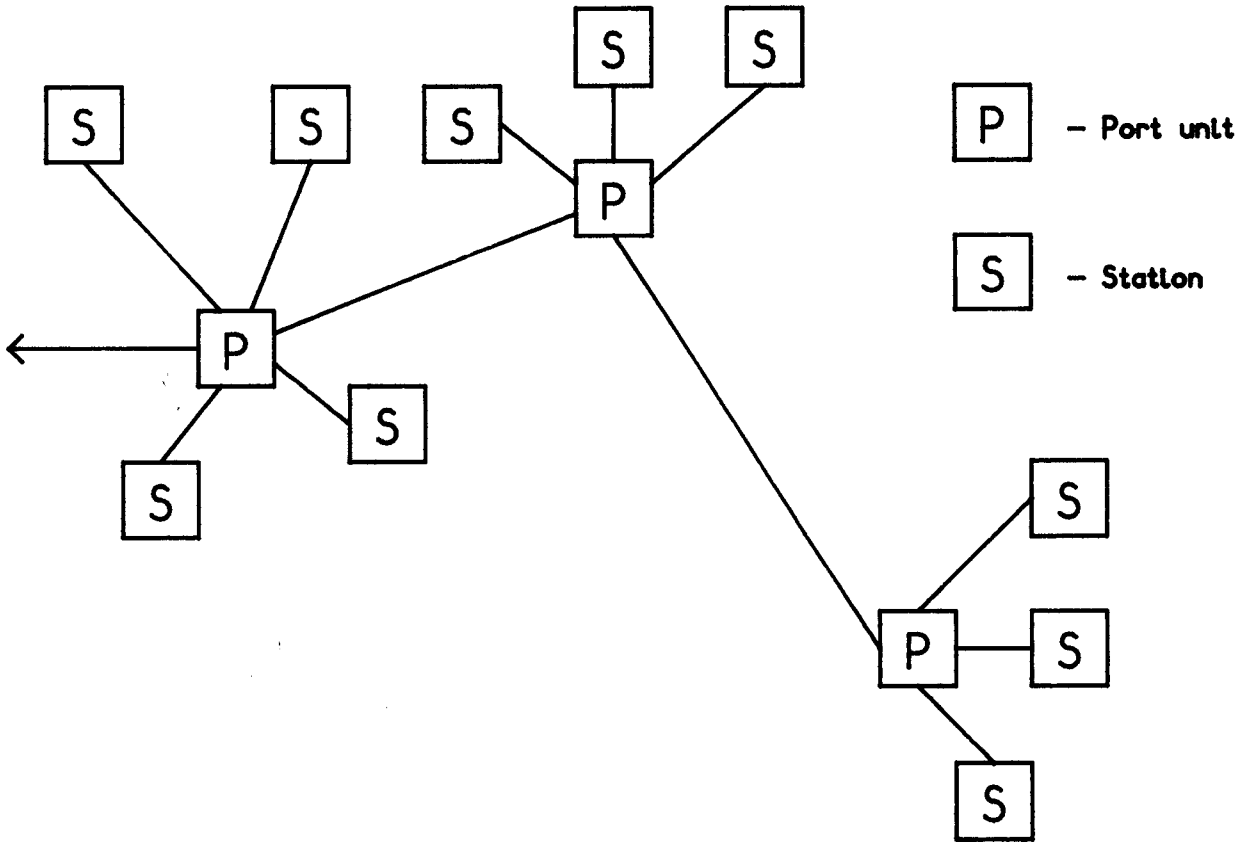



*Figure 2.4a* ***Structure of the Macrolan network***

Access to the network is by means of a token character (bit sequence) which is passed between stations by the port units. When a port unit is passed the token or "go ahead" character (GA) on one of its links, it sends it back down the next link in sequence. In this way the GA character will eventually reach a station and give it the opportunity to transmit.

When a station receives a GA character it will immediately send it back to the port unit if it does not wish to transmit. The GA will pass from station to station until one of them wishes to transmit. When a station wants to transmit a message and has the GA character, it sends a "start of frame" character (SOF) to the port unit followed by the message. The port unit recognises the SOF character and broadcasts it and the message on all its links. The message thus reaches all parts of the network and a further facility is provided whereby a message may specify that an acknowledgement is required. In this case the receiving station sends an acknowledge character (AK) back

to its port unit, which then sends it back down the link on which the message arrived. All port units do likewise and the AK is thus routed back to the source station. When the transmission is complete the source station sends a GA character back to its port unit which continues passing it to its other links.

One of the major points of interest in Macrolan is its implementation. The networks discussed so far have generally been constructed from "off the shelf" small and medium scale integrated circuits (SSI and MSI) and used conventional metal conductor transmission techniques. Macrolan has been implemented in large scale integrated circuits (LSI) and uses fibre-optic connections for the station to port unit links.

At present 50MBit/S is the maximum rate that can be achieved by inexpensive fibre-optic systems such as the Plessey HRCL system [Sumerling 82]. This rate is well within the capabilities of the Emitter Coupled Logic (ECL) from which the port unit is constructed. The port unit is implemented as three uncommitted logic arrays (ULA) and has a complexity of around 1200 gates. It provides for six links to stations or other port units.

The station is also constructed using a ULA to perform the majority of its functions, which include framing, bit stuffing and error detection. The advantages of this method of implementation are many. Cost is reduced significantly by reducing component count and construction costs. Simpler power supplies are required, since the power consumption is low. Reliability is enhanced, primarily because the hardware is much simpler than it would otherwise be using conventional techniques. The fibre-optic cables are light and easy to lay and provide a significant improvement when links must pass through electrically noisy areas.

## 2.5 IBM TOKEN RING

The IBM token ring [Bux 82] is based on the token passing principle also used in Macrolan. Access to the ring is gained by taking possession of a token which circulates around the ring. Having obtained the token, the transmitter sends its message round the ring, replacing the token when it returns. The token then continues round the ring giving each station an opportunity to transmit. This method of working, where stations transmit on a strictly round robin basis, is termed **asynchronous** transmission. The IBM token is a collection of several bits, one of which

indicates whether the token is in use. The other bits are used to detect failures of the monitor node and to provide for different transmission methods. Data is transmitted in frames with the token at the start of the frame (Figure 2.5a).

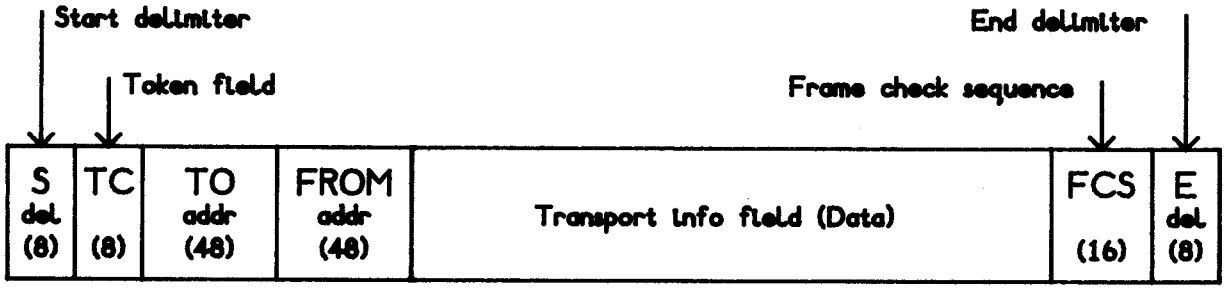



*Figure 2.5a Format of an IBM token ring frame*

The token and message frames are transmitted on the same cables and must be clearly distinguishable. The IBM ring uses violations of the line coding scheme to indicate the arrival of the token field. The ring is clocked at 4MHz and is part of a two level network which consists of many rings connected together by a high speed **backbone network** (Figure 2.5b).

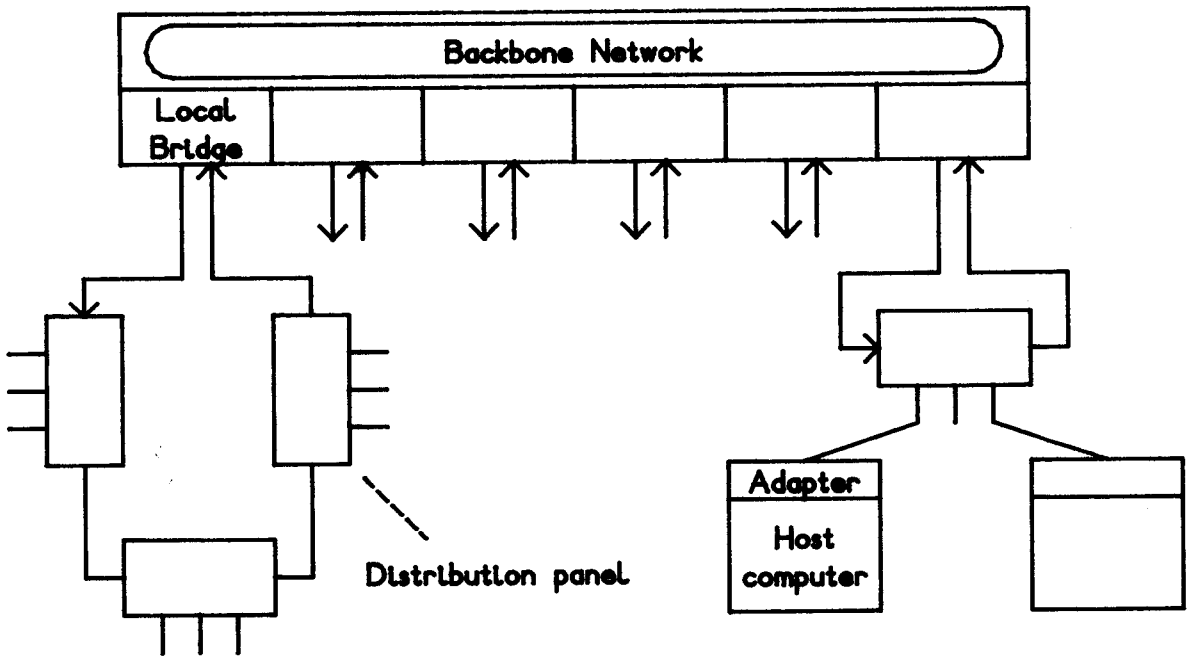



*Figure 2.5b Structure of the IBM network*

Early versions of the ring employed 32 bit addresses, but the latest version uses 48 bits to bring it in line with various proposed LAN standards. Of the 48 bits, 16 are used to address a single ring on the network and the remaining 32 identify a single host on that ring.

One of the major aspects of the design is its reliability. The configuration is a ring, but organised as a linked series of distribution panels with several arms radiating from each panel to **ring adapters**, which provide the host interface to the network (Figure 2.5c).

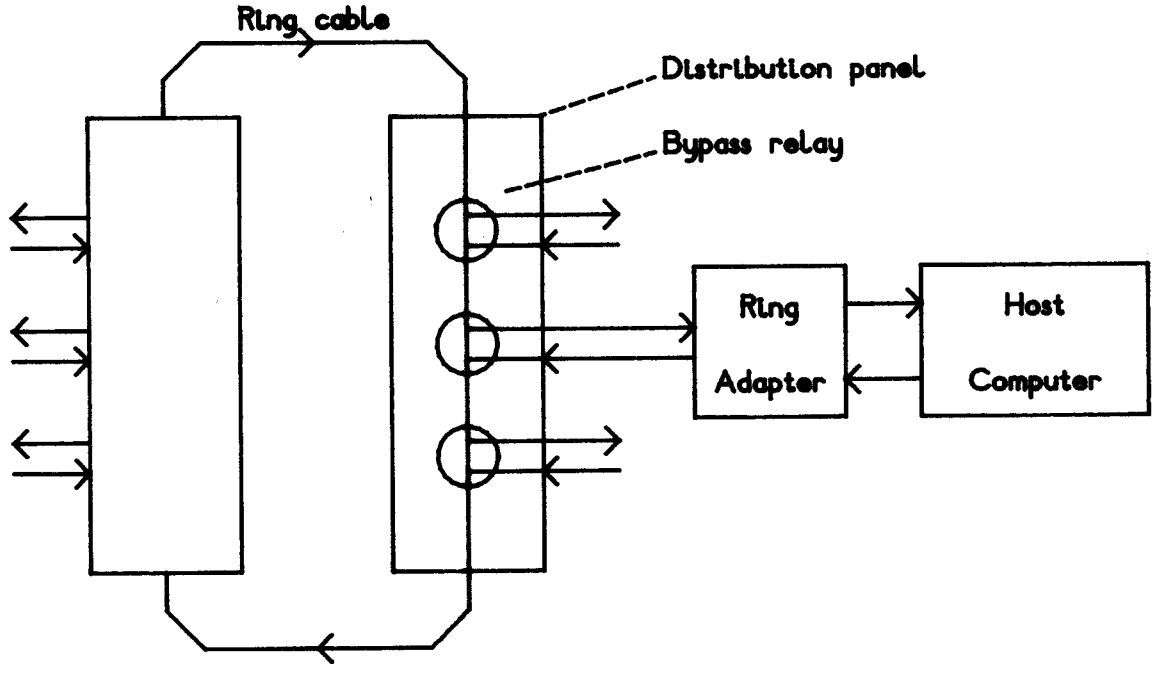



*Figure 2.5c Connection of a host to the IBM ring*

The distribution panel contains bypass relays to allow isolation of any host and when a host is isolated the relay loops the adapter's output back to its input so that self testing of the adapter link is possible. There is circuitry in each ring adapter to detect malfunction and hence operate the bypass relay.

Another aspect of reliability is ensuring the validity of the token. Normally there should only be one token on the ring but errors could remove it altogether or create several tokens. In previous token ring implementations a single dedicated monitor station checks for such errors and performs correction. The IBM ring takes a slightly different approach in that all ring adapters contain circuitry to perform the monitor function but only one is actually active at any time. The passive monitors check the operation of the active monitor and in the event of its failure arbitrate amongst

themselves to choose a new active monitor.

Another prominent feature of the IBM ring is the inclusion of a second transmission method with different properties to the asynchronous one already described. This new method is termed **synchronous** and is a way of bypassing the potentially long wait for the arrival of the token.

Synchronous mode is intended for use in applications which have a strict access time requirement, such as might be found in digital telephony or other real-time applications. The monitor coordinates synchronous mode by periodically marking the token so that asynchronous transmitters will no longer attempt to use it and synchronous ones will. The marked token then leaves the monitor and the first synchronous adapter downstream of the monitor will use the token to transmit to a (synchronous) destination. The destination may place data in the frame to give to the source and on its return the source passes the token on to the next synchronous adapter. Eventually the token will return to the monitor, which will mark it as being once again available for asynchronous use.

The backbone network is a store and forward system known as the **block switch**, whose purpose is the transfer of frames between rings. Frames are buffered in queues on and off the block switch and a method of queue jumping is provided for synchronous frames. An aggregate data capacity of over 100MBit/S is claimed for the block switch. Prototypes of the ring and block switch have been constructed and work is under way to integrate the ring adapters and other network logic onto LSI circuits.

## 2.3 PROTOCOLS AND STANDARDS

Most communication networks, no matter what their size or speed, can be usefully divided into a layered structure in which each layer provides a service to the next higher layer and is in turn dependent on the layer below. This simplifies the design of the networks and enables standardisation between networks by means of well defined interfaces to the various layers. This approach has been applied by the International Standards Organisation (ISO) in producing the Reference Model of Open Systems Interconnection. This is described in [Zimmerman 80] and consists of seven layers.

The lowest layer is the Physical Layer and this defines the mechanical and electrical interface to a network. The next (second) layer is the Data Link Layer and its function is to provide an error free transmission method for a stream of data. It will generally do this by dividing the data up into "frames" and having the destination acknowledge the arrival of frames.

The third layer is the Network Layer and this controls the sending of "packets" of data across the network. Messages passed down to this layer will be split into packets and routed to their destination. The fourth layer is the Transport Layer. This is responsible for transferring messages between hosts in an efficient manner and is the layer at which a network independent interface becomes possible. The topics discussed in this thesis correspond mostly to these lower layers, particularly the Physical and Data Link Layers. The higher layers (Session, Presentation and Application) are the user's interface to the network and are beyond the scope of this thesis.

Taking the Cambridge Ring as an example, the Physical Layer is the ring cable and modulation system. The Data Link Layer is formed by the minipacket transmission system and the Network Layer by the Basic Block Protocol. The Transport Layer is partially implemented by the Byte Stream Protocol [Johnson 80] and this has been enhanced to fully implement this layer as the Transport Service Byte Stream Protocol [Dallas 80].

The low bit error rates of LANs, their high bandwidth and low delay, mean that their protocols are somewhat different to those of earlier wide area networks such as ARPANET, which were generally based on slow telephone lines. The likelihood of error is much less and the cost and time of retransmitting a packet is small. Therefore the protocols can be simpler, performing error correction by retransmission rather than by redundant coding techniques.

The simple topologies of LANs means that there is generally only one route which data may take from source to destination. This means that packets will never arrive out of sequence, though they can still be lost on the way. Flow control can also be simplified in LANs. The low transmission delay means that it is frequently possible to send an acknowledgement for each packet sent. This arrangement would slow many larger networks considerably.

**LAN Standards**

In the past few years several committees have attempted to define standards for local area network implementations. In the USA the IEEE is the major force while in Europe ECMA has been the most active organisation. In the UK a definition of a standard Cambridge Ring, known as CR82 [JNT 82], has been prepared by the Joint Network Team. The ECMA standard overlaps the IEEE standard in places, in particular the IBM token ring and 10MHz Ethernet are acceptable to both. Overall, the emerging standard seems to be that set down by the IEEE, although even that is incomplete at the time of writing [Nelson 83].

The IEEE Local Networks Standard Committee was formed in 1980 to work on IEEE Project 802 whose aims are to provide a set of standards for LANs, from the wire level up to protocols for interconnecting local area networks. The IEEE approach is to define a layered system of standards, using the ISO model, in which the transmission medium, the topology and the distance of interconnection are all independent.

Initially the project favoured only a single LAN implementation based on the 10MHz Ethernet. This was proposed by a combination of the DEC, Intel and Xerox Corporations and known as the DIX Ethernet. This came close to being the adopted standard until it was pointed out that Ethernets were weak in certain areas, particularly real-time applications where access to the network must be obtained within a fixed time. As a result of this the project adopted two other network types as part of the standard. One of these is a token ring very similar to that of IBM and the other is a token bus. The token bus has a topology like that of Ethernet but with access controlled by passing a token between the network interfaces. The busses can use broadband or baseband signalling and all systems employ 48 bit addresses. The addressing is such that an individual host, a small group of hosts, or all hosts may be specified by a single address. The two token systems specify a priority function to give certain transmissions precedence over others.

The present status is that Project 802 has divided into a number of sub-groups each with responsibility for one of the systems or some other specific area. One group (802.6) is looking at metropolitan area networks which have a somewhat greater size than LANs and might cover a small town.

## 2.7 SUMMARY

While the networks described in this chapter are but a small cross section of the many LANs which have been built, they illustrate a number of general points. The Pierce Loop is interesting because it was an ambitious design which satisfied the need for a local network for resource sharing, but was expandable, using facilities explicitly included in the design, into a very large network. Later LANs, like the Cambridge Ring and Ethernet, were simpler and did not allow for network expansion. Recently, however, the need to expand and interconnect local networks to increase their geographic range has become apparent. The mechanisms to do this were not designed into the Cambridge Ring or Ethernet and the interconnection of these networks, in particular the addressing of packets, is performed by software. The IBM token ring and the local network described later in this thesis allow the connection of many small networks to make a larger one, by means of facilites provided in their hardware.

The methods of implementing local networks have also evolved over the years. Early networks were generally constructed with small and medium scale integrated circuits. The demand for networks has increased and it is no longer appropriate to use such methods to build network hardware which will be duplicated many thousands of times. The initial implementations of the Cambridge Ring and Ethernet were done in this way and later they were re-implemented using LSI techniques. More recent networks such as Macrolan have been designed for direct implementation in LSI.

Local networks can carry a wide range of traffic types and some are designed with a particular type in mind, or to cope with a variety of types. Macrolan was designed primarily for the interconnection of mainframe computers and disc stores. The IBM ring provides two types of transmission method, one of which provides a guaranteed access time and is particularly suited to real time applications. The network described in this thesis incorporates a similar mechanism whereby a transmitter can send data at a guaranteed rate, regardless of other traffic on the network.

Standards are emerging for local networks. The method of standardisation is that an existing network is examined in detail and then, with only minor changes, declared to be a standard. The standard is therefore only of use in constructing copies of that particular network. This is very useful but provides little guidance for the designer of a new network.

Chapter 3

# A CASE STUDY OF A CAMBRIDGE RING

This chapter details a study made on a particular implementation of the Cambridge Ring. The ring is used to support an experimental distributed computer system. This is reflected by the traffic patterns and performance characteristics observed. While other distributed systems might offer a similar load to the network, other types of application may present a significantly different load. A comparison with another distributed system based around an Ethernet is made later in the chapter.

## 3.1 THE TEST ENVIRONMENT

The system outlined below is the Cambridge Model Distributed System (CMDS) [Needham 82]. The CMDS is based around a pool of single user minicomputers known as the Processor Bank. These machines are allocated to users for the duration of an online session by a server called the Session Manager. A server is a (usually small) computer which is dedicated to a single function, such as name lookup or providing the date and time. The Processor Bank machines have a Cambridge Ring as their only peripheral and thus have no file store or terminal connected. A File Server provides a filing system for the Processor Bank machines via the ring. This is a dedicated minicomputer which interfaces a number of disc drives to the ring and provides a suitable set of primitive operations with which to implement a filing system. Terminals are connected via servers known as Terminal Concentrators, which connect up to 8 terminals to the ring. Another server is connected to a line printer and provides a printing service for machines on the ring.

Additionally, several other computers which are not part of the distributed system are connected to the ring for reasons of convenience. These include an IBM mainframe, an assortment of mini and microcomputers and the CAP computer which maintains a virtual storage system over the ring, using the File Server as secondary storage [Dellar 80]. A uniform set of protocols based on the Basic Block Protocol [Walker 78] is used throughout the system.

A typical session on the system proceeds as follows. The user at his terminal asks the Terminal Concentrator to connect him to the Session Manager and then asks the Session Manager to allocate one of the Processor Bank machines to him and load it with an operating system from the File Server. The Session Manager may well delegate some of these tasks to other servers but the user is unaware of this. When this is done (a few seconds) the user's terminal is connected via the ring to the allocated machine. The user then sees a conventional terminal session and can begin his dialogue with the operating system. The Terminal Concentrator allows several connections at once to a terminal and the filing system that the user now sees is located on the File Server. Thus, when the user compiles a program, the compiler and source file are read from the File Server and the resulting object code file is written back to the File Server. The response times for such operations are comparable to those experienced on a conventional time sharing system.

It is possible to identify some probable causes of traffic on the network. The File Server is likely to contribute heavily, since it supports filing systems for all users of the system and also supports the CAP's virtual store. The terminals and printer can be expected to receive more data than they produce and there will also be administrative data passing between servers and other machines.

## 3.2 THE MONITORING DEVICE

The measurement of traffic on the ring may be approached in several ways. If we are interested in overall traffic patterns, such as the volume of data transmitted in a given time, then we must look at every minipacket which passes a given point on the ring. The time between the arrival of consecutive minipackets is less than 4uS. The operations which would have to be performed on a minipacket would take much longer than this using a conventional computer. A dedicated piece of hardware could perform the necessary manipulations much faster, but it could be quite complex.

We might also want to observe traffic at the Basic Block level, of interest are the number of blocks sent and their lengths and types when viewed as part of a higher level protocol. A number of methods for monitoring at the block level are suggested by Ody [Ody 80]. His "promiscuous station" method uses a ring station which is capable of receiving minipackets from any chosen source without marking the response bits. This method allows monitoring of blocks between a given pair of stations and imposes no extra load on the network which might upset the

measurements.

An alternative method for monitoring blocks is to incorporate a facility in the software of each machine to send a logging minipacket to a logging device on the ring. Such minipackets would be sent at the start and end of a block and could provide information about the size of the block and its transmission time. This method will, of course, add traffic to the network, though hopefully an insignificant amount. A failing of both the above methods is that they do not allow inspection of the response bits of the constituent minipackets of a block and so are unable to correctly observe minipackets which are rejected and then retransmitted. Such information is of some importance in the design of ring interfaces.

The approach taken in designing the monitoring device was to use hardware to speed up certain operations. A small computer was used to control this hardware and perform calculations on the measurements obtained from it. The device is capable of counting minipackets with chosen contents at full ring speed. It can be configured to work as a promiscuous station while also allowing inspection of the response bits. It is attached to the ring at a repeater and is readily moved around to monitor at different points on the ring.

A diagram of the traffic monitor is shown in Figure 3.2a. The device contains a number of registers one of which, the packet register, is filled with a copy of each minipacket as it passes the repeater to which the monitor is connected. This register is therefore 38 bits long and synchronisation circuitry provides a pulse each time a new minipacket is loaded into it. Another register, the comparison register, is of the same length and may be written by the computer. The bits of this register are compared with corresponding bits in the packet register and the resulting bits fed to a third register, the mask register. The mask register is also written by the computer and its contents are ORed with the result of the previous comparison. It therefore selects the set of bits on which the comparison is made. The masked comparison bits are then ANDed together to give a signal which says whether or not the minipacket matched the pattern set up in the comparison and match registers. If a match occurs then the packet register is copied into a fourth register, the capture register, where it may be read by the computer. A count register is incremented when a match occurs, it too is readable by the computer.

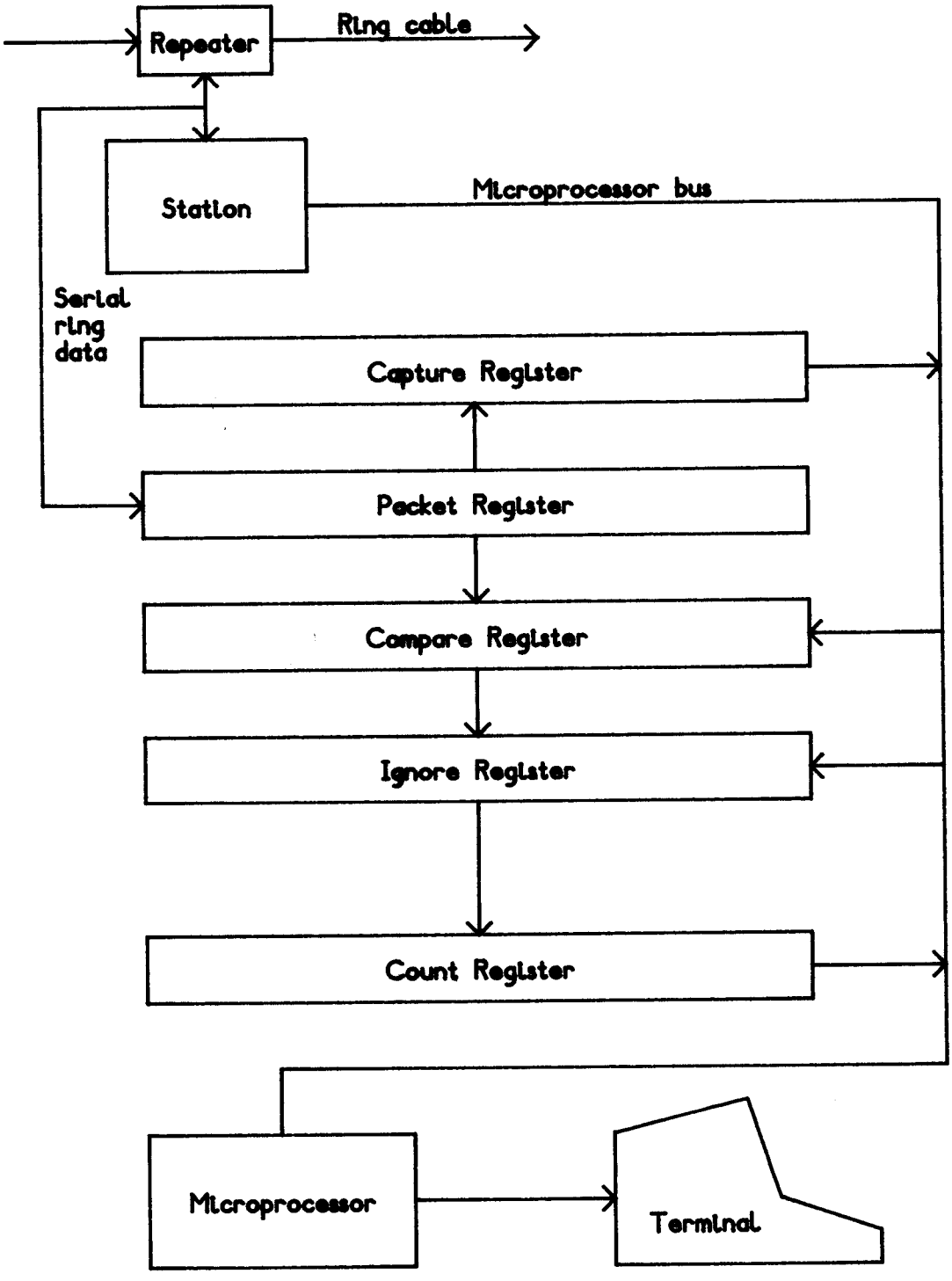



*Figure 3.2a Diagram of the Traffic Monitor*

The device therefore provides facilities for counting and reading minipackets whose contents match a pattern set up by computer. While it is not possible to alter the pattern conditionally within a minipacket time, this can be done within a ring

revolution time and this allows satisfactory monitoring of blocks.

A characteristic of the Cambridge Ring which makes traffic monitoring difficult is the use of response bits. These are marked at the destination station and thus the monitor must be placed after the destination if they are to be observed in their marked state. This means that if the monitor is correctly placed to observe minipackets sent from station A to station B, then it is not possible to fully observe minipackets from B to A without changing the monitor's connection to the ring. Two monitors can be used to get round this problem.

One of the first tests performed with the monitor was the measurement of ring utilisation, that is the proportion of the available bandwidth in use over a given time interval. The fundamental measurement that the monitor must make for this experiment is to measure the number of full minipackets passing in a set time. The method used is described below to illustrate the use of the monitor.

### 3.2.1 Use of the Traffic Monitor

In order to count full minipackets the monitor is configured by the controlling computer as follows. The bit in the comparison register which corresponds to the Full/Empty bit is set to 1, indicating that a match should be given on full minipackets. The corresponding bit in the mask register is set to 0 and all others to 1, which means that a match will be given on any full minipacket regardless of the state of any other bits in it. Having done this the counter is cleared by the computer and then allowed to start counting. The computer now waits for the desired time interval before stopping the counter. The counter may now be read and the average utilisation over that time interval can be calculated.

A slightly more complex use of the monitor is in looking at Basic Blocks passing from one station to another. To do this the monitor must be positioned after the destination station in order to see the marked response bits. The registers of the monitor are initially set up to look for full, accepted packets with the appropriate source and destination and the Basic Block header pattern in the data field. This pattern is '100100' in the 6 highest bits of the data. If the ring carried completely random data then the chance of this pattern occurring in a data minipacket is one in 64. There is a possibility therefore that one of these minipackets will be mistaken for a Basic Block header and the block checksum must be used to verify the validity of any block which is received. The data carried by the ring was not completely random and

the chance of the header pattern occuring in data was rather less than one in 64 but still significant.

The computer detects the arrival of a minipacket which matches the pattern set up in the monitor by seeing the count register increment. It then reads the data field of the capture register to deduce the length of the block and reloads the data field of the mask register so that subsequent minipackets are captured regardless of the content of their data fields. The remaining minipackets of the block will now be captured by the monitor and placed in a buffer by the computer. When all have been captured the computer can validate the checksum and thus ensure that the first minipacket was really a block header.

A number of variations of this test are possible, such as measuring the time between successive minipackets and observing the number of rejected minipackets at various stages of the block transfer.

## 3.3 OBSERVED TRAFFIC PATTERNS

On the ring tested the total bandwidth available to all stations, the system bandwidth, was 3.2MBit/S while the maximum bandwidth available to a single station, the point-to-point bandwidth, was 0.64MBit/S (20% of the system B/W). The ring was clocked at 9.8MHz and contained 3 slots and a gap of 33 bits. There were 30 active stations and these connected the following computers to the ring:-

1 IBM 370/165 mainframe

2 PDP11/45 minicomputers

9 LSI4 minicomputers (File server and Processor Bank)

2 NOVA minicomputers

15 Z80 microcomputers (small servers, terminal concentrators)

1 CAP experimental computer

The LSI4 machines and the Z80's make up the Cambridge Model Distributed System. The CAP computer has a virtual memory system which uses the File Server as secondary storage, while the remaining machines are connected mainly for convenience. For example, the IBM 370 is used for tape archiving. Terminal sessions and job submission on this machine are also possible from terminals on the ring.

### 3.3.1 Utilisation

The utilisation was observed to vary widely over short time periods. There were short term peaks of activity, but over longer periods the average utilisation was moderately stable. At times of heavy use the utilisation was around 4% when measured over 30 minute intervals. Heavy use implies that most of the Processor Bank machines were in use, as was the CAP computer. At the time of the tests there were 6 active computers in the Processor Bank. All of these made frequent use of the File Server and this accounted for the majority of the traffic on the ring. At times of minimum ring use the utilisation was 0.1%, most of which can be attributed to the Logger, a server which "prods" each of the 255 ring addresses with a minipacket once per second. Over 24 hours the utilisation was typically 1.3%.

With short sampling times the utilisation can be much greater, though the majority of samples exhibit a very low utilisation, for instance more than 80% of all 1mS samples have a utilisation of well below 1%. A single station on the test ring could produce a utilisation of 20% if it transmitted at full speed and a Basic Block can consist of up to 1027 minipackets. It is reasonable, therefore, to expect to see utilisations of 20% lasting for up to 25mS. This was indeed seen to occur but was never seen to exceed 20%, indicating the low probability of two stations transmitting at the same time. The only pair of stations which were capable of this transfer rate were CAP and the File Server, no other machines having ring interfaces of sufficient speed. An operation such as loading a Processor Bank with an operating system of 30 KBytes lasts around 0.5 seconds, during which the utilisation is 10%.

This experiment demonstrates that ring traffic is bursty in nature, a low level of background activity is interspersed with short periods of high activity. Overall, the utilisation remains low. At no time is the system bandwidth inadequate, though the point-to-point bandwidth is a limiting factor for those machines with inherently high transfer rates. The File Server could clearly benefit from higher bandwidth, but only if the stations with which it communicates could keep up.

### 3.3.2 Useful Traffic

Useful traffic means the proportion of minipackets which are accepted by their destination. This may be influenced by the level of protocol above the minipacket level and also by the characteristics of the receiving computers and their ring interfaces. With the exception of the prodding minipackets sent by the Logger, almost all

minipackets in the test environment were part of Basic Block transfers.

Exact measurement of the overall numbers of minipackets with each of the four responses is not possible since one of the responses (ignored) cannot be counted properly. All minipackets are sent bearing this response initially and we cannot, while monitoring a single point on the ring, deduce whether an ignored response is implied or whether the minipacket has not yet passed its destination. A solution to this problem is to also monitor empty packets which have a valid response in them. The monitor station (not the traffic monitor) may be configured to fill empty minipackets with zeros. By placing the traffic monitor just before the monitor station we can estimate the relative numbers of three responses including ignored. The busy response, encoded as zeros, cannot be measured in this part of the experiment. An assumption must be made that very few minipackets which have been marked empty at their source are re-used before they reach the monitoring point. This is reasonable at low utilisations and the figures below were obtained at a utilisation of 3.4% (over 30 minutes).

Empty minipackets (not possible to count BUSY response)

| | |
|---|---|
| Accepted | 86200 |
| Unselected | 1030 |
| Busy | ? |
| Ignored | 7200 |

Full minipackets (not possible to count IGNORED response)

| | |
|---|---|
| Accepted | 21800 |
| Unselected | 900 |
| Busy | 3600 |
| Ignored + unmarked | 65100 |

Combining these two gives an estimate for all responses

| | |
|---|---|
| Accepted | 86 % |
| Unselected | 2 % |
| Busy | 7 % |
| Ignored | 5 % |

These measurements indicate that the Basic Block protocol is an efficient means of transporting data. The low proportion of unselected responses means that little bandwidth is wasted by excessive numbers of attempts to establish a block connection. The ignored responses can be largely attributed to the Logger transmitting to non-existent addresses, while the slightly higher busy figure suggests that there is some degree of speed mismatch amongst the various ring interfaces. The busy

response is intended to promote efficient transmission in the event of such mismatches. A station getting a minipacket back with the busy response will retransmit it a short while later, by which time the receiver should be ready to receive it.

### 3.3.3 Block Lengths

Measuring the lengths of blocks is performed as described previously. The header minipacket is accepted without regard for source or destination and the addresses found in that minipacket are used subsequently. When the block has been captured another block header is looked for, again regardless of source and destination. Clearly every block cannot be monitored in this way but over a long period of time (several hours) a representative picture of the distribution of lengths can be obtained. The results of this experiment are shown below in Figure 3.3a and Figure 3.3b, the first histogram shows an overall picture while the second shows the low block length interval in more detail

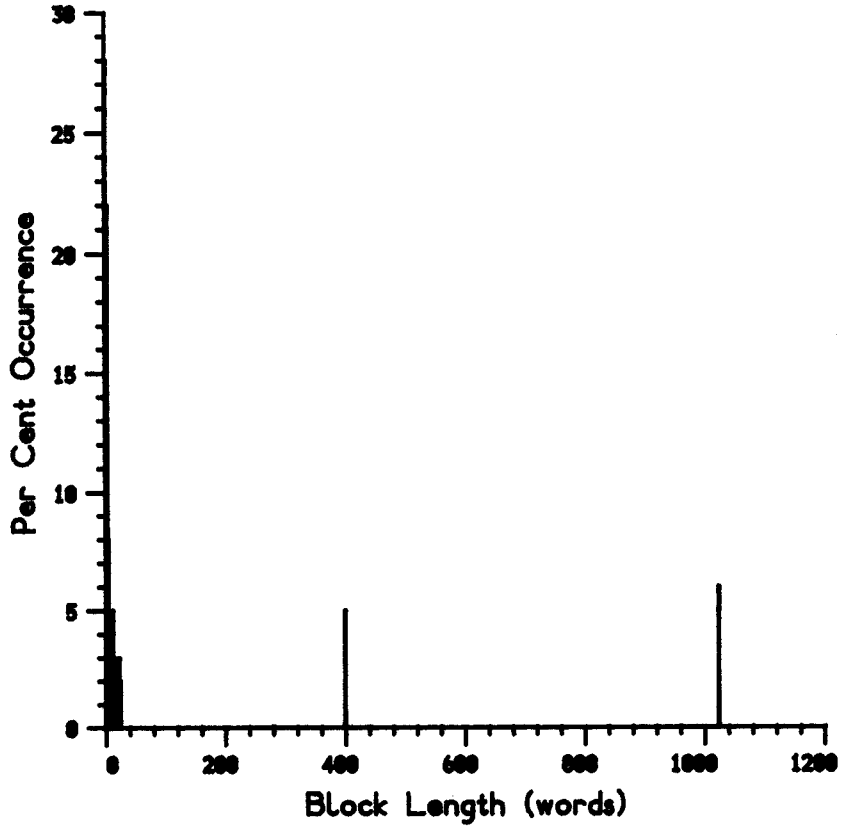



*Figure 3.3a Distribution of Basic Block Lengths*

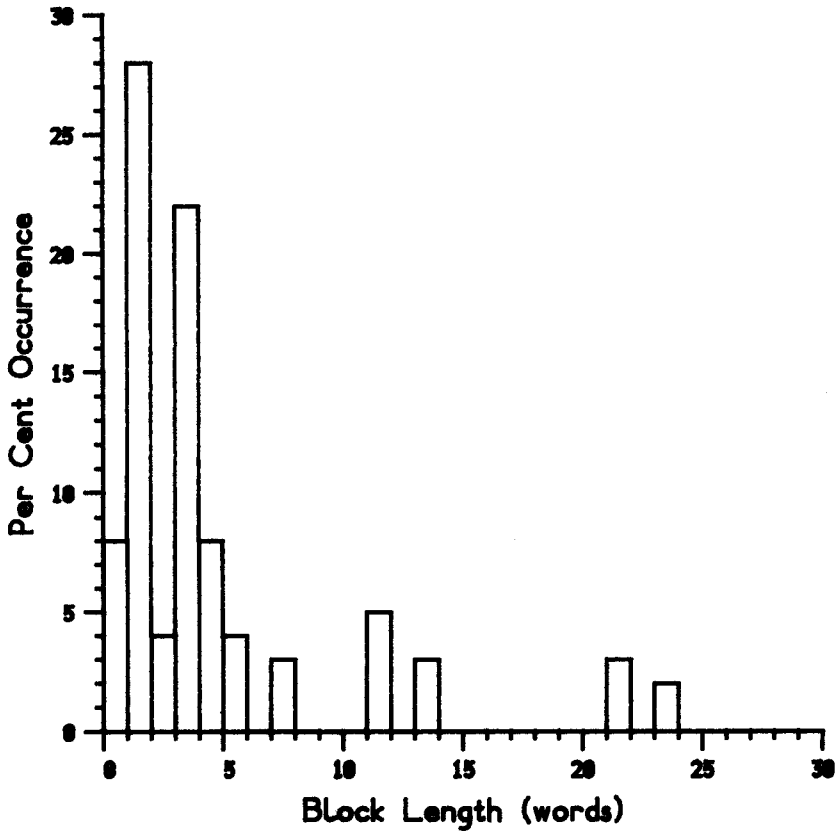



*Figure 3.3b Distribution of Basic Block Lengths*

As expected there is a bimodal distribution of lengths, the majority of data being transmitted in maximal length blocks. However, the majority of blocks are quite short. Over 80% of blocks carry less than 16 words of data and 90% of the data is carried in the remaining blocks, most of which are 400 or 1024 words long. The blocks of length 400 are due to the Processor Bank machines having a 400 word buffer with which to perform transfers to and from the File Server.

Most of the small blocks are control blocks used in the higher level Byte Stream protocol, many of these being associated with the Terminal Concentrators. These send large numbers of small data blocks, particularly when working in single character mode and generate similar numbers of control blocks. At times of heavy use it was observed that about 200 blocks per second were being sent.

## 3.4 PERFORMANCE UNDER HEAVY LOAD

While the ring in the experiments described above was never heavily loaded, for example with a utilisation of 50% for a prolonged period, it is interesting to consider how the performance would change in such circumstances and what factors determine the bandwidth available. The system bandwidth (sysBW) is calculated as follows:-

$$\text{sysBW} = \frac{\text{Number of data bits on ring}}{\text{Total number of bits on ring}} * \text{clocking rate}$$

On the test ring there were 3 slots and a gap of 33 bits, the clocking rate being 9.8MHz. Hence the system bandwidth was:-

$$\frac{3 * 16}{3 * 38 + 33} * 9.8 = 3.2 \text{ MBit/S}$$

Provided that the gap is small (less than 5 bits) the sysBW is roughly independent of the number of slots and in 10MHz rings is just over 4MBit/S.

The point-to-point bandwidth (ppBW) is dependent on the number of slots for the following reason. Consider a ring with N slots; when a station transmits it must wait for the minipacket to return (N slot times + gap time) and then wait a further 2 slot times before transmitting again. This wait is a feature of the Cambridge Ring and could be reduced to one slot time by redesigning some of the circuitry. Thus the minimum time between transmissions is N+2 slot times (plus the gap time). Assuming the gap to be small then we have:-

$$\text{ppBW} = \frac{\text{sysBW}}{N + 2}$$

This assumes that no other stations are using the ring, if M stations are transmitting as fast as the ring will allow, then the ppBW available to each is as follows:-

$$\text{ppBW} = \frac{\text{sysBW}}{M + N} \quad \text{(given } M > 1\text{)}$$

and the utilisation under such conditions is:-

$$\text{Utilisation} = \frac{100 * M}{M + N} \quad \text{(per cent)}$$

Note that this means that 2 stations may transmit with the maximum ppBW without interfering with each other. Computer simulation of situations in which many stations compete for bandwidth indicates that it is shared fairly and an experiment involving up

to 6 stations was performed which verified this. A number of microcomputers were loaded with a program which caused them to transmit minipackets as fast as the ring would allow. The resulting utilisation when one to six of the microcomputers were transmitting was measured with the traffic monitor and found to agree with the predicted figures.

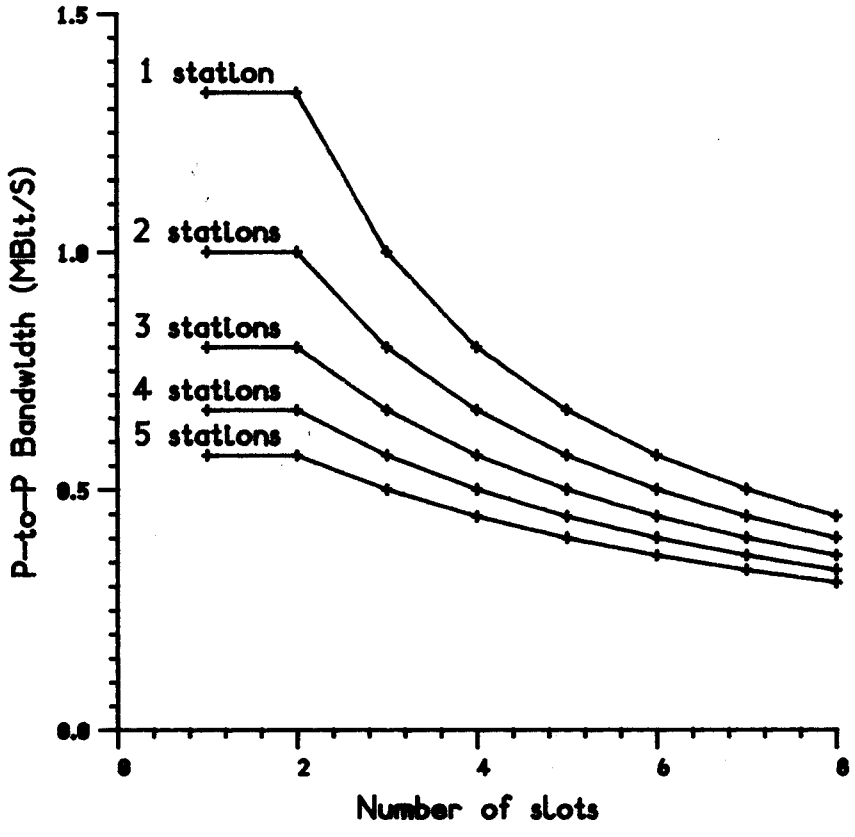



*Figure 3.4a* ***Variation of maximum point-to-point bandwidth***

A graph of ppBW as a function of the number of slots is shown in Figure 3.4a. Several sets of points are plotted, representing different numbers of stations transmitting at full speed. A graph showing utilisation for similar parameters is shown in Figure 3.4b. From Figure 3.4a it can be seen that a single slot ring provides the highest ppBW and that the degradation in ppBW with increasing load is quite gradual in rings with several slots.

Increasing the ppBW can be done by reducing the number of slots in a ring or by increasing the sysBW. Increasing the clocking rate increases the sysBW but also increases the number of slots on a ring of fixed size, thus the ppBW remains approximately constant. Increasing the sysBW can be achieved by increasing the

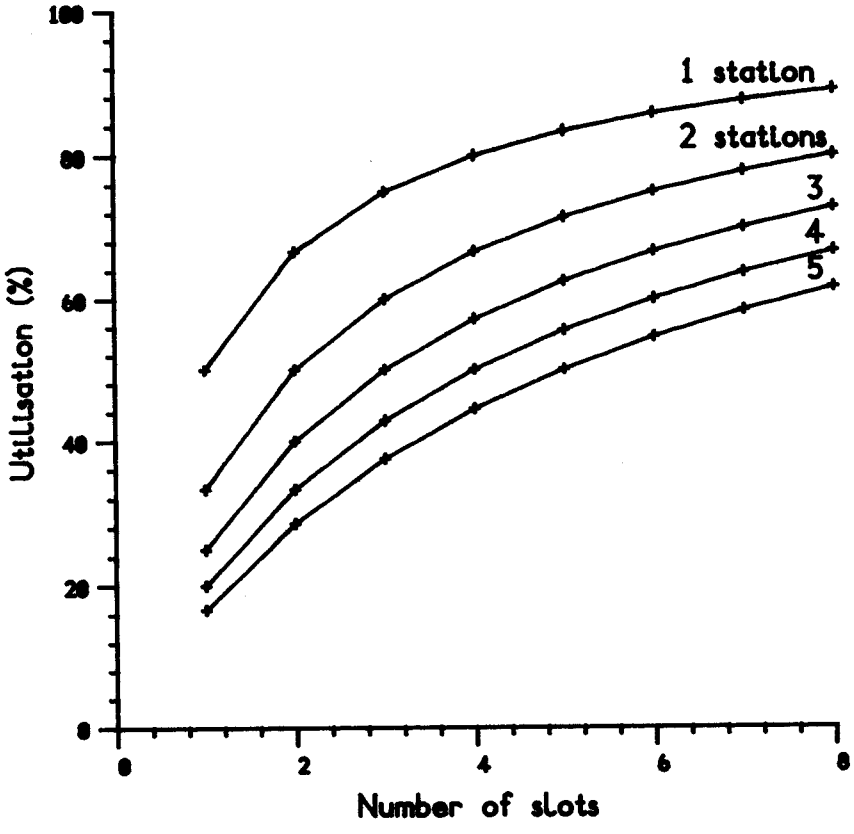



*Figure 3.4b* ***Variation of utilisation***

amount of data in a minipacket though this is a design change and so not applicable to existing Cambridge Rings. If we consider the present minipacket format enlarged to hold 8 data bytes then the sysBW would become 7.44 MBit/S (assuming a small gap and 10MHz clock). The number of slots on a ring of fixed size will be roughly halved and the ppBW will be more than 4 times that of a Cambridge Ring of the same size.

Reducing the number of slots amounts to reducing the delay in the ring. This is made up of delay in the ring cables and a 3 bit delay at each repeater. It is possible to use a **repeater extender** which allows up to 6 stations to use a single repeater and will reduce the ring length by up to 9 bits. It is, however, possible to cause a **fall** in bandwidth in this way, since a ring with a large gap has less ppBW and sysBW than a ring with one more slot and a small gap. This is illustrated in Figure 3.4c.

A more drastic way of reducing the slot count of a large ring is to separate it into two smaller rings joined by a **bridge** device (normally a dedicated computer). In this way ppBW on the two rings will be increased, though traffic through the bridge may be slower and care must be taken in deciding which machines should remain on the same ring and which can tolerate slightly slower communication through the bridge. This

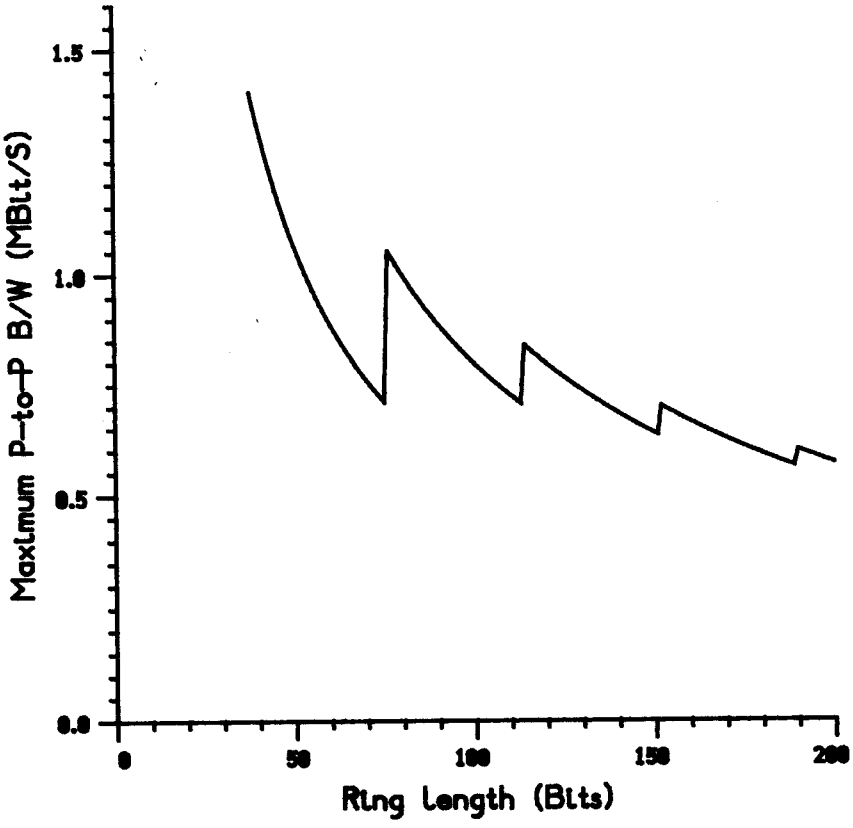



*Figure 3.4c* ***P-to-P bandwidth as a function of ring length***

course of action has been taken on the Cambridge system; the File Server, Processor Bank machines and CAP are sited on the same ring, while the Terminal Concentrators and printing servers are on the second (larger) ring.

A further worthwhile consequence of splitting rings in this way is that reliability is enhanced since the failure of one ring will not affect the operation of the other. Of course it may be that machines on the working ring require resources on the failed ring, but the situation is nonetheless better than if everything were on a single broken ring! Bridge design and the splitting of rings are discussed in [Leslie 83].

## 3.5 COMPARISON WITH ETHERNET

A series of experiments have been performed by Shoch and Hupp of Xerox PARC on an Ethernet in an environment similar to that described above [Shoch 79]. The following table compares the two systems over some interesting parameters.

| | Ethernet | CMDS |
|---|---|---|
| Number of active stations | 120 | 40 |
| Daily throughput | 300 MBytes | 430 MBytes |
| Average utilisation over 24 hrs | 0.86 % | 1.3 % |
| Maximum utilisation over 1 hr | 3.6 % | 3.7 % |
| Maximum utilisation over 1 sec | 37 % | 20 % |

One of the major differences between the two systems is in the location of disc files. Processors on the Ethernet have a disc drive directly connected to them, which is used for most short term storage. There are also file servers but these are mainly used for shared data and longer term storage. Processors in the CMDS have no local discs and all files are kept remotely on the File Server. This is probably the reason for the much larger volume of traffic moved, on a per station basis, by the CMDS.

Another difference is in the maximum observed utilisation over short time periods. The Ethernet's maximum over one second was 37% compared to only 20% on the ring. This reflects the fact that a single host on the Ethernet can induce close to 100% utilisation by sending consecutive long packets, whereas a single host on a (3 slot) ring can only produce 20%.

An Ethernet packet can contain up to about 540 bytes of data and so is comparable to the ring's Basic Block in size and also in function. The Ethernet experiment demonstrated a bimodal distribution of packet lengths, with the average size of short packets being somewhat larger than that of short ring blocks. The Ethernet median was 14 (data) words and the ring median 2 words. This disparity disappears when one considers that all Ethernet packets contained protocol information for internetworking (they conformed to the PUP protocol [Boggs 80]) and thus contained 11 words over their true data content. The Ethernet's uses of short packets are much the same as their ring counterparts, that is they are largely protocol acknowledgements and terminal traffic. On the Ethernet 17% of packets were "long" as opposed to 11% on the ring and 66% of data was carried in these long packets (94% on the ring).

Under normal operating conditions the Ethernet and ring are remarkably similar, both in terms of the uses to which they are being put and the traffic characteristics which are observed.

## 3.6 THE GROWTH OF THE CMDS

The CMDS configuration described earlier in this chapter has evolved and grown since the traffic study was made. The number of stations grew until the ring had 4 slots and a large gap. Further stations would have meant a 5 slot ring and transactions with the File Server had become unacceptably slow by this stage. This was due in part to the increased number of clients of the File Server, but was mainly because of the lower point-to-point bandwidth of the ring.

To remedy the latter failing, the ring was split into two and the new rings joined by a bridge computer. The faster ring of the two had 2 slots and connected the Processor Bank machines and File Server along with a small number of essential servers. The remaining machines, including the Terminal Concentrators, were placed on the other ring which had 3 slots and was the slower of the two. Interactions between the File Server and Processor Bank machines could now proceed at high speed, while the less demanding terminal traffic passed through the bridge.

The bridge is a computer with two ring interfaces, one on each ring. Rather than passing minipackets from ring to ring, the bridge deals in Basic Blocks. If minipackets were passed the responses would be useless and different protocols would be required. The use of Basic Blocks as the unit of bridge traffic also means that the same station address can be used on both rings and more than 256 stations can be accommodated on the combined network. The method of bridge operation is as follows.

On the CMDS system, hosts and the services they offer are identified by textual names. Servers known as **name-servers**, whose address is known by all hosts, are used to translate the names into ring addresses. When a name-server is asked to translate a name to an address and the address is on another ring, it supplies not the true address, but the address of the bridge which is used to reach the address. It also supplies a unique port number and notifies the bridge that it should retransmit any incoming Basic Blocks which have that port number to the true address. The host which originally asked for the name to be translated sends Basic Blocks bearing the appropriate port number to the bridge, which then retransmits them to their proper

destination.

The bridge is a potential bottleneck since it can only be receiving one Basic Block at a time. This can cause problems when a machine sends many long blocks through the bridge. There is a chance that other blocks trying to get through the bridge will be delayed for so long that timeouts will expire and the transmissions fail.

The splitting of the rings was a major undertaking, principally because of the many changes which had to be made to the communication software. It was a successful exercise, however, and the perceived performance improved as a result. The Processor Bank now has over 30 minicomputers in it and a second File Server has been added.

The bandwidth of slotted rings is very closely related to the number of slots on the ring and this is, in turn, related to the number of stations. As the ring becomes larger, so it becomes slower and if it becomes too slow, the only solution is to split it as described above. Splitting implies a bridging mechanism and if the bridge is slower than the rings then careful division of hosts between the rings is necessary. The Cambridge Ring bridge is complex, both in hardware and software. It uses three microprocessors and needs to interact with name-servers on each ring. It is vulnerable to overload and malicious traffic and expensive enough to prevent its duplication for reliability reasons. A much simpler bridge would be preferable. Operating at the minipacket level would be better but the Cambridge Ring architecture will not support this. A new network based on similar principles to the Cambridge Ring could incorporate such a facility.

## 3.7 CHANGES TO THE CAMBRIDGE RING

One of the facts to emerge from the study detailed earlier is that when a disc drive is connected to a network like the Cambridge Ring, the network limits the performance that may be had from the disc. A modest disc drive can move data at rates in excess of 5MBit/S and high performance drives are capable of several times this rate. The Xerox distributed system based around an Ethernet gets round this problem by having local discs on its hosts, in addition to file servers on the network. The disc is used as a cache and in certain circumstances can speed file based operations. On the ring system it was the speed of the ring interfaces on the Processor bank machines which limited their transfers with the File Server.

Current Basic Block Protocol implementations are such that multiplexed reception of minipackets from different sources is not possible at a host. Thus, the bandwidth into a station is limited to the point-to-point bandwidth as opposed to being able to approach the system bandwidth. While this protocol works well for most machines on the ring, it does preclude multiplexed reception which could improve performance at servers such as the File Server. Whitehead describes a study made using the traffic monitor which supports this argument [Whitehead 82]. The Ethernet does not suffer from such problems and moreover the whole of its channel capacity is available to a transmitter once it has gained access to the network.

If a slotted ring is made faster by increasing its minipacket size (in terms of data) and possibly also increasing its clocking rate, then not only would the performance increase, but the way in which the ring was used might change. If a minipacket could now hold what was previously a small Basic Block a number of improvements would ensue. First, the transmission time of such blocks would be considerably reduced. If an N minipacket block were previously used then the transmission time would be at least N ring revolutions whereas it would now be 1 revolution, provided the minipacket is accepted. Second, there would be a considerable saving both in host software and processing time, since the data no longer needs to be split over several minipackets.

If the minipacket were able to carry a significant amount of data in addition to any protocol information, then the need for Basic Blocks up to a certain size would disappear and multiplexed reception would be made possible. The minimum minipacket data field needed to efficiently support existing protocols and still allow some data to be carried is about 8 bytes.

Thus a larger minipacket would improve the efficiency of some existing applications and also lead to a simplification of protocols. A larger address field would allow many more stations and make network interconnection by minipacket bridges a possibility. It is assumed that suitable host interfaces to the network could be designed. Existing fast interfaces are costly and complex. Gibbons describes some of the problems involved in the design of "Type 2" interfaces for the Processor Bank machines [Gibbons 80] and Garnett describes a more sophisticated interface which reduces the processing load on the host by implementing protocols up to the Byte Stream level [Garnett 83].

Chapter 4

# DESIGN CONSIDERATIONS OF THE CAMBRIDGE FAST RING

This chapter presents some of the topics which were considered in the design of a high speed LAN called the Cambridge Fast Ring (CFR). As its name suggests it is similar to the existing Cambridge Ring. Whilst other types of network were considered, the wealth of experience which had been gained in the Cambridge Computer Laboratory with slotted rings made the choice of this type of ring easier than it might have been. The following section discusses the suitability of various network types for high speed operation. Subsequent sections explore some aspects of design for a high speed network with particular reference to the slotted ring.

Apart from speed we are also bound to consider the cost of implementation and connection and the reliability of the network. Cost is reduced and, to a lesser extent, reliability enhanced by implementing the network in VLSI logic.

## 4.1 TOPOLOGY

Existing LANs use a small number of network topologies. Early, experimental LANs explored a range of topologies; stars, busses and rings figured prominently, often connected together to improve reliability or geographic range. More recently, commercially available networks have centred on bus and ring architectures. One of the primary attractions of these simple topologies is that explicit routing is unnecessary, a transmitted message is seen by all potential receivers.

### 4.1.1 Mesh Networks

Mesh networks generally comprise two sorts of nodes, switch nodes which route packets through the network and terminal nodes which connect devices to the network. A number of interconnection schemes are possible, ranging from completely general interconnects through systematic schemes (e.g. hypercubes), down to the simple topologies described later in this section. Mesh networks have most commonly been used over large areas in a store and forward mode. This is unsuitable for a local network due to the problem of routing packets between switches with minimal delay. If a switch has to buffer many bits of addressing information before it can decide

where to send the packet then the delay will be high and the throughput low. Hopper and Wheeler propose a binary routing network [Hopper 79] which overcomes the delay problem but the overall network design is complex by comparison with simpler topologies offering similar performance.

### 4.1.2 Star Networks

Star systems employ a central switching node which controls bidirectional links to hosts arranged radially around the central switch. A variety of switching algorithms, implemented in hardware or software, may be employed in the switch. The centralised control inherent in the switch has both advantages and disadvantages. The complexity of the switch may be high and thus the initial cost of a star network may be large. The network is vulnerable to the failure of the switch but the topology is in other ways potentially more resilient to failure than others. The simplicity of connection to the switch makes its duplication simple and the location and isolation of faulty links is easy. Also damage to one link will not affect operation of the remainder.

Only point-to-point links are needed and thus the transmitters and receivers are simple and may employ fibre optics. Compared to a bus or ring system the cabling costs may be higher since more cable is required, particularly when hosts are clustered together a long way from the switch.

### 4.1.3 Bus Networks

The best known bus system is Ethernet. However, many other implementations exist, including a number using broadband rather than the more usual baseband transmission method. LocalNet [Biba 81] is a broadband network capable of carrying a variety of differing traffic types on a single coaxial cable. The most common access strategy for busses is CSMA, though a number of other systems exist including one based on token passing between stations. In the latter scheme the time spent waiting to transmit (the latency) is high since token passing is slow. In addition, since each station must know to whom to pass the token, the switching on and off of stations makes the token passing method complex.

Ethernet systems using CSMA provide good performance but are not well suited to working faster than the 10MHz at which the fastest present systems run. The CSMA access method means that the smallest packet must be able to span the bus. At higher speeds the minimum packet size will rise and collision arbitration will take

longer in terms of bit time. The efficiency of the network will be lowered, particularly when small messages are sent, as is frequently the case. All requests for access are treated with equal priority and it is not possible to share the bandwidth other than completely fairly.

One of the advantages of bus systems is that the bus can be made passive and thus not dependent on the reliability of active repeaters. However, in the event of a cable break it is unlikely that the two portions of the severed bus would continue to function independently, since they would no longer be correctly terminated. Contention busses must be kept small to remain efficient. Small size also has the advantage that the line drivers, which must drive the full length of the bus, can be kept simple.

Overall bus systems do not scale well for high speed working though they have advantages of low start up cost and no need for centralised control.

### 4.1.4 Ring Networks

There are 3 widely recognised access methods for ring networks. The slotted ring (e.g. Cambridge Ring) and token ring (e.g. IBM token ring) have been widely implemented, whilst the register insertion system [Hafner 74] has been employed rather less.

Rings scale well to high speed working, particularly if their size can be kept small. Rings are active systems and this is a major drawback from a reliability point of view. Many schemes have been proposed to overcome the problem of repeater failure and ring breaks, involving duplicate rings or repeater connections which automatically bypass faulty links (see [Penney 78] for a discussion). In an environment where high reliability is of utmost importance, a more complex network with high redundancy may be more appropriate than the simple systems discussed here.

The three access methods provide slightly different characteristics which change when the speed of operation alters. Token rings provide the highest bandwidth but their latency can be high. Slotted rings have lower latency but high packet overheads and thus the bandwidth is rather lower. Register insertion systems have very low latency and high bandwidth but are complex in design. A simple form of access priority can be readily implemented on slotted and register insertion rings.

A drawback of rings is that some form of centralised control is generally required. The slotted ring needs a special node to initialise and maintain the slot structure and rings are vulnerable to the failure of this node. The token ring and, to a lesser extent, the register insertion ring lend themselves to the distribution of control amongst all stations. This is done at the cost of greater station complexity.

### 4.1.5 Summary

None of the local network architectures discussed is ideal, bus systems do not work well at high speeds but otherwise there is plenty of choice. Mesh systems provide for redundancy but the complexity of routing outweighs their advantages. Overall, rings appear to be the best candidates for general purpose working at high speeds, provided due care is given to reliability. Of the ring systems, the slotted ring is appealing because of the ability to have several slots in existence at once. The slots may be differentiated by some marking mechanism which allows a number of schemes for restricting the use of the different slot types by individual hosts. Therefore the bandwidth of the ring may be allocated in a controlled, not necessarily fair, manner. The IBM ring provides a similar sort of function, but its realisation on a token ring is complex. The slotted ring allows "partitioning" of bandwidth in a much simpler fashion, although the partitions may be rather more static than can be achieved with the IBM system. A slotted ring was chosen as the basis for a new high speed network and the rest of this chapter discusses various design issues with relevance to slotted rings.

## 4.2 BANDWIDTH

By definition, the bandwidth of a fast local network should be high. A secondary requirement which was thought to be important is that it should be partitionable. In terms of speed a factor of two increase would make existing applications run faster but a factor of ten increase would be much more interesting, opening up new fields of application. Thus a clocking rate of 100MHz was selected as a target speed and the rest of the design process assumed that this rate was achievable. It was also recognised that operation at lower speeds was likely and should be catered for. When varying the speed of a slotted ring, it is wise to vary the slot size in proportion to the speed to keep a reasonable number of slots on the ring. Just what a reasonable number is will depend on the environment in which the ring is used. Current uses of rings suggest that the number of slots will generally be less than 10.

Being able to vary the slot size to suit the conditions of use is a useful facility and was included in an early LSI implementation of the Cambridge Ring [Hopper 81] in which the slots could carry between 1 and 8 bytes of data, a range of slot size from 32 to 88 bits. In a ring with several slots there is no reason why all the slots must be of the same length and the mixing of long and short slots may be advantageous if there are a number of different traffic types on the network. A network with both computer data and voice traffic could benefit from such a scheme. The segregating of the two traffic types means that an appropriate slot size may be chosen for each and the two sets of traffic will not interfere with each other. Voice data would be carried in short (1 data byte) slots and other traffic, requiring more bandwidth, in longer (8 data byte) slots. Separating the traffic in this way makes the performance for a given traffic type more predictable. In networks which allow variable packet sizes such mechanisms are unnecessary but the segregation of the different traffic types is not generally possible.

One thing which few local networks provide is the ability to guarantee a user a given amount of bandwidth or a given access time. The slotted ring lends itself to a mode of operation which is analogous to the Time Division Multiplexing (TDM) used in satellite communications. In TDM a time slot is allocated to a user who has exclusive use of that slot until he no longer requires it. Applied to a slotted ring this means allocating a slot to a source for its exclusive use. This mode of operation will be called **channel mode**. When a slot is not allocated for channel use it will be available for use in the usual way. A problem associated with this style of operation is that in a ring with a small number of slots it is possible that all the slots may get allocated and thus starve other stations of bandwidth. This may be overcome by marking certain slots to indicate that they may be used in this way, all other slots may only be used in **normal mode**. Therefore if at least one normal slot is available there will always be bandwidth available to all stations.

A further problem is the allocation and deallocation of a channel slot. A reasonable approach to allocation is to give a station two types of transmission method, one for normal mode and another for channel mode. Channel mode transmissions may only take place in channel slots and when the first transmission takes place the slot is marked allocated and is subsequently only used by the station which claimed it. Stations must know the slot structure of the ring for this scheme to work. This method of working is practical but potentially wasteful if the slot is used only infrequently during the allocation period. A further problem arises when the slot must

be deallocated, the host could ask the station to do this but some mechanism must also exist in case the host forgets to release the slot. The monitor station could perform this function or it could be provided in all stations. The slot could be deallocated if it had not been used in the last ten ring revolutions, for example.

An alternative scheme is to insist that the transmitter supply data at a rate sufficient to keep the slot filled. In this mode the slot is filled once per ring revolution and so the bandwidth is greater than would be achieved in normal operation. Deallocation is automatic in this method when the transmitting host stops supplying data. Allocation of the channel slot may be done either by explicitly asking to use a channel slot, or by simply transmitting in the normal way and always having data ready when the slot returns. If this latter strategy is used then the transmitter may use a uniform transmission protocol for placing data on the ring and will get a channel slot if one is free.

The channel slot approach partially solves the problem of guaranteeing and partitioning bandwidth. However, it is somewhat inflexible in the amount of bandwidth which is partitioned. Having two or more different slot sizes gives a wider range of choice. A prototype system which embodies the concepts of differing slot sizes and channel mode transmission has been built by Williamson and is described in [Hopper 83]. This system uses the LSI Cambridge Ring implementation to make a ring with two sizes of slots. Short slots contain only 1 byte of data while long slots contain 8 and may be used in channel mode.

Channel mode transmissions provide the highest bandwidth that may be obtained from a slotted ring when slots must return to their source. Still higher bandwidth may be obtained from a slotted ring by allowing a station to have more than one slot in use at once. Imposing a limit on the number of slots that a station may have in use will prevent hogging. The station logic will have to keep track of the number of slots in use and a limit of 2 to 4 seems feasible. Sequencing problems could arise in such systems if data from one slot is rejected, since the station may well have already sent another slot before learning of the rejection. Straying still further from the CR style of working we could have the destination mark slots empty as in the Pierce Loop. This would provide a point-to-point bandwidth equal to the system bandwidth on an unloaded ring but with attendant hogging problems.

A reasonable solution to dynamic bandwidth allocation is the provision of a variety of slot sizes, two being a good start, and the ability to use certain slots in channel mode. Such a system would offer four levels of performance, the two levels using channel mode being stable, the two using normal mode dependent on other traffic on the ring.

## 4.3 ADDRESSING

There are good reasons why a LAN should not be allowed to grow too large. There are also good reasons why we might want a large network. The ability to connect many hosts together is clearly advantageous but the reasons why they should not all be on the same LAN are less obvious.

The hardware of a network must be designed to cope with the maximum size of the network. Ethernet transceivers must be able to drive up to 255 others over 1km of cable. If the Ethernet allowed four times as many hosts on the network, the transceivers would need to be more powerful and hence more expensive. On rings, where the ring delay is dependent on the number of stations, the performance may fall to an unacceptable level if the network grows too large. There may also be instability problems with phase locked clocking systems. There may be administrative problems in dealing with such large numbers of hosts, name translation tables will become large and slow. Reliability too is an issue, placing 200 hosts on a single network assumes a great deal about the reliability of the network. Dividing the network up into autonomous **sub-nets** improves reliability as well as increasing performance on the sub-nets. The data rates achievable between hosts on different sub-nets may be lower than that between hosts on the same sub-net. The division of hosts among sub-nets may therefore have to be done with some prior knowledge of their likely communication requirements.

Thus, while single LANs may not be allowed to grow arbitrarily large, it is possible to interconnect large numbers of hosts by interconnection of LANs, provided a suitable addressing scheme can be devised. This leads naturally to a hierarchy of networks in which a host may be addressed by specifying the sub-net on which it resides and also its address on that sub-net. In the past LANs have been interconnected in an ad hoc manner, the Xerox Internet, for example, is a collection of Ethernets connected together by serial lines running at rates between 2400 and 56000 Bit/S. On each Ethernet host addresses are just 8 bits and so each packet which conforms to the

Internet (PUP) protocol [Boggs 80] carries additional addressing information within the **data field** of the packet. An Internet address is hierarchically structured and consists of an 8 bit net number, an 8 bit host number and a 32 bit socket (port) number. Packets destined for another network are sent explicitly to a gateway host which forwards the packet via other gateways to its destination. The Internet is like a store and forward network but with an Ethernet at each node rather than a single host.

The Universe network [Adams 82] is an interconnection of Cambridge Rings by satellite and high speed land lines. Here again a hierarchical address scheme is used, the base networks having no provision for interconnection. Newer networks, such as IBM's token ring network, include provision for interconnection of rings by providing a large address with fields for both ring and host numbers.

A characteristic of all interconnected LANs is the existence of a gateway on each sub-net, which is responsible for receiving and passing on packets destined for other sub-nets. There are two extremes of LAN interconnection. At one end of the scale is the Xerox Internet, in which no attempt is made to keep performance the same for traffic within an Ethernet and between connected Ethernets. The interconnection is over long distances, typically hundreds of miles, and is largely a convenience rather than a necessity. At the other end of the scale is the IBM ring where the interconnection of rings is intended to be transparent to the hosts and there may be functional dependence between hosts on different rings. The distances here are small, a few miles at most between rings. Between the two extremes lies the Universe network in which the bandwidth available between two rings is high, around 1MBit/S over the satellite channel, and the distances between rings large.

The introduction of high speed land lines, such as British Telecom's Megastream, which offer bandwidths of up to 1MBit/S over distances of hundreds of miles, means that widely distributed connections of LANs will be possible which offer performance similar to that seen on single LANs.

When designing a new network it would seem sensible to allow for the possibility of connecting networks together, either over quite short distances or over longer ones with a consequent increase in delay. There are a number of ways of organising the addressing mechanism to allow packets to pass between sub-nets rapidly and so avoid the store and forward effects which reduce performance.

One method of organising addresses is in a hierarchical arrangement so that the bits of the address are split into fields, for sub-net and host number for example. Pierce's nationwide loop employed addresses of this form but with a three level hierarchy, IBM's ring uses a two level hierarchy. An alternative is to simply have a flat address space where hosts are numbered without regard for their position in the network hierarchy (if any). There are other schemes which might be employed such as Saltzer's source routing method [Saltzer 80] but the two already mentioned are the most suitable for incorporation in a new design.

The hierarchical scheme has the advantage that no central address allocator is necessary, addresses on a sub-net may be freely assigned without conflicting with those on any other sub-net. The fields of the address may also make routing packets particularly simple in some networks. There are also disadvantages to this scheme, the relative sizes of the fields must be decided at the outset and may prove unsuitable in certain cases. Making the address large will alleviate this problem. Another drawback is that if a resource moves its physical location between sub-nets then its address must be changed.

A flat address space requires that addresses be allocated rather more carefully if several agents are allocating them. However, there is nothing to stop conventions being applied to the flat address to form fields for the purposes of allocation or whatever. Moving resources in a flat address space means that the address may move with the resource. The principal difference between the two schemes is that the flat address scheme requires the whole address to be inspected to perform routing, while the hierarchical scheme requires just part of the address to be looked at. The hierarchical scheme may impose restrictions on the topology of interconnection which a flat space would not.

### 4.3.1 Broadcast Addressing

Broadcast addressing is a means of addressing a single packet to all hosts on a network and many current local networks have this facility. Generally a special address is allocated for the purpose and when this address is used as a destination all stations may receive the packet. This can be a very useful facility, particularly for resource location in a distributed environment. For example, it could be used to allow a newly switched on host to locate a name server. When LANs are interconnected a decision must be made whether or not to allow broadcast data to spread over the

whole interconnection. If there are loops it is possible that the broadcast packet may be propagated indefinitely throughout the system and steps would have to be taken to avoid this.

### 4.3.2 Source Selection

Source selection allows a host to receive packets from selected sources only. The Cambridge Ring allows no sources, all sources, or one particular source to be selected by means of a register which the host may update. A useful extension to this might be to allow a certain group of sources to be selected by means of a table lookup on source addresses. A possible use of this mechanism is to mask troublesome sources which send unwanted messages. A problem with the masking of a source in this way is that it is difficult to know when to start listening to it again. A timeout mechanism would solve the problem but would be complex to administer.

## 4.4 ERROR DETECTION AND CORRECTION

The fundamental operation performed by most local area networks is the transfer of blocks of memory from one computer to another and there are many stages in this process at which errors might occur, the network being just one of them. The most effective means of checking is thus on a memory to memory basis, a checksum could accompany each data block and be checked when the data is in the destination host's memory. In some circumstances this may be too much trouble to perform and a less thorough method is preferable, such as checksumming individual packets as they are received.

The network should do its best to detect errors which arise during transmission and report them to the appropriate authority. At this level we can distinguish between two types of network error. Both types may have a common cause but their implications are rather different. The first type of errors are those which occur in the data portion of a packet and the second are those occurring in the control portion of the packet. The latter may have implications both for delivery of the packet and for correct operation of the network. Corruption of a destination address may result in a packet being wrongly delivered and further packets to the true destination being out of sequence.

Local area networks are usually constructed of transmission media with a low error rate and the error checking and protocol systems will be designed with this in mind. Errors on the network will occur however and it is thus common to have some sort of error checking field in packets. This is often a Cyclic Redundancy Checksum (CRC) of some or all fields in the packet. Whilst it may be useful to know that the data field has been corrupted at the network level, it is rather more important to know that control information is in error. Since the CRC contributes to the overhead in a packet we wish to keep it as small as possible, while still providing the level of error detection that we require. Using the CRC to check just the control fields may be a useful strategy, provided we are confident that adequate checking of the data occurs at higher levels.

In all networks we wish to check packets when they arrive at their destination and refuse to accept them if they appear to contain an error. In loop networks where packets return to their source carrying a response we also wish to have confidence that the response is correct. It may be possible to mark the response to indicate that the packet was rejected due to error and thus facilitate a low level retransmission method. The action taken by a destination on receiving a packet containing an error may vary according to the protocol in use. The data will not be used but a number of courses of action might be taken. Reporting the error to the source might be useful in some cases but there is the possibility that the source address field may be incorrect. Reporting the error to some central logger could be useful and aid maintenance of the network.

In certain types of ring the packet may be altered by various legitimate means on its way around the ring and recomputation of the error checking field will be necessary. Cambridge Ring minipackets may be altered at the monitor station and also at the destination station where the response bits are marked. In such systems it may be difficult or expensive to recompute the error check field and parts of the packet will have to go unprotected. In the case of response bits this is particularly unfortunate and means that sequence checking of packets is essential if packets are not to be lost or duplicated due to errors in the response bits. Alternatively a simpler error detection mechanism such as parity may be used to guard a small number of bits with some confidence. Having responses means that a destination can signal its rejection of a packet due to error back to the source, which can then retransmit the packet. This prevents having to pass the error up to a higher level and thus is more efficient.

### 4.4.1 Maintenance

Another important aspect of error detection is that it enables a maintenance system to be constructed. Whilst a station receiving a bad packet might not report it to the source it would be useful if the error were reported to a centralised logger, where the information would be available to maintenance staff. In this way an overall picture of the error rate of the network could be built up and parts of the network which are particularly error prone identified. Faulty components may be isolated and links which pass close to sources of electro-magnetic interference located. In loop networks each station on the loop has the potential to look at all packets passing through it and report any errors it sees. If each error message contains its source address then it will be possible to localise the source of the error to a single link. The Cambridge Ring performs such a function by means of a parity bit in each slot which is checked and corrected at each station. Such a mechanism also locates a broken ring cable since a stream of error messages will be generated by the station after the break. This method of testing is most effective in systems such as the slotted ring where all slots, whether full or empty, can be checked and the network is therefore under continuous test.

When deciding on a suitable error detection scheme for a LAN it is necessary to have some idea of the types of error which are likely to occur and their frequency. Error rates for transmission systems used in LANs are frequently quoted as being in the range 1 bit in $10^9$ to 1 in $10^{12}$ but little work has been done in finding out just what types of errors actually occur. Any transmission system has an associated signal to noise ratio. The noise comes from random thermal effects in the transmission and reception circuitry and is the source of a basic level of errors. External phenomena, such as Electro-magnetic Interference (EMI), will cause additional errors. EMI is generated by the switching of large currents in the vicinity of LAN cables. Fibre-optic cables are unaffected by such phenomena but the very low signal levels encountered in fibre-optic receivers mean that they are much more susceptible to EMI than their wire cable counterparts. Whilst thermal noise may cause corruption of 1 or possibly 2 bits, EMI will generally persist for anything from a few microseconds to a few milliseconds and many bits may be corrupted in what is called a burst error.

Another kind of error occurs in systems which use a phase locked loop (PLL) for clock recovery and jitter removal. If the degree of jitter is high the PLL may skip a bit time either backwards or forwards, with the result that a bit may be lost or a random bit inserted into the data. All subsequent bits in the data are thus shifted by one place and this amounts to a long burst error.

Error detection must thus cope with a wide range of errors and an effective check is a CRC. This is a fixed size check field and is easy to compute yet detects a wide variety of errors. A single parity bit is very easy to implement but is incapable of detecting all errors. It may be useful in a maintenance system where it is not necessary to detect all errors, but just sufficient to allow the maintenance mechanism to function. A parity bit applied more frequently, say to every byte in a packet, would be a much stronger check for long burst errors and could possibly be propagated throughout the entire transmission process, giving an end-to-end check on the transfer. The overhead in doing this is significant, especially since all data paths in the system would need to be enlarged to hold the extra bit.

## 4.5 RELIABILITY

Local networks are vulnerable to failure in many ways. An Ethernet will fail if the cable breaks, since the resulting segments will be incorrectly terminated. Certain modes of failure of an Ethernet transceiver may cause it to jam other transmissions. A Cambridge Ring will fail if a link is broken or if a repeater malfunctions. Failure within a station may prevent a particular host communicating via the network, while failure of a host such as a name-server may prevent other hosts from functioning correctly. Thus, as with error handling, there are many levels to be considered when discussing reliability.

At all levels there are two complementary approaches to improving reliability, partitioning and redundancy. Partitioning involves dividing the network into sub-units which will be unaffected by the failure of one of their number. Redundancy implies the provision of alternative resources to be used in the event of failure, two name-servers, for instance, or two network interfaces for a host or two rings. Providing redundancy means that some means of detecting failure must be provided. Systems vary in their requirements for switching to backup equipment, in some it should be performed automatically, in others it may be sufficient to inform an operator.

Practical work on redundancy techniques for LAN hardware have so far only been performed on ring systems. Pierce suggested redundant cabling techniques to allow a loop to survive certain types of cable break. Recently, rings have been constructed with dual cables which are able to survive a cable break or repeater failure by using a "loop back" method to reconfigure the ring without including the broken element. Racal Milgo's Planet network uses this method.

Closely allied to reliability is the maintenance system of the network. It should detect and report equipment failure and take any corrective action that the network is capable of. It should also make the location of faulty equipment as easy as possible. Making a network sufficiently redundant to survive cable failure, station failure and network monitor failure is a very complex process and is only attempted in applications which demand exceptional operational reliability such as aircraft control systems and process control applications.

Reliability may also be enhanced by the use of high quality components and high standards of construction in the network. Reducing the component count, for example by implementing in LSI circuits, will also improve reliability. Practical experience has shown, however, that the main causes of failure of LANs are events such as lightning striking part of the network or accidental cutting of the cables.

A general purpose network will use partitioning to isolate faulty sub-nets until they can be repaired. Partitioning implies that a suitable addressing mechanism exists on the network and that bridges between the sub-nets can be constructed.

## 4.6 COST

It is frequently the case that the major cost in installing a local area network is that of cabling. Running cables around a building or campus is a labour intensive, disruptive and hence expensive process. There is still a good case for reducing the cost of network interface devices and when this is done the overall network cost will fall. It will also be economically sound to connect cheap computers to the network which would not have been connected were the cost of connection higher.

The circuitry which is required to interface a computer or similar digital device to a network generally divides into two parts, termed the station and access box. All stations on a network are very similar, while the access box is specific to a certain type of computer system. The function of the access box is to interface the computer to

the station and can vary in complexity from a few gates and latches to a large microprocessor controlled system. The station is the obvious first candidate for cost reduction and this topic is now discussed further.

The only way to significantly reduce the cost of digital logic systems which run at high speeds and are traditionally constructed in MSI logic is to implement them in LSI. Until recently only the large semiconductor manufacturers were able to design and build such circuits. Methods of simplifying the design process have been developed and many manufacturers now undertake the fabrication of LSI circuits from designs supplied by clients. Two methods of simplifying the design process are prevalent, uncommitted logic arrays (ULA) and semi-custom cell based logic. ULAs are silicon chips containing a regular arrangement of devices such as gates or flip-flops. The designer specifies an interconnection pattern for these devices which the manufacturer can then place on the chip, using one or two layers of metallisation. Semi-custom design involves the designer using a small set of predefined circuit elements such as latches, multiplexers and gates to realise his circuit. The cells on the chip which perform these functions are of regular shapes with predefined contact points so that they will tile together on the chip without wasting space.

Development cost for a ULA design is lower than for a semi-custom design but the cost per chip is higher and so ULAs are suited to low volume production of a few thousand chips and semi-custom to higher volumes of tens of thousands. At present only Metal Oxide Semiconductor (MOS) chips can be produced in semi-custom while faster bipolar chips must be made as ULAs.

The design of a LAN station in LSI will involve considerations of speed, power consumption and the size and complexity of the circuit. The design may require more than one chip in which case the partitioning of the logic amongst the various chips must be decided. A single chip implementation is likely to be cheaper than a multi-chip one and so it is desirable to try to fit the whole design onto a single chip. The loss of some features of the design may be considered worthwhile in order to do this. Pin count is also a consideration. Chips with many pins are expensive, yet a LAN station will probably require a parallel interface of at least 8 bits and we can expect to need at least 40 pins and possibly more. An early LSI implementation of the Cambridge Ring required 1000 gates and was implemented using two 40 pin bipolar ULAs. Advances in technology now allow several thousand gates on a ULA and semi-custom designs are limited in size only by the area of the chip, 10000 gates being an economic

maximum.

In many chip technologies power consumption is related to the actual or expected maximum speed of operation and if several chips are being used it may be possible to place all the high speed logic on the same chip. Part of a LAN station will have to work at the clocking rate of the network and this is likely to be the highest speed encountered. It may be possible to divide the clock rate in other parts of the logic and handle data in parallel, an obvious choice being to divide the clock by 8 and deal with bytes of data. This approach has several advantages. If power consumption is related to clocking rate then the overall consumption will be reduced and if the whole design is on the same chip then timing tolerances will be increased in the slower parts, making the design process easier and operation more reliable.

Rough estimation suggests that a station will require between 1000 and 5000 gates to implement the control logic and if a clock division method is used no more than about 10% of the logic will have to function at full speed. With a slotted ring, where the amount of data carried in each slot is quite small, we may also consider providing transmit and receive buffers on the chip. We might expect buffer sizes in the range 1 to 100 bytes, which would require up to several thousand more gates, bringing the total requirement to a maximum of around 10000 gates. The provision of buffers on the chip may be unwarranted as the silicon area of the chip and hence its cost will be increased. Buffers are just memories and readily available at low cost in a variety of forms.

Considering now the integration of a station for a slotted ring running at 100MHz, it has been found by computer simulation that gate delays of not much more than 1nS will be required in those parts of the circuit clocked at 100MHz. Gates as fast as this are only available on bipolar ULAs and the largest ULAs which will run at this speed have about 2500 gates. Generally only about 70% of the array may be utilised and so only about 1800 gates will be available for use in the design. Larger ULAs are available but at the cost of increased gate delays. With gate delays of 10nS we can consider using MOS processes and semi-custom design. An alternative strategy is therefore to use a small fast bipolar chip to interface to the ring's serial data stream and divide the clock of this device to a level suitable for use with a slower and much larger chip. A parallel interface would be necessary between the two and this might cause problems in that the pin count of the slower chip could be high.

Thus a slotted ring station might be approached in one of two ways, placing the entire circuit on a single fast chip might restrict the complexity and hence the number of features that could be provided. Using a small fast chip in conjunction with a large slow one would be more expensive but allow a much more complex system to be implemented.

Chapter 5

# THE CAMBRIDGE FAST RING, MARK 1

This chapter describes a high speed slotted ring designed in the Computer Laboratory at Cambridge in 1981 and 1982. The design was influenced heavily by the ideas described in the previous chapter, in particular the method of bandwidth partitioning by having different types and sizes of slots was used. The design is referred to here as the Cambridge Fast Ring, Mark 1 (CFR/1) to distinguish it from a later version.

## 5.1 OVERVIEW

CFR/1 is made up of three kinds of nodes, **stations**, which transfer data between devices attached to the ring, **bridges**, which copy minipackets between rings and **monitors**, one of which is required on each physical ring to set up and maintain the slot structure. Additionally, each ring may require an error logger which is a normal station configured to receive maintenance minipackets. The monitor can be used as a simple error logger if necessary. There is a global address structure with which it is possible to make arbitrary connections of rings such as that shown in Figure 5.1a.

The slots of CFR/1 are divided into a number of categories according to their length and the way they can be used for transmissions. Slots may have one of two lengths, short slots which have a data field of 8 bytes and long slots which have a data field of 8, 16, 32 or 56 bytes. Each node on the ring has a ROM connected to it which supplies configuring information. The ROM tells the node whether it is a bridge, station or monitor and supplies other data such as the station address and the length of long slots. Although it is possible to dynamically change the length of long slots this would not be done in most ring implementations. Slots form a train round the ring with a **gap** to fill out the excess delay. Transmissions can be made in the normal CR way with each slot being passed on after use (**normal mode**), or they can be made in **channel mode** where once a slot has been marked full it can be replenished for an indefinite number of revolutions. The implementor of a particular CFR/1 system can choose which slots may be used in this way and which may only be used for normal mode transmissions. A typical system might have a single long channel mode slot for bulk

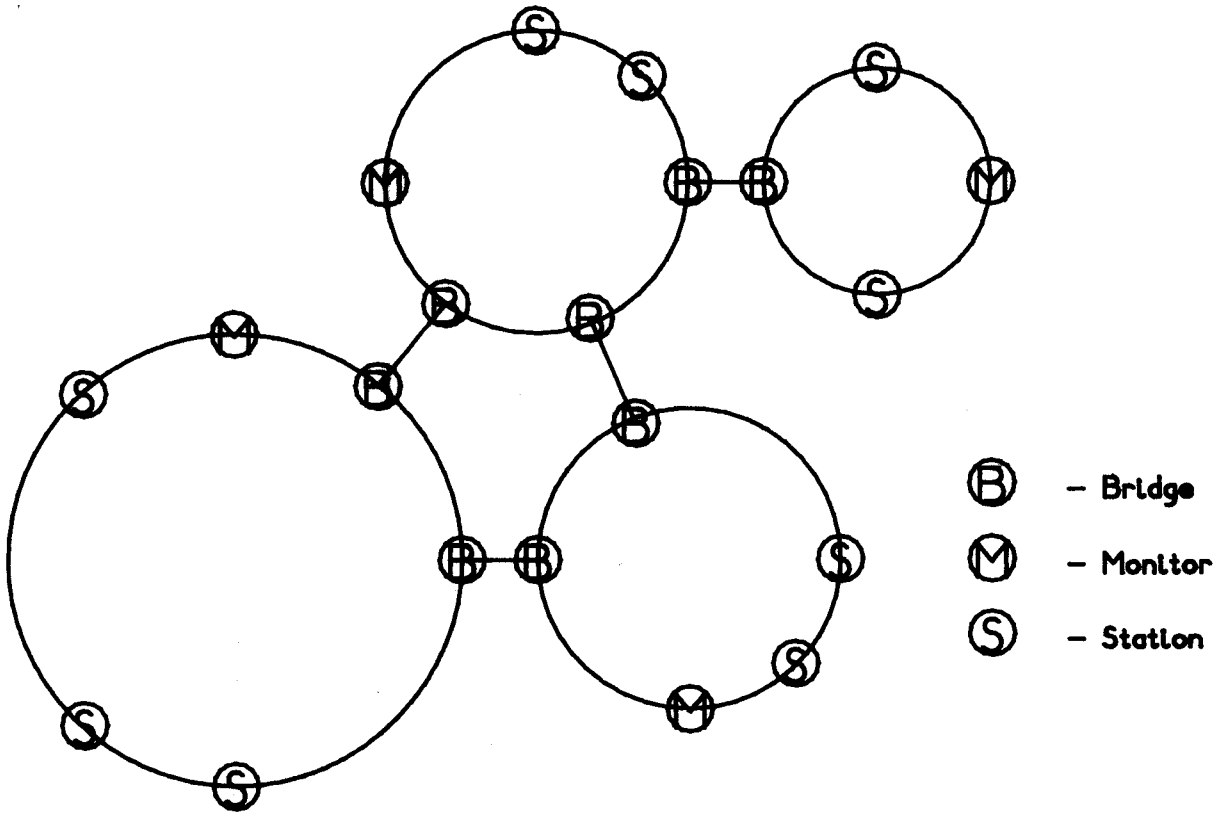



*Figure 5.1a Interconnection of CFR/1 to form a complex network*

data transfers and a number of short normal slots to guarantee that some bandwidth will always be available regardless of the usage of the channel slot. The minipacket format is shown in Figure 5.1b.

At the byte level each minipacket consists of a start byte followed by a 2 byte (16 bit) destination address and a 2 byte source address. Now follow 8, 16, 32 or 56 bytes of data, a CRC error check byte and an end byte. The CRC is computed from the address and data fields, but not the start or end byte.

The start byte consists of two start bits which are always set to one. These are followed by two bits which indicate the slot type. The first of these bits indicates whether the slot is short or long and the second indicates whether the slot can be used for channel mode transmissions as well as normal transmissions. The full/empty bit indicates whether or not the slot is being used. The monitor passed bit is set by the source of a transmission and cleared by the monitor. Its purpose is to detect and correct corruptions of the full/empty bit, which would otherwise cause the system to fail. The last two bits of the start byte are used to indicate a slot that is being used in

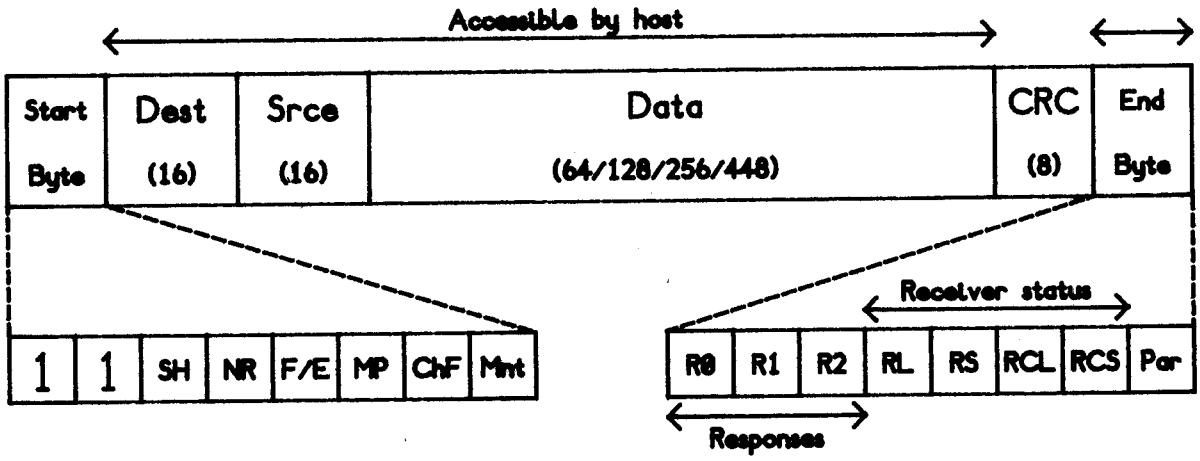



*Figure 5.1b Format of a CFR/1 minipacket*

channel mode and a slot being used as a maintenance minipacket. The first of these is required because a channel mode slot can be used for normal transmission and the receiver has to be able to discover what kind of minipacket has been received.

The end byte is used for responses, maintenance information, and the maintenance parity check. A CFR/1 minipacket is marked at the destination with a number of responses which can be read by the sender of the minipacket when it returns to its source. These responses, which are encoded in three bits, indicate how the minipacket was handled at the receiver. In addition four other bits are used to indicate to the transmitter which types of minipackets the receiver is prepared to accept. Because minipackets can be of four types (short/long, normal/channel) and a receiver can be configured to receive any type or any combination of types, four bits are required for this purpose. These bits are used differently in a maintenance minipacket where they indicate the type(s) of error which caused the minipacket to be sent. The parity bit is the parity of all bits in the minipacket and is checked and corrected at each node on the ring. If it is ever found to be incorrect a maintenance minipacket is sent from the node which detected the error to an error logging station. This provides a valuable way of identifying links which are error prone.

## 5.2 ADDRESSING

CFR/1 uses unstructured (flat) 16 bit addresses. The addressing mechanisms built into the node hardware allow local transmissions to occur and also cater for minipackets directed to nodes not on the ring on which they were transmitted. To allow for this, certain nodes can be configured as bridges and these nodes are able to recognise which minipackets should be received by the bridge and retransmitted on another ring.

The destination address 65535 (all bits of the address are one) is the broadcast address. Minipackets sent to this address may potentially be received by all stations on the network.

### 5.2.1 Address Filtering.

Each station of CFR/1 contains a 16 bit **select register** and a 64K bit **select map** which are used when receiving minipackets. The select register and map contents are set by the host. Selection operates by looking at the source address field of arriving minipackets. When the select register contains zero the station rejects all minipackets addressed to it and such minipackets are marked with the response **unselected by register**. When the select register contains a station address, minipackets from all addresses other than that in the register will be rejected with the unselected by register response. When all the bits of the select register are set to one then the source addresses of minipackets addressed to the station are used to index the map. If the map bit is zero the minipacket will be rejected with the response **unselected by map**. If the bit is one the minipacket is eligible for reception. The mechanisms outlined above allow a receiver to choose from whom to receive minipackets; no stations, just one station or a group of stations. It is expected that, initially, the map will be set to allow reception from all addresses and that certain addresses will be marked to prevent reception as time goes by.

### 5.2.2 Inter-Ring Addressing.

Nodes which are configured as bridges use the map in a rather different way to that described above. At a bridge the map is indexed by the destination address of passing minipackets. Thus, for a given address, the map bit should reflect whether the address can be reached through this bridge or not. The bridge itself does not have an address and thus cannot be transmitted to directly. In some implementations it may

be configured as a low level hardware bridge with the map in ROM. The bridge can be processor controlled, however, to allow the map to be configured when the network starts up. In this case the processor must be connected to the network via its own station.

Care must be taken with broadcast minipackets in networks which contain a circular path through bridges. If a broadcast minipacket follows such a path it will circulate forever. The bridge maps can be configured to constrain broadcast minipackets to selected areas of the network and thus avoid this problem.

## 5.3 HARDWARE IMPLEMENTATION

The physical implementation of the CFR/1 evolved through a number of stages. To minimise chip design time it was decided to use gate array (ULA) technology. The most suitable gate array which was then available contained 2400 gates, each with a best case propagation delay of just over 1nS. This meant that it was not possible to consider integrating the complete system onto a single chip. The node logic was therefore partitioned into four sections:-

1) the line drivers, receivers and clocking circuitry

2) the transmit and receive buffers

3) logic implementing the network control functions

4) the map (a 64K dynamic RAM chip)

With this partitioning the network control logic was designed and found to consist of about 2700 gates. Given that only about 1800 of the 2400 gates on the ULA could be expected to be utilised, a further division of the network control logic was made. The division was such that two chips were used. One chip implemented the main part of the logic and allowed transmissions and receptions to take place, while a second chip implemented the address filtering, that is the select register and map handling. Figure 5.3a shows a typical station configuration.

It was intended that simple systems would be able to dispense with the facilities provided by the second chip thus keeping the cost down. One of the major constraints in the design was the number of pins available on the chips. Both chips are housed in 68 pin chip carriers and some multiplexing of pin function is necessary in the main chip.

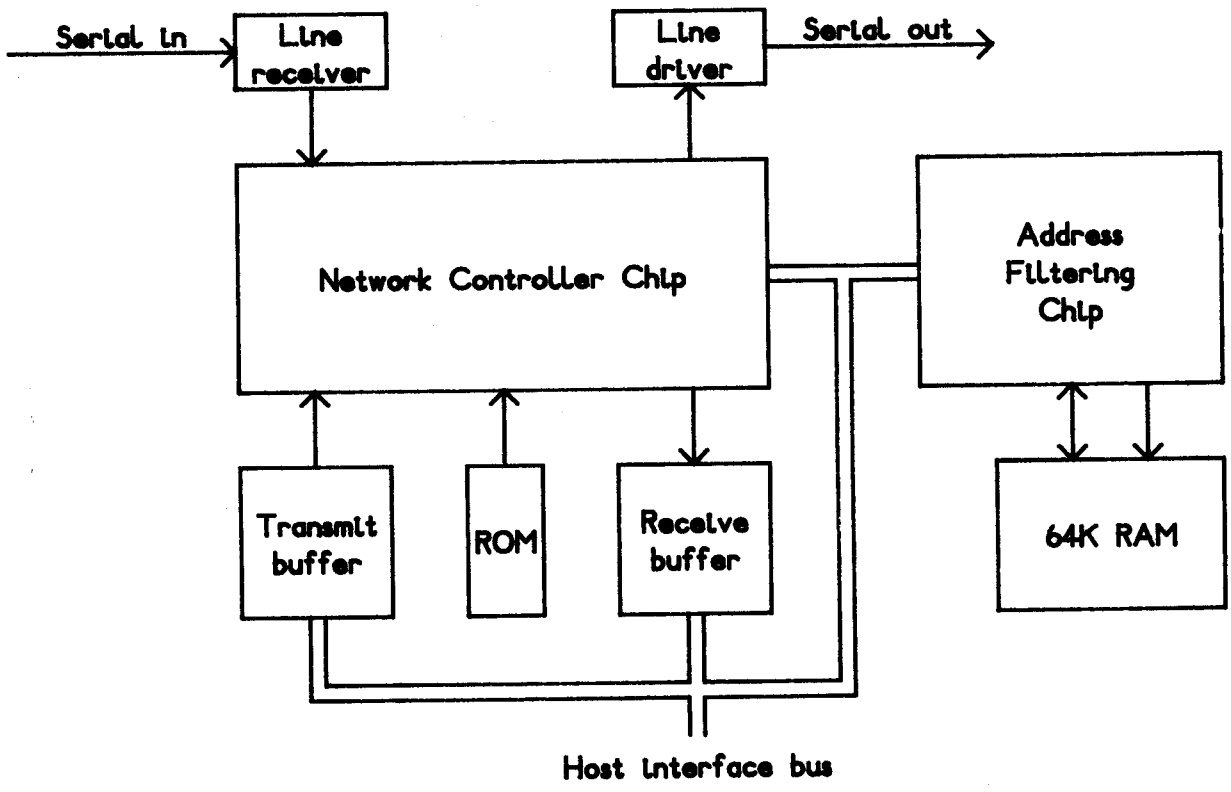



*Figure 5.3a Station configuration of CFR/1*

Most of the logic required to implement CFR/1 nodes is on the Network Controller Chip (NCC). This chip accepts serial data from the inter-node link and allows transmission and reception of minipackets. The NCC delays data passing through it by 24 bits. This delay and the delay in the inter-station links make up the total ring delay. The transmit and receive logics are separate and duplex communication is possible. The NCC is configured by reading 4 bytes of data into the chip, typically from a small ROM. This data is the node address and configuring information for the node and has the following format:-

Byte 0 - node address (high byte)

Byte 1 - node address (low byte)

Byte 2 - configuring byte as follows

| | |
|---|---|
| Bit 0, 1 | - code defining length of long minipackets |
| Bit 2 | - set if node is to be a bridge |
| Bit 3 | - set if node is to be a monitor |
| Bit 4 | - enables delaying of responses |
| Bit 5 | - enables transmit on accepted mode |
| Bit 6 | - enables sending of maintenance minipackets |
| Bit 7 | - enables CRC checking and response marking |

Byte 3 - configuring byte as follows

| | |
|---|---|
| Bit 0 | - chip operates with 2 bit parallel data when set |
| Bit 1 | - line data is input in serial/modulated form |
| Bit 2 | - line data is output in serial/modulated form |
| Bits 3-7 | - unused |

The select register and map handling are implemented in the Address Filtering Chip (AFC). The map is implemented using a 64K dynamic RAM chip and functions as the select map at stations and the bridge map at bridges. All timing for reading, writing and refreshing the RAM is provided by the AFC. The RAM is initialised by the passing of the first 64K slots after the AFC powers up. The AFC interfaces with the NCC to get the address fields of minipackets and indicate the result of the address filtering operation. The AFC is clocked at one eighth of the speed of the NCC and so can be implemented in a slower logic family.

While it had been hoped that the CFR would be clocked at 100MHz, simulation showed that a speed of about 75MHz would be the maximum at which the NCC would function.

### 5.3.1 Line driving system.

One of the desired characteristics of the line driving system was generality. A series of experiments was performed by Banerjee [Banerjee 83] which showed that high quality twisted pair cable could be be used for data transmission at up to 100MBit/S, using the Cambridge Ring modulation scheme. Fibre-optic systems of low cost are not yet available to run at this speed, but it is likely that they will be in the future. Thus the line driving system provides for a variety of options to accommodate a range of line types.

The primary line driving mode of the NCC is the CR delay modulation system, modulated data is input on two input pins and output on two output pins. There is also a clock input and a phase comparison output. The latter compares the earlier of the two data edges with the active clock edge and drives a phase locked loop to enable clock recovery from the data signals. The implementor must therefore provide line drivers, line receivers and a phase locked loop to complete the line driving system.

In order to allow other modulation systems to be used it is possible to disable the modulator and demodulator and instead use only one input and one output pin for unmodulated data. This may be particularly useful where the system is being used with a transmission medium such as fibre optics. In this mode the system implementor is required to provide the clock as well as the data to the system and thus must provide some kind of clock regeneration mechanism.

In order to improve tolerances and possibly operate at still higher speeds (over 100MBit/S), the NCC can also be operated in a two bit parallel mode. Again, the designer has to provide an external modulation and clocking system. However, with this option the chip operates completely in a two bit parallel mode and the external clock runs at half the speed of the serial data rate on the line.

### 5.3.2 Transmit and receive buffers.

The transmit and receive buffers are used to buffer a single minipacket at a node. The NCC interfaces to the buffers over two 8 bit busses and provides some control signals for standard FIFO memories which implement the buffers. For transmission the buffer must contain the destination address as well as the minipacket data and thus requires between 10 and 58 bytes of storage.

The receive buffer is similar but will contain the source address instead of the destination. Because many successive transmissions may be made to one destination, it is beneficial if an option is provided to allow an address to be loaded once and then used in every transmitted minipacket until changed. On the receive side there is a similar situation where we wish to receive many minipackets from one source only. If the source address had to be read from the FIFO for each arriving minipacket, implementation of simple Direct Memory Access (DMA) logic would be more difficult. In both the transmit and receive cases a timing signal is provided which can be used to gate the appropriate clock to another buffer which holds the address. This buffer can be implemented as a FIFO or register and can be 8 or 16 bits wide. To allow a range of

widths for the host interface, a two bit byte counter is provided to route data to up to 4 byte wide FIFOs, thus allowing for 8, 16 or 32 bit bus structures. When an inter-ring minipacket is being received or transmitted at a bridge it is necessary to buffer both address fields and the CRC in addition to the data field and larger buffers will be required in this case.

## 5.4 TRANSMISSION AND RECEPTION

As mentioned earlier there are a number of different transmission and reception modes available on CFR/1. The standard CR method of transmission is provided as is a similar mode known as **transmit on accepted** (TOA). This gives a slight performance improvement over normal mode but requires a double transmit buffer. It works by preloading a second minipacket into the second buffer while a first is in transit around the ring. If the first minipacket returns accepted then the second will be transmitted automatically. A signal from the NCC (TOAOK) indicates that this has occurred. If the first minipacket was not accepted then the host will inspect the responses and possibly retransmit it.

Channel mode transmission is also available in which the sender must keep reloading the transmit buffer sufficiently fast to allow transmission once per revolution in the same slot. Minipackets may also be broadcast in any mode, a special address being reserved for this purpose. When not in channel mode the NCC can be programmed to automatically retransmit a minipacket up to 16 times if it is rejected due to a CRC error or the destination being busy, this is the **repeat on busy** option. When transmission is taking place two NCC pins indicate the state of transmission, FRESP (fast response) becomes active when a minipacket is transmitted and SRESP (slow response) becomes active when the minipacket returns and the responses are available. Transmission is controlled by a set of registers and pins of the NCC as shown in Figure 5.4a.

Access to the registers is controlled by pins TR0 and TR1. The transmit control register (TCR) is used to determine whether a long or short slot should be used and whether a normal or channel mode transmission is to take place.

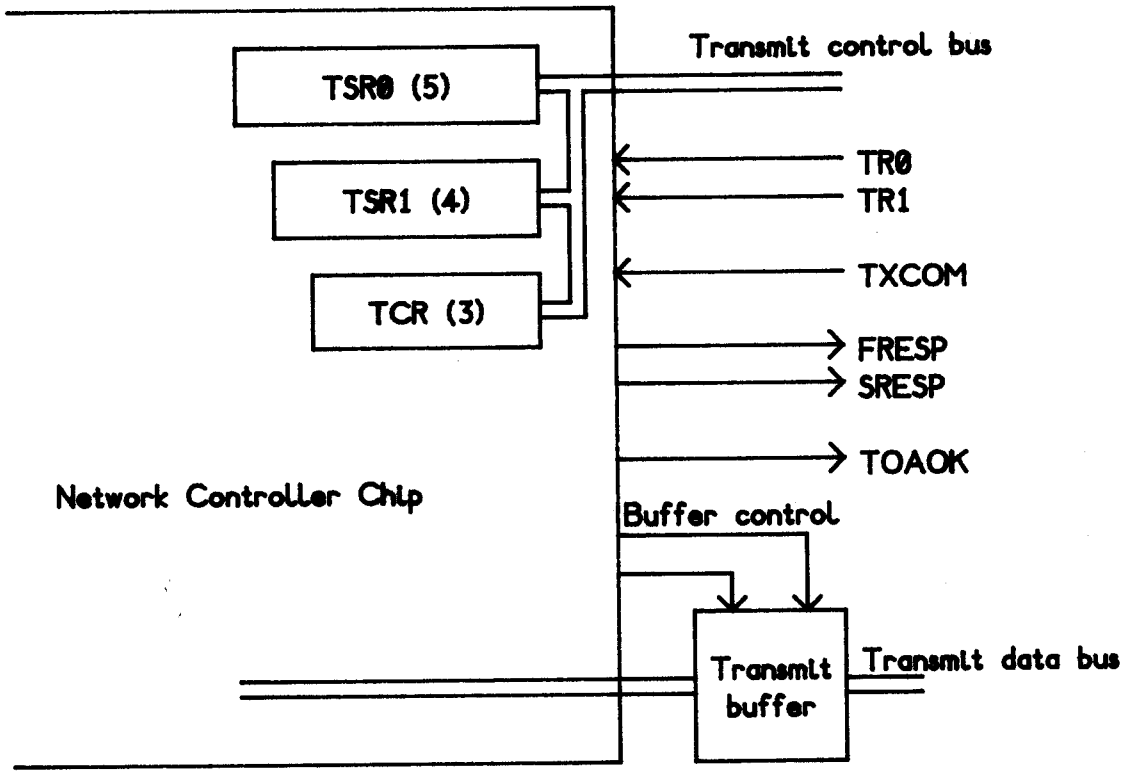



*Figure 5.4a* ***Transmit subsystem of CFR/1***

The format of the TCR is as follows:-

Bit 0 - transmit in long/short slot

Bit 1 - transmit in normal/channel slot

Bit 2 - enable repeat on busy option

There are two status registers associated with transmission, the first (TSR0) holds responses from the destination of the last transmission and the second (TSR1) holds a copy of part of the destination's receive control register (RCR).

The format of TSR0 is as follows:-

Bit 0 - response bit 0 (R0)

Bit 1 - response bit 1 (R1)

Bit 2 - response bit 2 (R2)

Bit 3 - set when a failure occurs in TOA mode
(reset when TSR0 is read)

Bit 4 - set if CRC of returned minipacket was incorrect

The RCR controls which sorts of transmissions a station is prepared to receive and so is useful to a transmitter to enable it to send data in the appropriate slot type. The RCR contents and responses are copied from the destination in the last byte of the minipacket. The receive control logic is similar in design to that of the transmit side. There is a control register (RCR) and two status registers (RSR0 and RSR1) as well as a number of control lines as shown in Figure 5.4b.

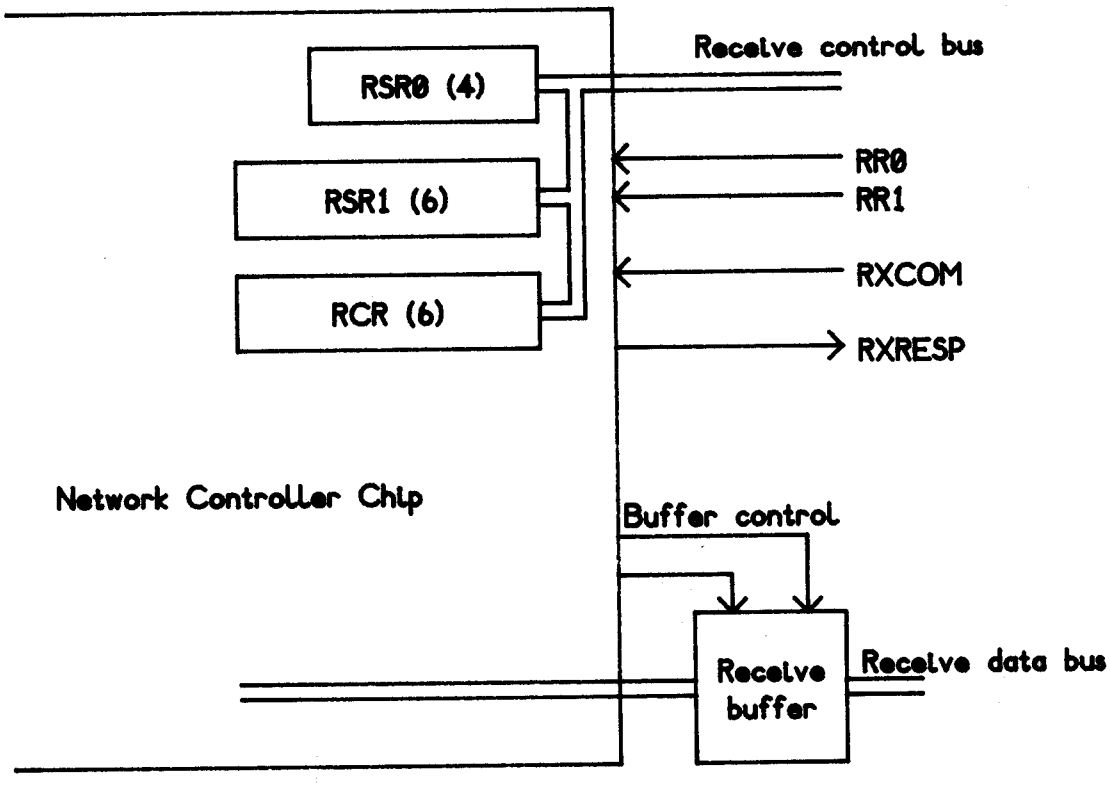



*Figure 5.4b Receive subsystem of CFR/1*

The two lines RR0 and RR1 control access to the registers. The control register contains six bits which are used to enable the reception of various types of minipacket, the first four are copied back to the source of arriving minipackets.

The format of the RCR is as follows:-

Bit 0 - enable reception of short normal minipackets

Bit 1 - enable reception of long normal minipackets

Bit 2 - enable reception of short channel minipackets

Bit 3 - enable reception of long channel minipackets

Bit 4 - enable reception of broadcast minipackets

Bit 5 - enable reception of maintenance minipackets

RSR0 indicates what kind of minipacket has been received and RSR1 is used when the station is configured to receive maintenance minipackets, to indicate the type of error that the minipacket is reporting.

The format of RSR0 is as follows:-

Bit 0 - set for a broadcast minipacket

Bit 1 - set for a maintenance minipacket

Bit 2 - received minipacket was short/long

Bit 3 - received minipacket was normal/channel

### 5.4.1 Responses

The response bits indicate to the source of a minipacket how it was handled at the destination. Only the **accepted** response indicates that the minipacket was received, all the others indicate an error condition of some sort. Certain errors can be expected to disappear if the minipacket is retransmitted (**busy** and **CRC errors**), while the rest indicate a more serious fault and imply that a change of strategy is necessary at the source. Responses have an associated priority such that if more than one response is applicable at the destination, then the one with the higher priority will be marked. The responses are as follows:-

| Response | R0 | R1 | R2 | Priority | Meaning |
|---|---|---|---|---|---|
| Accepted | 1 | 0 | 0 | 0 | Minipacket was received at destination |
| User fault | 1 | 1 | 0 | 1 | Destination host rejected minipacket |
| CRC error | 1 | 0 | 1 | 2 | CRC was found incorrect at destination |
| Busy | 0 | 1 | 0 | 3 | Destination receive buffer full |
| Wrong type | 0 | 0 | 1 | 4 | Destination unwilling to receive this type of minipacket |
| Unsel by Reg | 0 | 1 | 1 | 5 | Select register forbade reception |
| Unsel by Map | 1 | 1 | 1 | 6 | Map forbade reception |
| Ignored | 0 | 0 | 0 | 7 | Destination address not recognised by any node in the system |

**Wrong type** indicates that although the destination is willing to receive from the source, a minipacket of the wrong type has been sent. For instance, a long minipacket being sent when the destination is only willing to receive short minipackets. Because the destination's receiver control register is copied back to the source, the source may take appropriate action for this response. An **ignored** response may mean that an erroneous address was used or that the destination is powered down. In addition to the responses generated by the NCC and AFC, an input pin on the NCC allows the user to force a **user fault**. This may be useful where sequence numbers or some higher level protocol functions are performed in hardware which can effectively abort reception very quickly.

The responses for transmissions in channel mode and broadcast mode have different meaning because the response field may be overwritten several times. In channel mode the responses will be read at the end of a number of transmissions and it will be possible to tell if all minipackets were received, but not the fate of individual ones. For broadcast transmissions the minipacket will make a single revolution but every station will mark the responses. Again, we can discover if everyone accepted the minipacket or if no one saw it but we cannot tell what happened at each station.

## 5.5 ERROR CHECKING AND MAINTENANCE

CFR/1 separates the functions of error checking at the data level and of line level maintenance, primarily to reduce the complexity of the NCC. The maintenance mechanism uses a single parity bit at the rear of the minipacket to detect corruptions, while the data checking is performed by an 8 bit CRC which checks those bytes in the minipacket which do not change during transmission. The division of function comes about because recomputation of the CRC at the monitor and destination station would increase the size of the NCC quite seriously, whilst parity checking is quite easy to implement. The maintenance function is hidden from the user's view and this means that parts of the minipacket which are not checked by the CRC, the start and end bytes, are essentially unchecked as far as the user is concerned. The monitor is able to detect certain corruptions of the start byte but the end byte containing the responses is totally unchecked. The low level protocol must be aware of this and be designed appropriately.

### 5.5.1 Maintenance

The maintenance mechanism detects bit corruptions and line breaks near where they occur and forwards a maintenance minipacket reporting the fault to an error logger. Maintenance minipackets are identified by a bit in the start byte. They are launched with the monitor passed bit set and are therefore deleted at the monitor when they pass it a second time. They contain the address of the station which sent them and the end byte contains information on the type of fault they are reporting. Except for maintenance minipackets sent from the monitor, each maintenance minipacket circulates round the ring at least once and possibly twice. Thus an error logger, which is just a station configured to receive maintenance minipackets, can be placed anywhere on the ring and receive each maintenance minipacket at least once. If the error logger is immediately after the monitor, each maintenance minipacket will only be received once.

There are three kinds of faults reported by stations in maintenance minipackets. The first is when a modulation error is detected by the NCC. This is only operative when the on-chip demodulator is being used and is disabled for the other transmission modes. The second type of fault is when an error is detected in the parity bit of a slot. The parity is inserted correctly when a slot leaves a node, but if the incoming parity was incorrect a maintenance minipacket is sent. A **gap too long** fault occurs when a

node detects that the gap has exceeded its maximum length. This may occur when the ring breaks and the node then forces a short normal minipacket slot onto the line which is used as a maintenance minipacket.

The implementation of a station to receive maintenance minipackets can be very simple because no receive buffer is required. The type of maintenance minipacket is given in RSR1 and the source address can be clocked into a 16 bit register.

### 5.5.2 Error Checking

Error checking in CFR/1 was not intended to be a substitute for checking at higher levels, but rather to reduce the effects of bit corruptions as far as the higher level protocol was concerned. The two main effects of the CRC are that it brings a level of confidence that the addresses and data of the minipacket are correct and that it enables low level retransmission of minipackets which are found to have CRC errors at the destination. This is made possible by having a **CRC error** response which is seen by the source.

## 5.6 BRIDGE DESIGN

When a CFR/1 node is working as a bridge, minipackets can be routed between rings according to the destination address. Thus, when a source sends a minipacket it uses a 16 bit global address. This address will be recognised by a bridge node which will receive the minipacket. The responses passed to the source will indicate how the minipacket was handled at the bridge. If the minipacket is received it is passed from the bridge node to another bridge node on an adjacent ring. This bridge node then transmits the minipacket on its ring. This may be the ring containing the destination station or the minipacket may have to pass through further bridges.

The responses received by a bridge node will therefore be local to the ring on which the minipacket is being transmitted. The minipacket may not be accepted and the bridge node may try to send it again a number of times before giving up and discarding it. This may pose a problem because the original source can not know from the local responses whether a minipacket has been accepted at its final destination. This is the normal problem with store and forward networks and one way of solving it is to adopt some end-to-end flow control procedure which notifies the sending station about the progress of the transmission. Alternatively we could arrange that a bridge station

does not accept any more minipackets for the destination and thus a form of back pressure flow control would be obtained. This would prevent minipackets getting out of order but would still require some timeout at the source to detect the failure of a transmission.

A bridge can also be designed to cope with channel mode minipackets. These have to be handled with care because delay fluctuations and interactions on the two rings may cause minipackets to be lost. When the first such minipacket is received at a bridge, a channel slot has to be found on the next ring and the minipacket transmitted in it. Provided that this can be done before the next minipacket arrives from the source then the transmission will continue successfully. Any other traffic trying to use the bridge is likely to upset the channel mode traffic and such operations are only likely to succeed with co-operation from all stations on the network.

## 5.7 MONITOR STATION DESIGN

The network controller chip can be configured to perform monitor functions. The monitor has address zero and can be used to receive maintenance minipackets. The functions performed by the monitor are to set up and maintain the required slot structure, monitor the ring for errors and generate error minipackets accordingly.

Two further useful features provided by the monitor are the generation of random numbers in empty slots and the extension of the ring length by buffering. The former is useful for maintenance of the ring. It provides constantly changing data in empty minipackets and thus keeps the maintenance circuitry exercised even when the ring is lightly loaded. The latter is useful to extend the ring when very short physical cable lengths are being used. One convenient way of implementing this extra delay is by using a FIFO, such as is used for transmit and receive buffers at stations. The FIFO is inserted in series with the ring at the monitor.

The slot structure required on the ring is specified to the monitor using the first 16 bits of data read from the configuring ROM. Up to 16 of each of the four types of slot can be specified in this way. The frame structure always begins with at least one short normal slot and is followed by as many long normal, short channel and long channel slots as are specified. In a simple implementation of a monitor the length of the gap is not checked. The monitor puts out the correct frame structure after which it waits for the incoming gap to end before putting out the frame structure again. Providing

there are no errors this continues indefinitely. If an error occurs the monitor waits for one ring revolution, then marks all slots full (this is known as start mode) before releasing them again. If the fault persists the monitor remains in start mode until at least two revolutions without error occur.

The monitor has a number of control inputs and outputs which can be used to monitor the state of the ring. The outputs indicate when the monitor is in start mode and when an error occurs. This gives an indication of the performance of the ring and a counter can be used to count errors. The input pins can be used to force the monitor into start mode, enable the generation of random numbers and to extend the ring by including the external FIFO in series.

As well as detecting the three normal types of faults (modulation, parity and gap too long), the monitor can detect a number of other faults. These include framing faults such as the corruption of a start or gap bit and also certain corruptions of bits in the start byte. Such faults cause the monitor to send a maintenance minipacket marked to indicate a **monitor error**.

## 5.8 PERFORMANCE

Because CFR/1 can be configured in many ways, a general performance analysis is not attempted here and instead two configurations will be examined which should give an idea of the possible performance of systems based on CFR/1. The parameters which will influence performance the most are the clocking rate and the slot structure of the rings.

The first system to be considered is a low speed system clocked at 20 MHz and having two slots on the ring, one containing 8 bytes of data and the other 16 bytes.

Assuming a gap of 4 bytes the system bandwidth of this configuration is:-

$$sysBW = \frac{(8 + 16) * 8}{((7 + 8) + (7 + 16) + 4) * 8} * 20$$

$$= 11.4\ MBit/S$$

Assuming that the short slot may only be used in normal mode and that the long may be used in either mode, we find that the point-to-point bandwidth from the long slot in channel mode is 4.3MBit/S. Since there is just this one channel slot then when it is being used in this way no other stations can use the long slot. The bandwidth

available from the short slot is dependent on the network load but is a maximum of 1.1MBit/S on an otherwise unloaded ring. Use of the long slot will not degrade this figure however, since users of the two different slot sizes are not competing with each other. The long slot may be used in normal mode in which case a maximum bandwidth of 2.2MBit/S will be obtained.

The second configuration is a ring clocked at 50MBit/S with a single short slot and two long slots each with 56 data bytes. Such a system might be useful in an environment where many fast bulk transfers must be performed, the long slots being used to carry the data while the short slot carries protocol information to administer the transfers. The system bandwidth of such a configuration is 40MBit/S, assuming a gap of 9 bytes. A wide range of transfer rates is possible with this system, the short slot being used only in normal mode and one or both of the long slots being available for channel use. The channel mode point-to-point bandwidth is 15MBit/S, while the maximum bandwidth from a long slot used in normal mode is 10MBit/S. From the short slot a maximum bandwidth of 1.1MBit/S will be obtained.

A wide variety of performance options are therefore possible. The ring configuration is governed to a certain extent by the length of ring and speed of operation, since these factors will determine the number of slots which can be accommodated. Short rings may be lengthened at the monitor to house more or longer slots, while long rings can be shortened by splitting and connection by bridges.

## 5.9 DISCUSSION

Possibly the most striking feature of the CFR/1 system is its generality. It implements a slotted ring architecture but beyond that as many options as possible were left open to the designer of systems based around it. The ability to configure the NCC to operate as station, monitor or bridge makes networks very cheap to implement and the variety of transmission modes and packet sizes means that many performance options are possible. The style of interface to the host is notable too. From the description of the transmission logic given earlier it can be seen that a very general interface is provided. There is very little encoding of signals to and from the NCC, this was to allow for easy control by hardware in the next level of protocol.

There are a number of points which are less than satisfactory, the error checking is rather inconsistent, failing to check those parts of the minipacket which most need protection. The need for a second (AFC) chip is unfortunate and purely due to lack of space on the NCC gate array.

Implementation of CFR/1, in particular of the NCC, failed because the manufacturer of the ULA on which the NCC logic was to be placed was unable to produce more than a few samples of the gate array and dropped all projects based upon it. A new medium on which to implement the CFR was therefore sought. Some time had passed since the decision was made to use the CFR/1 gate array and the range of options had widened somewhat. Much experience had been gained with CFR/1, particularly with logic simulation of such a large circuit, and the re-implementation became a redesign exercise also. This redesign is decribed in the next chapter.

Chapter 6

# THE CAMBRIDGE FAST RING

This chapter describes the version of the Cambridge Fast Ring which is currently being implemented in the Cambridge Computer Laboratory. This system is the result of rethinking the CFR/1 design and changes were made, either because the new implementation method made them necessary, or because they were felt to improve the system.

The major changes are a new implementation method using two chips, the use of just one slot size, changes to the error detection method, a different host interface and a change to channel mode transmissions. The most drastic change was made to the response mechanism. The new system does not allow the host to see the response which the destination marked. The response now only encodes two meanings, the first means do not retransmit the minipacket, the second means retransmit it. These changes and their implications are described in detail in this chapter.

## 6.1 HARDWARE

When it became clear that CFR/1 could not be made by the original manufacturer, the first instinct was to find an alternative manufacturer who offered a similar gate array and simply transfer the design to that. Such a product was hard to find, possibly indicating that the original manufacturer's claims for his gate array had been a little optimistic.

Simulation of CFR/1 had predicted clocking rates of around 75MHz for the original gate array and the alternative gate arrays were somewhat slower. This suggested that a different approach might be necessary to get the clocking speed up to 100MHz. It is true, to a certain extent, that as the speed of a gate array increases so its size (number of gates) decreases. Gate arrays constructed in ECL may be clocked at well over 100MHz, but are limited in size to a few hundred gates. Power consumption of such devices can be high and heat sinking may be required for large arrays. Another problem with ECL comes when trying to interface it to other logic types. The construction of ECL logic is such that the more positive power supply is conventionally

connected to earth, whereas all other logic families connect their more negative supply rail to earth. Complex level shifting circuitry is then required to interface the two types of logic. One manufacturer was found whose ECL arrays interfaced directly to CMOS and TTL logic and this directed attention to an implementation method mentioned in Chapter 4, where a pair of chips is used, one fast and one slow. Having found a candidate for the fast chip a slow companion for it was sought.

One of the less desirable features of CFR/1 was the large number of chips required to construct a node. Apart from the network controller and address filtering, chips a 64K RAM was required and a number of FIFOs and other chips. The largest suitable FIFO chips which were then available only held 64 bits and at least 4 were needed in the simplest node, over 20 in a more complex system which used long slots. Placing the FIFOs on the network chip would reduce the chip count of a node considerably and would probably reduce the overall node cost, despite making the CFR chip more expensive.

Having the FIFOs on the chip ruled out the use of a gate array as there were no arrays available then which were large enough. A semi-custom implementation was therefore investigated. The technology currently employed by the semiconductor industry in medium speed applications is CMOS. This has low power consumption and can be clocked at speeds up to 15MHz. Several manufacturers offer semi-custom CMOS design capability and given the ability of the ECL array mentioned above to interface directly to CMOS, it was decided to use this approach to implement the CFR.

### 6.1.1 ECL chip functions

In order for the system to function at 100MHz the ECL chip must divide this frequency, the line clock, by a suitable factor to provide a clock for the CMOS chip. A factor of 8 is suitable since this requires the CMOS chip to clock, within its capability, at 12.5MHz. Data must therefore be passed between the two chips in groups of 8 bits. This requires 16 pins since data must flow bidirectionally between the two chips.

The primary function of the ECL chip is to interface to the serial lines of the ring and convert the serial ring data into bytes which it then passes to the CMOS chip. The CMOS chip accepts the bytes, processes them and then passes them back to the ECL chip. The ECL chip converts them back into serial form and transmits them to the next node. The ECL chip is limited in size (300 gates) and the only other function it is able to incorporate is to detect the start of the slot train and signal its arrival to the

CMOS chip. In common with CFR/1 a variety of line driving options are provided and the ECL logic fits comfortably in a standard 40 pin dual-in-line package. The line inputs and outputs are ECL level signals which will drive twisted pair or coaxial cables directly. The signals which interface to the CMOS chip are level shifted within the ECL chip to interface directly to CMOS logic.

### 6.1.2 CMOS chip functions

The CMOS chip must interface to the ECL chip and also to a 64K RAM which forms the select map and bridge routing table. It must also interface to the host device. It must perform all the operations that the NCC and AFC chips performed in CFR/1 and also include transmit and receive buffers. Interfacing to the ECL chip and RAM requires 30 pins and to provide a host interface similar to that used in CFR/1 would involve a chip with around 90 pins. Packages with this many pins are available, but they are very expensive and so the pin count was kept down to 68 by changing the style of the host interface. The CMOS chip contains a large amount of circuitry, the equivalent of the NCC and AFC logic (around 2500 gates) uses around 35% of the chip, the remainder being occupied by the minipacket transmit and receive buffers. The ECL and CMOS chips and the way in which they are connected are shown in Figure 6.1a.

Connecting the ECL and CMOS chips together uses 18 pins on each device. If the ECL chip is not present, then these pins on the CMOS chip may be used for another purpose. The logic of the ECL chip is duplicated on the CMOS chip and the 18 pins are used to perform the function of serial interface to the ring. This results in a single chip network controller. It will only run at low speeds (around 10MHz) in this mode, but will make for a very low cost network.

Another facility is the ability to use just one ECL chip with several CMOS chips. This is useful when many network connections have to be made close together, for example within a single cabinet. There are two advantages to this. First, the number of chips is reduced and second, the delay inserted in the ring is less than if several pairs of ECL and CMOS chips were used. The delay inserted into the ring by the ECL chip is 24 bits and by each CMOS chip, 16 bits. When using several CMOS chips with one ECL chip the gap must be long enough to span all of the CMOS chips connected to a particular ECL chip. Figure 6.1b shows how the chips are connected in such configurations.

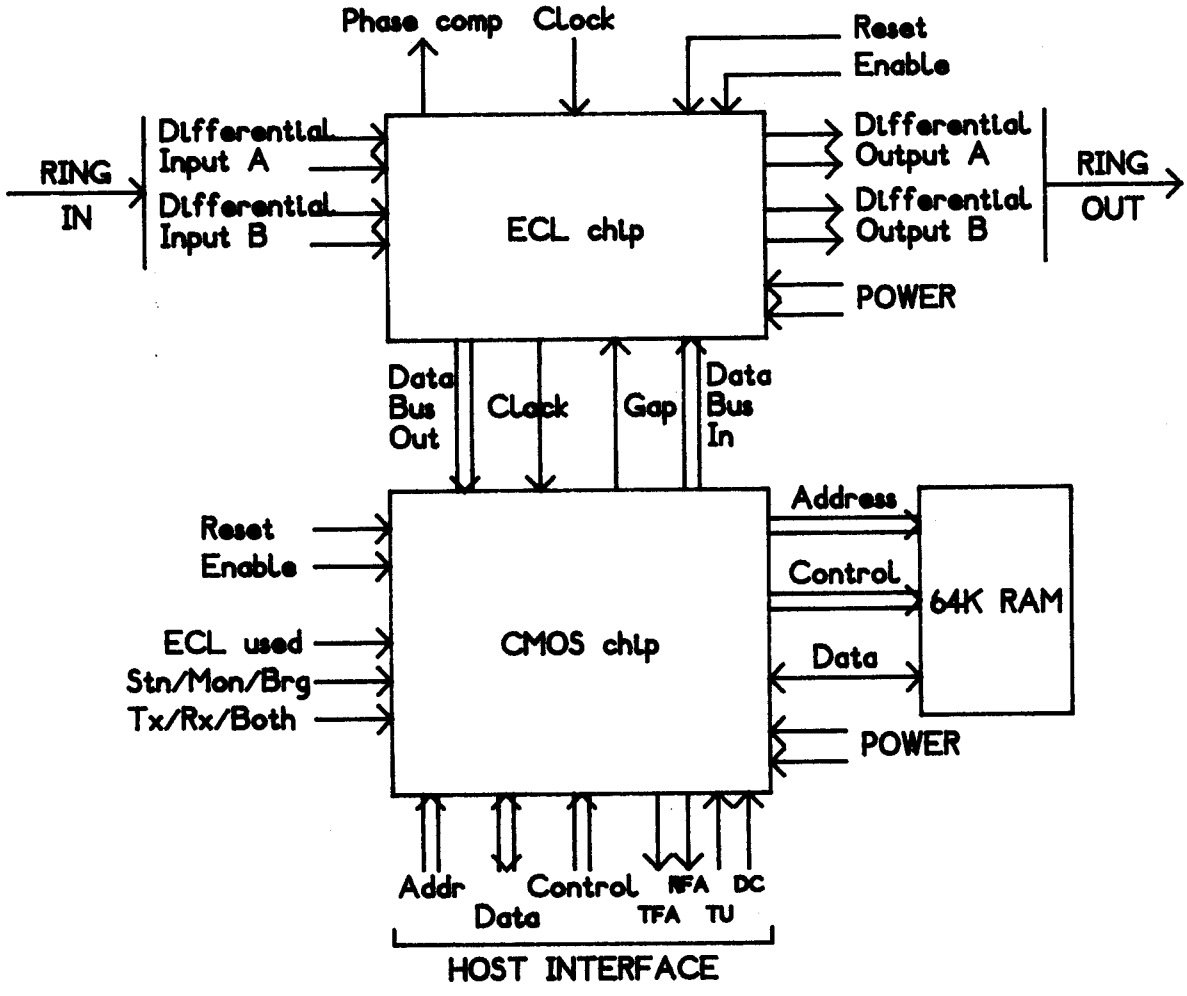



*Figure 6.1a CMOS and ECL chips of the Cambridge Fast Ring*

## 6.2 HOST INTERFACE

The host interface of CFR/1 consisted of four busses, two each for transmission and reception and 14 control lines. Simultaneous use of transmit and receive buffers was therefore possible and the whole interface was designed with high speed control by hardware in mind. When interfacing CFR/1 to a simple, slow device, such as a microprocessor, extra logic was required to interface to the microprocessor's bus. Thus, the cost of connecting simple devices was higher than it might have been. The new CFR revises this approach and makes connection to simple, bus oriented devices, such as microprocessors, easy and cheap. More complex interfaces for faster, more sophisticated hosts require extra logic and hence are rather more expensive. This is a more sensible approach, given the expected use of the single chip CFR in low cost,

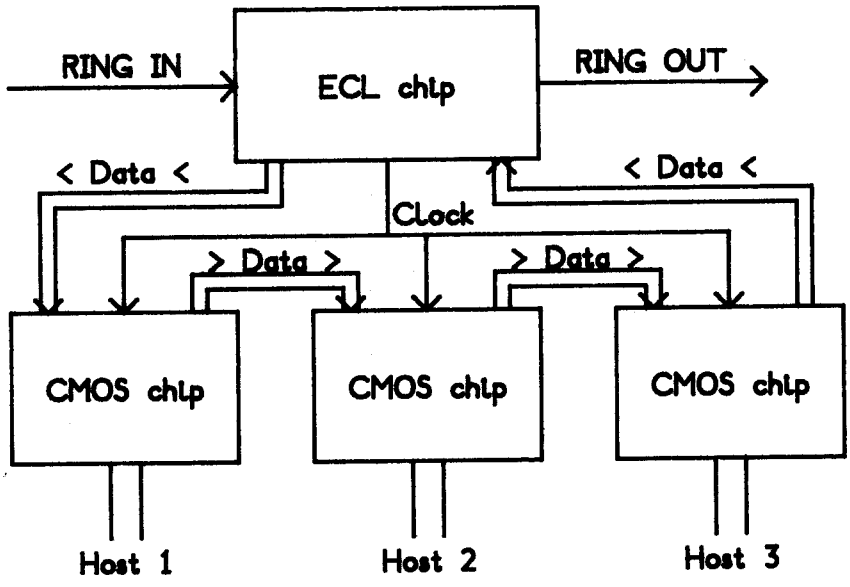



*Figure 6.1b Connection of multiple CMOS chips to one ECL chip*

slower applications. A further advantage is that the pin count is much lower, just 22 pins in the host interface compared with 40 in CFR/1.

The CMOS chip is controlled by the host writing and reading registers contained within it. The structure of the host interface is shown in Figure 6.2a. An 8 bit data bus and 5 address lines allow the host to access the registers. To allow effective control by hardware of the most common functions, transmission and reception of minipackets, four pins are provided to monitor the state of the minipacket buffers and to force transmissions and receptions. Less frequent operations must be performed by addressing data to the appropriate register. The transmit buffer appears as a single 8 bit register and successive writes to it cause data to be added to the FIFO. The receive buffer works in a similar manner.

The handshake line says when a read or write operation is complete and allows data to be moved to and from the chip at maximum speed. The chip may also be programmed to provide an interrupt to the host on a variety of events, such as the reception of a minipacket.

The reduction of the number of pins in the interface means that it is now only half duplex. CFR/1 allowed simultaneous reading and writing of the receive and transmit buffers, but this is not possible in the new design. A facility whereby this full duplex mode of operation is possible has been implemented. The CMOS chip may be

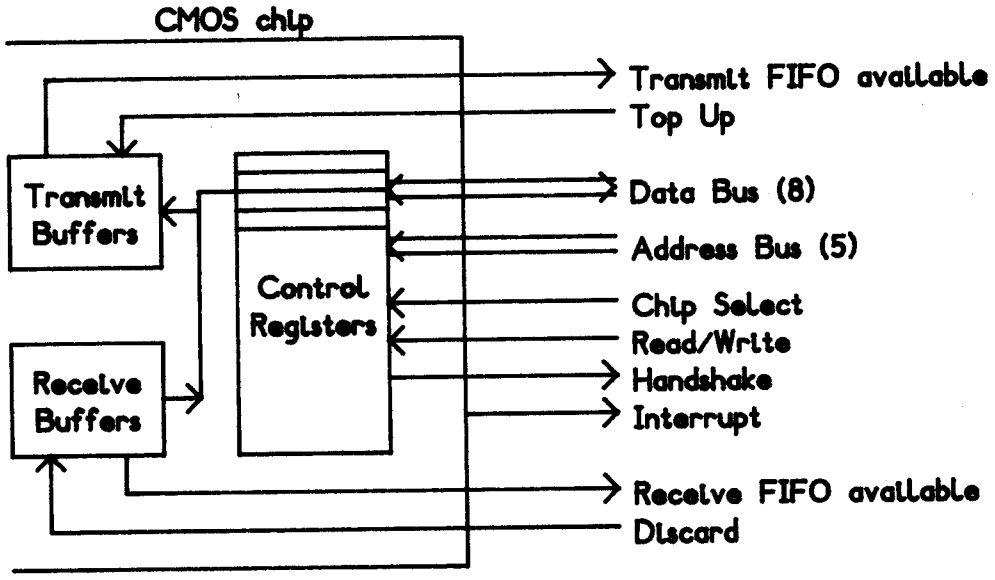



*Figure 6.2a Host interface of the CFR*

configured into one of three modes. The first is the normal, half duplex mode while the second and third are modes in which the chip is just a minipacket receiver or a minipacket transmitter. A full duplex station can then be built using two CMOS chips, one as receiver and the other as transmitter. Inside the chips the transmit and receive logics are quite distinct and so no communication is necessary between the two. The only constraint is that the receiver must be before the transmitter on the ring. A single ECL chip can connect the two CMOS chips to the ring as described in the previous section.

## 6.3 MINIPACKET STRUCTURE

The new minipacket structure is shown in Figure 6.3a and is rather different to that of CFR/1. The control bits at the front of the minipacket now occupy only four bits and comprise a start bit, the full/empty bit, the monitor passed bit and a channel slot bit. Three bits at the start of the packet in CFR/1 are not present in the new CFR. All minipackets are now of the same length and so a length bit is unnecessary. Maintenance minipackets are identified by having a destination address of zero and so a bit to identify them is no longer needed. The way in which channel mode operates has changed and so the "channel full" bit is also unnecessary.

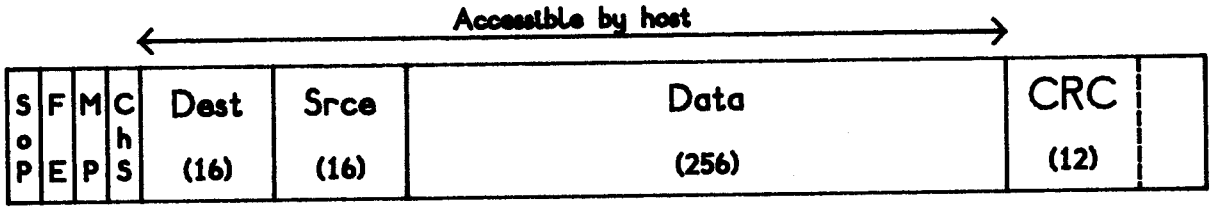



*Figure 6.3a Minipacket format of the CFR*

Source and destination addresses are still 16 bits long and they are followed by 256 bits (32 bytes) of data. At the end of the minipacket is a 12 bit CRC of all the bits in the minipacket. The end byte of CFR/1, containing responses and the parity bit, has been dispensed with, but a single bit of response is encoded in the CRC.

The fixing of the minipacket size was due to constraints imposed by implementing the transmit and receive buffers on the CMOS chip. In order to provide the facilities that a complex CFR/1 node could provide, two 60 byte transmission buffers and a 60 byte reception buffer would be needed on the chip. There was some doubt as to whether placing this much buffer storage on the CMOS chip would be possible. There was also a feeling that the complexity introduced by having two slot sizes was too great and that bandwidth would be abundant anyway. The reason for having small slots was to allow low bandwidth transfers. If there is just one slot size and bandwidth is plentiful, then using just part of a slot, or using a slot less frequently will be acceptable.

The slot size was fixed at a compromise size, large enough to allow high transfer rates, yet small enough to allow the necessary buffers to be placed on the chip. 32 bytes of data was chosen and this allowed a second reception buffer to be placed on the chip. The CFR chip therefore has two reception buffers and two transmission buffers.

## 6.4 RESPONSES

Possibly the most radical change from CFR/1 took place at the rear of the minipacket. There is now just one response bit, encoded in the last four bits of the CRC. The response must encode two states and this is achieved by inverting the last four bits of the CRC. There are thus two "correct" CRCs for a given minipacket, each

representing one state of the response. This response is never seen by the host and is simply an indication to the transmitting station that a retransmission of the minipacket may be worth attempting. The response encodes two meanings **try again** (retransmit) or **don't try again**. When a minipacket returns with the response "try again" the station automatically retransmits the minipacket up to a fixed number of times. The "don't try again" response does not imply that the minipacket was accepted, only that it is not worth retransmitting.

The reason for this change is that when one station transmits to another, it may well not know the location of the destination. This was also the case in CFR/1 and the change reflects the feeling that bridges will be an integral part of CFR networks. If the destination must be reached via one or more bridges, then a different protocol to one which assumed both stations were on the same ring would have to be used. The "same ring" protocol would use the responses to determine whether minipackets were accepted and this protocol would be useless if the destination was "off ring".

Protocols for transmission through a bridge will have to use explicit acknowledgement minipackets and such protocols will also function if the destination is on the same ring as the source. If a uniform protocol is to be used then it should be of this type and there is then no direct need for the response bits of CFR/1. Certain reasons for the destination being unable to accept minipackets would possibly be cleared by retransmitting the minipacket, a CRC error or a full receive buffer might not occur again if the minipacket was retransmitted. Thus a single bit response to indicate these events is useful to improve the efficiency of transmission on a single ring.

The automatic retransmission is likely to be of use in two ways. The first is the case of a fast transmitter sending to a somewhat slower receiver on the same ring. The receiver will mark "try again" those minipackets which arrive while the host is still busy emptying the receive buffer. The source will automatically retransmit the marked minipackets and the whole process will occur efficiently without intervention from either host.

The second use of this mechanism is at bridges. It is possible that there will be an appreciable number of minipackets passing through bridges. This means that there is a good chance that a bridge will be busy for an appreciable proportion of the time and that automatic retransmission to a bridge would be a very good thing.

The number of retransmissions that a station makes before giving up may be altered by the host. The number may be set to 4 or 16 and the interval between attempts at retransmission can be 0 or 4 ring revolutions.

## 6.5 CHANNEL MODE

The changes made to the responses were done with the aim of simplifying the protocol used on the CFR and a change was made to channel mode for the same reason. It is not now possible to explicitly request that a transmission take place in a normal slot or a channel slot, though the two types of slots are still differentiated on the ring. All transmissions are made in exactly the same way by the host and only if it supplies data fast enough will channel mode be entered. The host is unaware that two transmission modes are being employed.

Channel mode works as follows. Since the station has two transmit buffers, the host may be filling a second one of these while the contents of the first filled buffer are in transit around the ring. If, by chance, the slot originally used was marked for channel use, then the slot will be replenished from the second buffer and channel mode will be entered. If the slot was not a channel slot then it will be passed on empty and a normal transmission started from the second buffer. If the host continues transmitting and there is a free channel slot on the ring, then it will soon be used by the host's station and channel mode transmission will commence.

The host cannot tell which mode is being used and will generally not need to. For channel mode to have the same overall effect as normal mode, transmission in channel mode must stop if the destination sends back the "try again" response. There is a problem with this because the slot will already have been replenished and be on its way around the ring before the source gets to see the response from the previous minipacket. The problem is solved by sending the response out to the destination the opposite way round to normal. The destination takes this to mean "disregard this minipacket" and then all is well. The response is thus used as a directive on the forward path.

## 6.6 OTHER CHANGES

### 6.6.1 Error Checking and Maintenance

The maintenance mechanism now uses the same error check field as is used for data level checking. This is the 12 bit CRC at the rear of the minipacket and has two correct values which encode the response as described earlier. The CRC is checked at each node, but only empty minipackets have the CRC corrected if it is found to be wrong. This is because in a full minipacket the node cannot know which of the two correct CRCs to generate. However, the CRC of a full minipacket is checked and corrected at its source and destination. The only failing of this scheme is that full minipackets do not take part in the maintenance function. A maintenance minipacket is sent by any node which corrects the CRC of an **empty** minipacket.

Maintenance minipackets are sent to destination address zero and it is this which distinguishes them from normal minipackets. The type of error they are reporting is encoded in the data field. Any station with address zero will receive maintenance minipackets. By default, the monitor has address zero and so will receive maintenance minipackets in the absence of an error logging station. The monitor removes maintenance minipackets using the mechanism which detects full minipackets circulating around the ring twice. On finding such minipackets the monitor marks them empty and thus a maintenance minipacket may pass an error logging station twice.

### 6.6.2 Node Configuration.

The CFR/1 node was configured by reading data from a ROM. The data indicated the node type and various other parameters such as the line driving mode and the length of long slots. The new CFR uses a slightly different approach. Those options which the host should not be allowed to tamper with are provided on pins of the CMOS chip. Those options which the host might legitimately be allowed to change are set by writing to a register within the chip. The hard wired options say what type of node the chip is and whether or not an ECL chip is attached. The "soft" options control such things as the number of retransmissions that are made after a "try again" response is received and the number of slots of each type that a monitor places on the ring when starting up.

### 6.6.3 Transmission and Reception

Two transmission buffers are used to make good use of the available bandwidth. The automatic retransmission of minipackets relies on the original contents of the minipacket being retained in the transmitter and it is convenient to be filling a second buffer while the first is being transmitted. The reception of minipackets is also made more efficient by having two buffers.

In order to transmit a minipacket a host simply has to write 32 bytes of data into a transmit buffer, having previously written a destination address into a separate buffer. Transmission takes place as soon as the $32^{nd}$ byte is placed in the buffer. An output pin on the chip called **Transmit FIFO Available** (TFA) indicates when a transmit buffer is free. The host uses the TFA signal as a prompt to place more data in the buffer, it is unaware of the fate of previously transmitted minipackets. If the buffer is not to be filled with the entire 32 bytes before being transmitted, the host can use the **Top Up** (TU) input to force the transmission of whatever is in the buffer at the time.

Receive buffers are managed on a similar basis. A signal to the host called **Receive FIFO Available** (RFA) indicates that there is a full buffer waiting to be read. Reading all 32 bytes initiates the reception of another minipacket. There is a **Discard** (DC) input with which the host may force the emptying of a receive buffer. There are also receive address buffers into which the source addresses of received minipackets are placed for reading by the host.

### 6.6.4 Source Selection

The same basic mechanism as is used in CFR/1 is used again, but with a few alterations to make selection operations slightly easier. The 16 bit select register and 64K bit map are still used, but rather than using specific values in the register to disable reception or to receive from anybody, a separate register is used to control these functions. A bit in this register also enables the reception of broadcast minipackets.

Because the receive buffers are now on chip, it is possible to coordinate selection operations with them. When the station is set to disable all receptions, the buffers are emptied. When the select register is set to receive from a given source, the receive buffers are emptied unless their contents happen to be from that source.

## 6.7 PERFORMANCE

Because the minipacket size is constant there is a narrower range of performance options possible than with CFR/1. Hopefully, clocking at 100MHz will be possible and higher rates may be attained if faster versions of the CMOS chip become available, as is predicted.

The performance seen by a host is less predictable than it was in CFR/1. All transmissions start off in normal mode and if there is a free channel slot on the ring it will be used after a small number of transmissions have been made in normal mode. If the host can transmit fast enough and the destination host receive fast enough, then channel mode will be used and there will be a sharp increase in the transmission rate. If the sender now slows down channel mode will cease and normal mode will be re-entered.

This approach has both advantages and disadvantages. A uniform transmission policy may be employed which provides for both very slow transmitters and very fast ones. This is a good thing in most cases. There may be cases where a transmitter might be inconvenienced by having to send the first few minipackets in normal mode before getting into channel mode. Certain real-time applications may demand that if transmission is to take place at all it should do so at the same rate throughout.

With a small gap the system bandwidth of the CFR is about 80% of the clocking speed. If clocked at 100MHz the system bandwidth will be 80MBit/S. The maximum point-to-point bandwidth will be obtained by transmitting in channel mode slots. If there are no channel slots free then normal mode transmissions will be used with a lower bandwidth, which is also dependent on other traffic on the ring. A graph showing the bandwidth that can be achieved for each of the modes is shown in Figure 6.7a. This assumes a clocking rate of 100MHz. The upper set of points are the channel mode figures. The normal mode bandwidth will degrade as the number of transmitters increases. The channel mode bandwidth will be constant but subject to the constraint that only as many channel mode transmitters as there are channel slots will be allowed.

As an example, a ring with 5 slots, clocked at 100MHz, can accommodate 4 channel slots each giving a constant bandwidth of 16MBit/S and one normal slot with 16MBit/S capacity to be shared between the remaining transmitters. If a channel slot is not

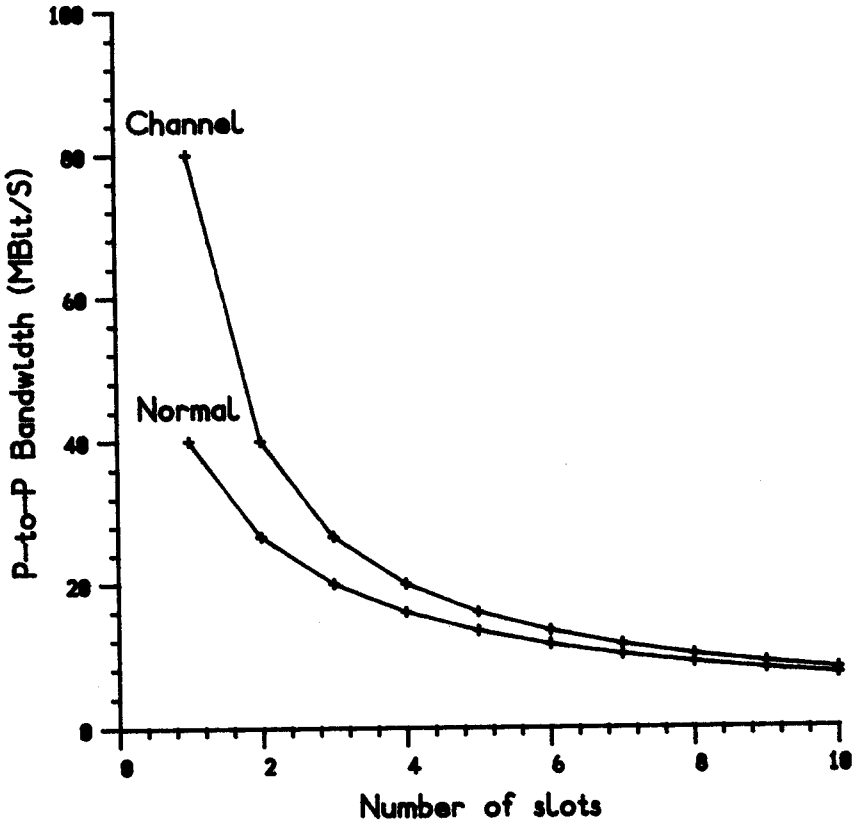



*Figure 6.7a* ***Point to point bandwidth of the CFR***

being used for channel transmissions then it is available for use by anyone in normal mode.

## 6.8 DISCUSSION

The CFR described in this chapter is significantly different from CFR/1, its predecessor. Much of the generality of CFR/1 has disappeared and it is much easier to construct simple network nodes, since the number of parts required has fallen dramatically. A slow, simple node may be constructed from the CMOS chip, a 64K RAM and a line driving chip. This will give a performance comparable to the Cambridge Ring but at a fraction of the cost.

The error detection and maintenance mechanism is improved over CFR/1, all bits of the minipacket are now protected and all detected errors are acted upon. The source selection mechanism is rather better too.

Providing an option whereby channel mode could be requested explicitly might have been useful. This might also be arranged so that a receiver missing the occasional minipacket would not cause the channel to be lost. This would be of value when transmitting video and similar types of data where the loss of a small amount does not matter but the loss of the transmission channel would be serious.

Chapter 7

# BRIDGES

This chapter discusses the bridge facility which is provided in CFR networks. The ability to send minipackets to destinations on other rings is implicit in the design of the CFR. A station can be unaware of the location of the destination of the minipackets it is sending and minipackets will cross bridges automatically to reach their destination.

This has implications for the protocols in use on the network since the time taken for a minipacket to reach its destination will vary widely. There is also a finite chance of a minipacket being thrown away (lost) at bridges on its path. The maps which control the routing of minipackets through bridges must be set up and maintained in a consistent state.

## 7.1 MODES OF USE

There are a number of ways in which the facilities of a bridge may be exploited. It may be used to connect two otherwise autonomous rings for convenience reasons only, or it may be used to allow a large slow ring to be split up into smaller and hence faster rings. A further reason is to allow the division of a network into small units for geographical or administrative reasons. Finally, the splitting of a large ring into small ones will improve the reliability of the network.

From these classifications we can predict two extremes of bridge use. One will be systems where the majority of data flow occurs on rings and very little passes through bridges. The other has much traffic passing through bridges, there being no distinction made between on-ring and off-ring minipackets. This suggests that a wide range of traffic handling capability may be required from bridges and that a number of different designs may be needed to cope with the various requirements. If the very fastest bridge we can build is not adequate then it is possible to place two or more of them between two rings. Care must be taken in this case to ensure that their maps are configured so that minipackets for a given destination will pass through only one bridge. If this is not the case then minipackets may arrive at their destination in a

different order to that in which they were transmitted. This would make the reassembly of messages from minipackets more difficult than it would be if minipackets arrived in order.

## 7.2 RING BRIDGE DESIGN

A bridge node is a CFR node which receives minipackets on the basis of their destination address being in a 64K bit map. When a minipacket passes a bridge node its destination address is used to address the map and the addressed bit says whether or not the destination can be reached through the bridge. If it can and the bridge node has a reception buffer free then the minipacket will be received. Not only is the data field of the minipacket buffered, but also the source and destination addresses. All of these items are placed in a 36 byte reception buffer. The bridge node can also transmit these 36 byte structures. Writing 36 bytes to a bridge node's transmit buffer will cause them to be interpreted as addresses and data and to be transmitted. Two bridge nodes are required to construct a **bridge** which connects two rings and transfers minipackets between them.

### 7.2.1 Simple Bridge

The simplest bridge which will transfer minipackets bidirectionally consists of two bridge nodes connected back to back. Some arbitration circuitry is required to ensure that only one transfer is attempted at once on the data bus which connects the two nodes. In addition, a means of initialising and updating the bridge maps is required. A single chip microprocessor will be adequate for this purpose (the **bridge processor**). It must be attached to one or both rings to enable it to communicate with whatever service provides the necessary information to keep the maps in order. Since bridges cannot transmit and receive in their own right the bridge processor must have a station on one of the rings to allow it to communicate. A more symmetrical arrangement is to have a station on each ring and this may be useful when traffic cannot flow through the bridge. This situation may arise during network initialisation or while updating the maps. An example of such a configuration is shown in Figure 7.2a. This design takes advantage of the ability to use more than one CMOS chip with a single ECL chip.

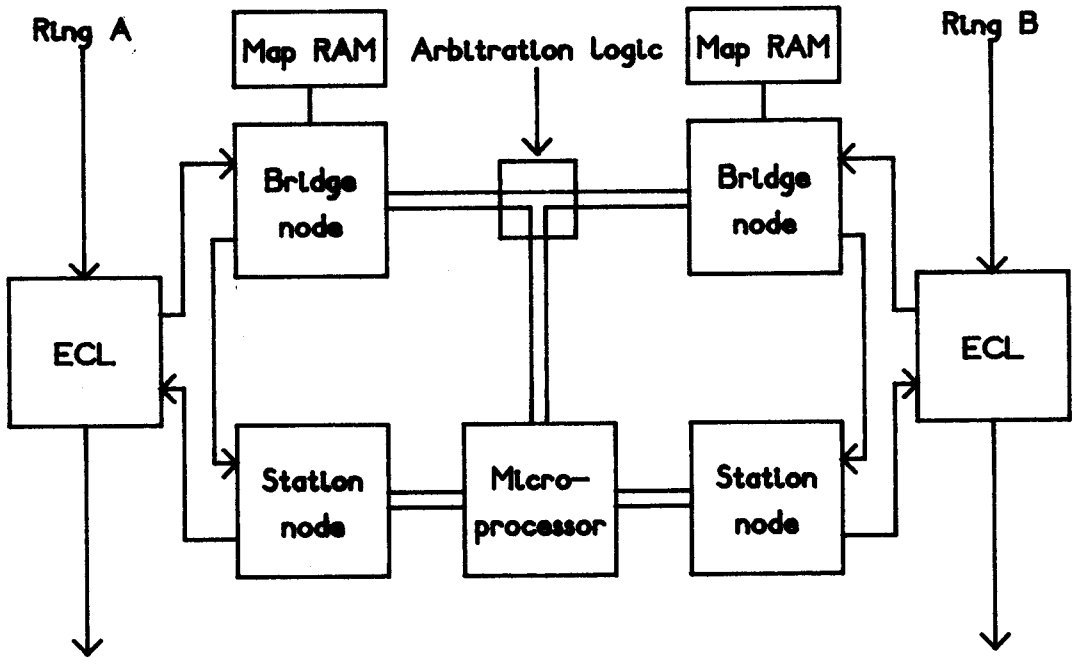



*Figure 7.2a Configuration of a simple bridge*

### 7.2.2 Full Duplex Bridge

Because the CMOS chip can be configured to act as a transmitter only or a receiver only, pairs of chips can be used to construct a fully duplex bridge. The performance of such a bridge under heavy load will be higher than the previous design since minipackets may cross the bridge in both directions at the same time.

Similar considerations to those of the simple bridge apply to the updating of the bridge maps. In this case the map RAMs will be attached to the receiver chips and no RAMs are needed for the transmitting nodes. The structure of the duplex bridge is shown in Figure 7.2b.

It has been assumed so far that bridges will be bidirectional. A unidirectional bridge is also easily constructed. It will require a bridge node configured as a receiver on one ring and a transmitter node on the other ring. A station node and microprocessor are associated with the receive side to maintain the map RAM. This configuration could prove to be a useful building block, since with minor changes it could behave as the simple bridge described earlier. Using two of these blocks would allow the construction of the full duplex bridge, while on its own it would function as a unidirectional bridge.

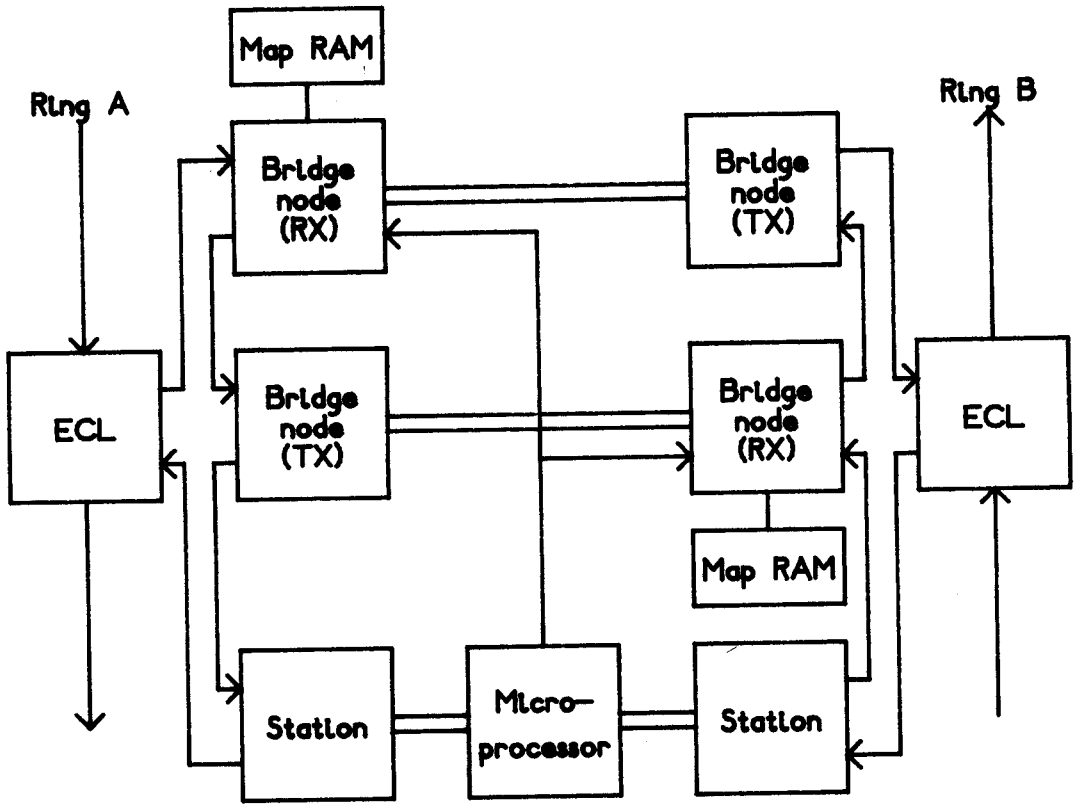



*Figure 7.2b* ***Configuration of a duplex bridge***

Unidirectional bridges may be of use in relieving traffic congestion problems where the flow of minipackets is predominantly in one direction. A ring with a continuous large influx of minipackets might have one unidirectional bridge taking minipackets off the ring and two or three bringing minipackets on to it.

## 7.3 BRIDGE MAPS

There are two fundamental operations which must be performed on the maps at bridges. The first is initialising the map when the bridge becomes active and the second is updating the map as a result of changes to the network structure. The changes which are relevant are the addition and removal of stations and the movement of addresses from ring to ring. It should be borne in mind that the bridge works only in terms of physical addresses and these are generally closely bound to a particular station. Frequent moves of addresses are unlikely under normal circumstances, although it is possible to conceive of situations where movement could occur quite often. For example, in systems which use redundancy of hardware to improve reliability, the failure of a node or ring might cause alternative hardware to be used.

The replacement nodes might well duplicate the addresses of the failed or isolated nodes. This would necessitate changes to the bridge maps if the replacements were on different rings to the original stations.

In general use, the contents of bridge maps are likely to be fairly static, with the time between updates being measured in days rather than seconds. In this case updating might be best performed by a complete reinitialisation. If updating is necessary on a more frequent basis then a more subtle approach can be used. Co-operation between bridge processors will be required to keep the maps in a consistent state. Inconsistency could lead to minipackets circulating indefinitely if there were a loop in the network. Algorithms for updating the maps are well known, having been devised many years ago for use with store and forward networks.

The initialisation of the maps will be done from some record of the system configuration kept on stable storage such as a disc. The following method uses broadcast minipackets to help with the initialisation process. On powering on, each bridge processor sets the maps so that no minipackets can pass through the bridge. This means that broadcast minipackets will be constrained to the ring on which they are transmitted. The bridge processor now sends a broadcast minipacket requesting the address of the machine which will supply map information, the **map server**. It repeats these requests at intervals until it receives a reply. The map server will receive these requests from bridge processors on its own ring and will respond by sending them its own address. The bridge processors can now send a request to the map server asking for their maps. The map server gives them the appropriate maps and these processors now reply to requests for the map server's address coming from bridge processors on adjacent rings. These bridge processors can now request their map from the map server via the recently set up bridges. This process continues until all bridge processors have obtained their maps.

A recent development in memories is the Electrically Alterable Read Only Memory or EAROM. This is a read/write memory which retains its contents when power is removed. Writing is somewhat slow, but reading takes place at conventional speeds. Using this memory for bridge maps would remove the need for initialisation every time a bridge was turned off and on again, though clearly it would need setting up at some stage.

### 7.3.1 Low Cost Bridges

If the configuration of a CFR network involving bridges is completely static, then the maps may be made from a non-volatile memory such as a PROM. This will remove the need for processors at bridges and make them very simple and cheap to make.

The flat address space of the CFR may be interpreted as the user wishes. It is also possible to treat it in various ways at bridges. One possibility, which leads to very cheap bridges, is to split the address into an 8 bit ring number and an 8 bit station number. Bridges then operate in the following manner. If the destination ring number of a minipacket is different to the number of the ring on which the bridge is situated, the minipacket is received. If the ring numbers are the same, then the bridge ignores the minipacket. Each bridge only needs to know the ring numbers of the two rings to which it is attached for this scheme to work. There is a problem with ring numbers of non-existent rings. If there is a loop through bridges in the network and a minipacket is sent to a non-existent ring then it will circulate forever. To prevent this the topology must be restricted to exclude such loops.

## 7.4 FACTORS INFLUENCING BRIDGE PERFORMANCE

Whilst high speed traffic is expected through bridges, it seems unlikely that channel mode will be effective through general purpose bridges. This is because any minipacket arriving at a bridge and having to be retransmitted on the other side will hold up the channel packets and cause normal mode to be resumed. This may be satisfactory in some cases, but if a guaranteed channel transmission is required across two rings a dedicated bridge will be needed. The bridge map must be set to pass minipackets only for the channel destination. A duplex bridge would allow bidirectional channel mode conversations, between the two stations assuming that they were duplex too.

The main objective of a bridge is to transfer minipackets from a variety of sources to a variety of destinations. The bridge is equipped with two transmit buffers and two receive buffers in each direction. The speed at which data may be passed from the reception buffers on one ring to the transmit buffers on the other will influence the delay experienced at a bridge.

Having received a minipacket a bridge node must transfer 36 bytes of data and addresses to the adjacent bridge node. Preliminary figures for the speed of the FIFOs on the CMOS chip suggests that the buffers can be read and written, via the external interface, at the rate of 150nS per byte. The whole 36 bytes will therefore cross in 5.4uS. Assuming rings to be lightly loaded, the average time before transmission can take place will be half a slot time, or just over 1.5uS on a 100MHz ring. The minimum delay from the reception of a minipacket at a bridge to its retransmission on the next will therefore be around 7uS. Minipackets sent across bridges will need explicit acknowledgement. The delay through bridges is sufficiently large to make acknowledging each minipacket very wasteful of bandwidth and, if possible, minipackets passing through bridges should be acknowledged in groups.

When the rings to which a bridge is attached are heavily loaded, the bridge may have to wait some time before transmitting and a backlog of minipackets may develop. Four minipackets can be buffered in a bridge in each direction, two in receive buffers and two in transmit buffers. When these buffers are full the bridge will start rejecting incoming minipackets, marking their response "try again". Judging from the traffic patterns seen in the study of Chapter 3, there will be bursts of activity during which bridge congestion is most likely to occur and longer spells of low activity. If the bursts can be readily handled by the bridge then there are no problems, if not, then additional buffering at the bridge may enable the burst to be endured.

A further consideration, under conditions of heavy load, is the way in which retransmissions are made. The issue here is what to do when a bridge is transmitting to a destination which does not receive the minipacket. If the response says "don't try again", then the bridge stops transmitting and throws the minipacket away. In the case of the response saying "try again", the transmitter will attempt a number of retransmissions. This will hold up the passage of minipackets through the bridge far more seriously than heavy load on the destination ring. This means that the retransmission algorithm must compromise between not occupying bridges too long and being sufficiently prolonged to allow slow receivers to receive effectively. When rings of widely differing speeds are interconnected by bridges the problem becomes even more acute. The CFR includes options to allow the retransmission algorithm to be altered. The number of retransmissions and the interval between them may be chosen from a limited range.

## 7.5 DISCUSSION

Bridges bring with them a number of useful facilities and a number of potential problems. Partitioning a large network into many rings is possible and brings higher reliability and higher data rates than if the network was made with a single ring. The problems arise when bridges carry a lot of traffic. Their limited buffering capability means that it is likely that minipackets will get thrown away because they cannot enter a bridge. This problem will be particularly apparent when bridges connect rings of different speeds. Because the retransmission rate is related to the ring's clocking speed, minipackets will be retransmitted much more slowly on a slow ring than on a fast one. At times of heavy load many minipackets sent to the bridge from the fast ring will be thrown away because the bridge spends a long time performing retransmissions on the slow ring. A larger buffer at bridges may help to reduce this problem if the average level of bridge traffic is much smaller than the peak level.

Chapter 8

# PROTOCOLS

To use the hardware of the Cambridge Fast Ring to build a computer network requires more than just the CFR chips and wire to connect them together. To connect computers to the network some form of interface hardware is needed. Having connected a computer to the CFR, a protocol system is required to allow ordered communication with the other computers.

The CFR carries fixed size blocks of data between its stations and provides no guarantee that they will be correctly delivered. The communication software in each computer must implement some form of protocol for converting the arbitrary data structures which are to be sent around the network to and from the 32 byte blocks which the CFR deals in.

This chapter discusses protocols suitable for use on the CFR and presents a hypothetical design which might be usable for computer communication on it. In the context of the CFR, the protocols of interest are the low level ones by which data is moved from machine to machine. These protocols are likely to be unique to CFR networks since they are dependent on the network hardware. They will then be used to implement higher level protocols in which some degree of standardisation is possible.

## 8.1 PROTOCOLS AND TRAFFIC TYPES

The types of protocols used will vary with the particular application. A protocol used for conveying voice data is unlikely to be useful for file transfers. The CFR was designed with at least three different traffic types in mind. First and foremost it supports computer communication. This includes file and mail transfers, terminal traffic and all the associated control information. Second, voice traffic is expected. This requires fast access to the network and low delay, but the loss of a small number of minipackets is acceptable since the listener can compensate for small losses. The third data type is video where a constant stream of data must be transmitted at high speed. Here again the loss of a small amount of data is unlikely to be a problem.

Protocols for voice and video applications are likely to be quite simple. The error rate of the network will be low and only small numbers of minipackets will fail to reach their destination. Extensive error checking and minipacket retransmissions will be unnecessary for these applications.

The protocols for computer data will need to be more complex. A range of services may need to be provided and it must be possible to provide reliable message transport. The remainder of this discussion of protocols is concerned with computer communication protocols.

## 8.2 PROTOCOLS AND THE CFR

The particular characteristics of the CFR which are relevant to the design of protocols are its high speed, its topology and its minipacket format. The high speed means that for very fast communication the protocol must be simple to minimise the amount of processing needed to implement it. For very high speed communication some of the protocol operations may have to be performed by a dedicated processor and/or specialised hardware.

The topology of CFR networks, in particular the existence of bridges, means that the time taken to deliver a minipacket will vary greatly. It will be short when the destination is on the same ring and much larger if many bridges are traversed. In the interests of simplicity minipackets should arrive at their destination in the order in which they were transmitted. This means that if a network employs bridges there should only be one route between any two stations. There may be many possible physical routes but the bridge maps can ensure that only one of these is actually used.

The data capacity of a minipacket is 32 bytes and this means that many messages will be too large to fit into a single minipacket. Large messages will have to be split across many minipackets by the sender and reconstructed by the receiver.

The protocols must take these properties into account. The ability to construct a protocol controller in hardware is important for reasons of speed. Another consideration is that the protocol should also be easy to use on slow simple machines. It may be that certain facilities of the protocol may be relevant only in very fast applications but it should still function without these non-essential parts. An example of this is in acknowledging blocks of minipackets. A high performance protocol might use sequence numbers to check for missing minipackets and send a minipacket as soon

as a sequence error occurs. A simpler and slower protocol might expect an acknowledgement minipacket at the end of the block and use a timeout at the source to detect that the whole block did not arrive. The combination of these two methods within the same protocol would not cause any conflicts and would cater for both simple and sophisticated protocol controllers.

### 8.2.1 Basic Requirements

Many current local networks use two basic forms of protocol. The first, known as the Single Shot Protocol (SSP) on Cambridge Ring networks, consists of an exchange of data and is a datagram type protocol. A caller sends a message to some recipient who responds by returning another message. Retrying the first message in the absence of a reply is not part of the protocol. The second type of protocol, known as the Byte Stream Protocol (BSP), is a bidirectional exchange of data with the concept of a connection. This is a virtual circuit protocol and messages pass between the two participants until one of them decides to close the connection. When the connection is opened there is the concept of a calling and a called host, but thereafter the connection is symmetrical.

Protocols such as these can be implemented on the CFR and will be adequate for present requirements. What is needed to allow their implementation is some means of sending an arbitrarily long message over the network. The message must be split over many minipackets if it exceeds 32 bytes in length. We wish to know if the message has reached its destination and so an acknowledgement scheme will be required. It is assumed that the network will include bridges and that the protocols must function whether or not minipackets pass through bridges to reach their destination. Ideally, an acknowledgement will be a single minipacket and in the interests of high data rates an acknowledgement should not be sent for each minipacket of the message.

Messages can be divided into blocks each containing a number of minipackets. Each block except the last will contain the same number of minipackets and an acknowledgement minipacket is expected after each block has been sent. Thus a block will not be transmitted until the previous one has been acknowledged. The size of a block will depend upon the likelihood of all its minipackets getting to the destination. If there is a high chance of a minipacket being lost then blocks should be small to reduce the number of block retransmissions. Changing the block size to suit the prevailing conditions may be possible but will add to the protocol's complexity.

Some means of detecting the correctness of each block is required. The main concern is in detecting lost minipackets rather than corrupt ones. The CRC in each minipacket will detect most data corruptions and ensure that a minipacket is rarely duplicated. Bridges may throw minipackets away at times of heavy load and this must be detected by the protocol so that correction can occur. A simple protocol might involve the destination counting the minipackets of a block as they arrive and sending an acknowledgement when all are received. If some of the block's minipackets fail to arrive then no acknowledgement will be sent and the use of a timeout at the source will detect that this was the case. If the timeout period expires the block is retransmitted. It may be that the acknowledgement from the destination was lost and so when the block is retransmitted it must be marked as a retransmission so that the destination is sure what is happening. A more complex protocol could use an end of block acknowledgement as just described but could also use a sequence number in each minipacket to detect the loss of a minipacket more quickly. A gap in the incoming sequence numbers would indicate the loss of a minipacket and the destination could immediately send a minipacket to the source indicating the error and giving the last good sequence number it had received. The source could then retransmit from that sequence number and a much faster recovery would occur than with the simpler protocol.

The maximum length of a block in which sequence numbers are used will depend on the size of the sequence number. An N bit sequence number allows blocks of up to $2^N$ minipackets. If many minipackets can be transmitted without loss or error then very long blocks are possible and the data rate will be higher than with short blocks. This is because the number of acknowledgements will be less.

### 8.2.2 Channel Numbers

Another important problem is arranging for two simultaneous message transfers between the same pair of stations. The minipackets of each message will arrive and must be distinguished in some way to allow correct reassembly of the messages. The problem can be circumvented by insisting that only one message be allowed at once and this may be reasonable in some cases, for instance slow hosts which could not cope with more than one incoming message at a time. A full solution to the problem is to have some way of identifying to which message incoming minipackets belong. Each minipacket can carry a **channel number** which is used for this purpose. Channel numbers will be used for the duration of a message and discarded or re-used when the

message is complete.

There are a number of ways in which channel numbers may be allocated. Since they are transient and liable to be re-used they must either be very large or very carefully treated to ensure that a channel is really "dead" before its number is used again. Since they must fit into every minipacket of a message it is better to keep them small and look after them carefully.

The channel number may be chosen by the source of the message, in which case the destination must use it in conjunction with the source address to uniquely identify a message. The destination must look up both the source address and the channel number in each minipacket in order to identify the message uniquely.

The channel number may be chosen by the destination. This sounds strange at first but has a number of advantages. The drawback is that an initial exchange of minipackets is required for the source to signify that it wishes to transmit and for the destination to reply, returning the channel number it has allocated. The advantages of this scheme are numerous however.

First, the destination gets the opportunity to reject the message before it is sent. If many rejections occur this will reduce the amount of data on the network and will help to prevent bridge congestion. The rejection may be because the destination does not wish to receive from the source or because it has no spare channels to do so. Second, it allows the source and destination to specify other information, such as the block length to be used or the speed of their interfaces. Third, the destination can identify each message by the channel number alone and it can allocate as many or as few as it likes. A simple host might only be able to cope with three incoming messages and so would allocate a maximum of three channels at once.

The need to obtain a channel number from the destination is only a slight inconvenience. It does not affect transmissions which deal only in single minipackets, nor does it affect BSP type connections since they need an opening exchange of information anyway. Only operations like SSP will be affected and the advantages gained outweigh the disadvantage of the initial exchange.

## 8.3 A PROTOCOL FOR THE CFR

The protocol described here uses the methods described previously, block transmission with sequence numbers and channel numbers granted by destination, to provide a framework for building BSP and SSP type protocols. It is expected that other protocols to this **Basic Protocol** may be used on the same network and so the first data byte of each minipacket is used to indicate the protocol type. Other protocols might be a voice protocol, a video protocol or a network protocol used for initialising bridge maps as described in the previous chapter.

The Basic Protocol uses the next three bytes of each minipacket as shown in Figure 8.3a.

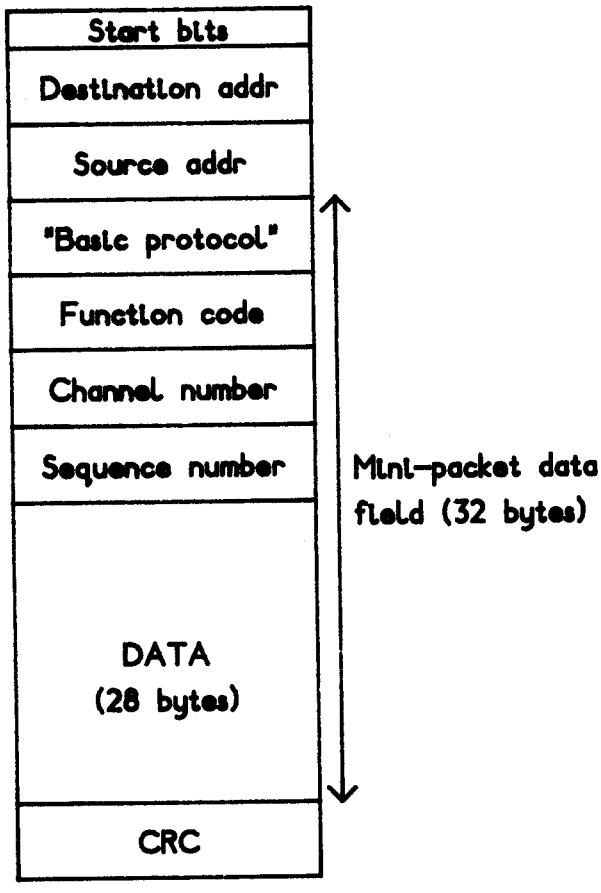



*Figure 8.3a Format of a Basic Protocol minipacket*

The first of the these bytes is a function code to describe the function of the minipacket, for instance an acknowledgement or a data minipacket. The second of the three is an 8 bit channel number and the third is an 8 bit sequence number. Up to 256 channels may be in use at a station and blocks of up to 256 minipackets may be sent

before an acknowledgement is transmitted. The Basic Protocol is based on two sub-protocols, the **Exchange protocol** and the **Transfer protocol**.

The Exchange protocol is simply an exchange of minipackets by two stations. A source sends a single minipacket and expects one in reply. A timeout at the source guards against non-delivery of either minipacket. The first minipacket is sent to channel zero at the destination which is the channel reserved for the Exchange protocol. The returning minipacket is sent to a channel specified in the first minipacket. The source may have set up a channel for the reply or it may expect it on channel zero. The Exchange protocol may be used to transmit small amounts of data or to set up channels for use by the Transfer protocol.

The Transfer protocol caters for the sending of arbitrary length messages. The messages are divided into blocks and each block is acknowledged when received by the destination. Both source and destination have a channel number allocated for the message and so minipackets may flow in either direction. There are a number of options available within the transfer protocol. These can be agreed upon when the transfer is initiated by an exchange. The default options should be such that the simplest, slowest machine on the network can use the Transfer protocol. One of these options concerns the acknowledgement scheme. A number of levels of acknowledgement are possible and the level could be agreed to suit both parties. Another option might determine whether messages may flow in both directions simultaneously, or if stations must take turns.

Messages must be delimited and the function byte is used to indicate whether a data minipacket is the first of a message, the last, neither or both. Thus messages may be of any length from one minipacket up. No interpretation is made of the data contents of minipackets in the messages, nor does the end of a message imply that a channel should be closed. The next level of protocol may be such that a channel is closed when a message on it ends but the Transfer protocol makes no assumptions regarding this.

The length of block sent between acknowledgements can be agreed in the initial exchange which sets up the channels. The simplest acknowledgement scheme involves the sender using a timeout to detect that the destination has not received the whole block and acknowledged it. If the timeout occurs then the block is retransmitted. The minipackets of the retransmitted block must be marked (in the function byte) to

show that this is a retransmission, since the destination might have actually received the block and sent an acknowledgement which failed to reach the sender. If this occurred then the destination would (wrongly) assume that the retransmitted minipackets belonged to the block following the one just received. A number of retransmission attempts may be made for each block. If they all fail, either the network may be broken or the destination has stopped listening or there may be congestion. In the latter case reducing the block size may allow communication to continue.

The acknowledgement scheme does not, so far, take advantage of the sequence numbers. These may be used in a scheme in which the destination sends a negative acknowledgement minipacket as soon as a sequence error occurs. This minipacket contains the last good sequence number that was received and the source retransmits the block from that sequence number. Recovery from a lost minipacket in a block is much faster with this scheme than the simpler one. Fewer minipackets are retransmitted and it is not necessary to wait for a timeout. As with the simpler scheme the retransmitted minipackets must be marked as such. There is also the added complication that minipackets from the first transmission of the block may still be in transit and could arrive at the destination which must therefore be prepared for them.

The allocation of channel numbers is done when a request is made for one. The protocol above the Exchange and Transfer protocols may specify when a channel is no longer required. Some means of reclaiming channels in the event of host or interface failure is also necessary. A timeout can be associated with each channel number and its period may depend on the protocol in use on the channel. A virtual circuit may have a long timeout period while a datagram protocol may have a shorter one. It is important that both sender and receiver using a particular channel are agreed as to when it no longer exists. If confusion arose, then the channel might be reallocated and minipackets could arrive on that channel from two different sources.

### 8.3.1 A Single Shot Protocol

As an example of the use of the Exchange and Transfer protocols a higher level protocol similar to the SSP used on the Cambridge Ring will be described. The SSP involves a source sending a message to a destination. When the message is received the destination sends a reply message back to the source. At this point the

transaction is complete.

The first stage in an SSP transaction using the CFR protocols is the establishing of channels for the message transfers. The source allocates a channel number for minipackets from the destination and uses the Exchange protocol to send this number to the destination. This first minipacket of the exchange requests that the destination allocate a channel for an SSP transaction and may specify the options which the source would like to use in the Exchange protocol. It might also usefully specify the amount of data that is to be sent, so that the destination could allocate a buffer for it. The minipacket is sent to channel zero at the destination. The reply minipacket from the destination will contain the channel number which the destination has allocated and a refined set of options which are acceptable to both parties. Alternatively it will contain a refusal to allocate a channel and may specify the reason for the refusal. This minipacket will be sent back to channel zero at the source. The minipackets of the exchange are shown in Figure 8.3b.

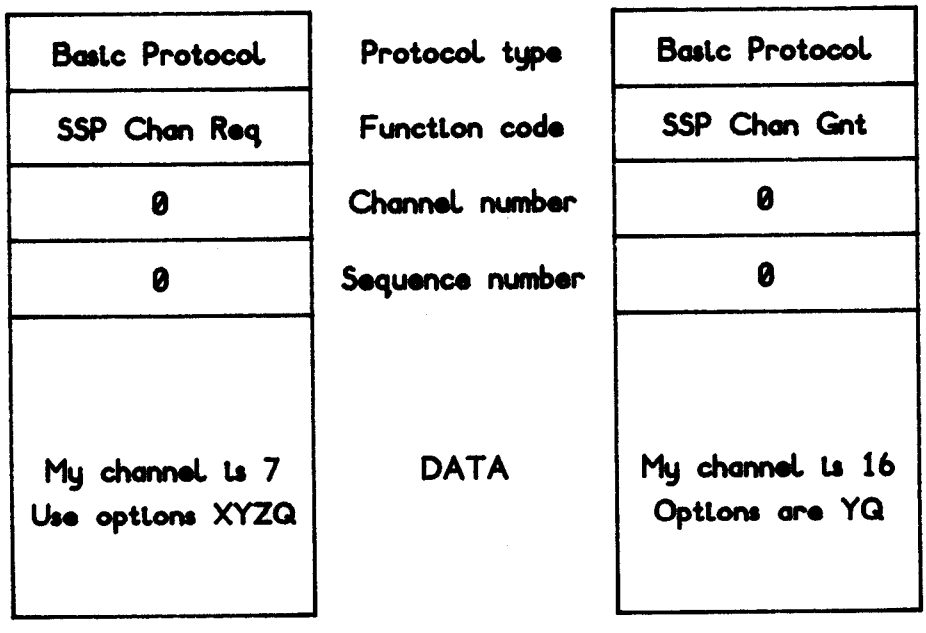



*Figure 8.3b Exchange minipackets setting up a SSP transaction*

Having set up channels at both ends the source now sends its message using the Transfer protocol. The message minipackets bear the channel number which the destination allocated and the destination sends acknowledgement minipackets to the channel allocated by the source. Data and acknowledgement minipackets for this stage are shown in Figure 8.3c

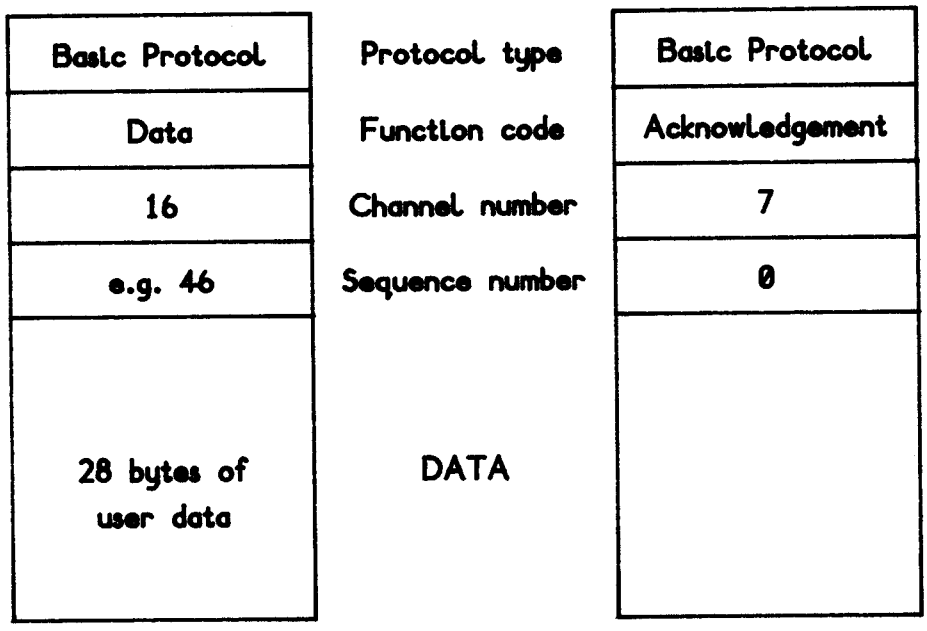



*Figure 8.3c* ***Data and acknowledgement minipackets***

When the message is received at the destination it sends its message back to the source, again using the Transfer protocol and the same pair of channel numbers. Finally, when this message has been sent, both source and destination free the channels and the transaction is complete.

## 8.4 DISCUSSION

The Basic Protocol provides a stream-like connection between two stations and facilities for single minipacket exchanges. These facilities are adequate for building protocols similar to those presently used on local networks. The construction of a SSP protocol was described. The implementation of a BSP type protocol is also possible, though it will be more complex than the SSP implementation.

One of the aims of any protocol should be that it be simple to implement on a wide range of hosts. In particular, it should allow high transfer speeds when the host and network are capable. The Basic Protocol is quite simple but is only the basis for higher level protocols which hosts will want to converse in. A number of decisions have been made in designing the Basic Protocol which constrict it in certain ways. It was decided that 256 channels would be enough at any host. Experience with current systems suggests that this will be adequate, around ten open byte streams being a

typical maximum. To this must be added a similar number of SSP transactions which use a channel for a short time and 256 channels are likely to be adequate for most applications.

A more uncertain choice is the limiting of block sizes to 256 minipackets. Many messages will require less than 256 minipackets and blocking will not be necessary for them. Very long messages will be sent more quickly if the block size is larger, assuming that no minipackets are lost. Increasing the block size to 65536 minipackets will cost another byte of each minipacket to make the sequence number larger. More minipackets will probably be required to send a message of given size therefore, but the number of acknowledgements will be considerably reduced.

To send 1MByte of data with the short (256) block size requires 35715 data minipackets and 140 acknowledgement minipackets assuming that no retransmissions have to be made. To send the same amount with a block size of 65536 requires 37038 data minipackets and 1 acknowledgement. The use of the negative acknowledgement scheme will be necessary with large block sizes, since there is a good chance that at least one minipacket from a block of many thousand will not be delivered. Using the larger sequence number field means that a minipacket will now hold 27 bytes of user data. This may be an inconvenient size when working with machines with word sizes greater than 8 bits since a whole number of 16 or 32 bit words will not fit exactly in the user data field.

Chapter 9

# NETWORK INTERFACES

This chapter describes the way in which computers may be connected to the CFR by a network interface. A network interface is a piece of hardware which connects some digital device (the host) to a network. In this chapter the various methods of interfacing are briefly described and an outline design for a high speed interface is given. This is intended to support the Basic Protocol described in the previous chapter.

## 9.1 INTERFACE METHODS

The function of an interface is to allow data to pass onto the network and to be taken from it. The data will normally be in the computer's memory and two methods are commonly used to transfer it to and from the network. The first is for the host to transfer it under program control. This will be done word by word and may be quite slow since several instructions will typically be required to move each word. The second method is to use Direct Memory Access (DMA). Here the host initiates the transfer of data but a separate piece of hardware actually moves the words around. The host computer can continue executing instructions while the transfer takes place, though there may be contention for the host's memory. It is often possible to make the two sets of accesses to the memory interleave so that one operation is not slowed by the other.

The first method is simple and cheap but is slower than the second. A further disadvantage is that the host may spend much of its time dealing with network transfers and will have less time to spend on its other computations. The DMA method, while complex, is faster and puts less load on the host. Using this method the transfers may be limited in speed by the access time of the hosts memory. This is typically between 50 and 500nS per word, but the time taken to acquire the memory bus from the host must also be taken into account. This term might be negligible if access is interleaved but could be several microseconds in other cases.

## 9.2 INTERFACES AND PROTOCOLS

The DMA interface removes from the host the need to explicitly manipulate each word which passes between it and the network. The comings and goings of packets of data and hence of these words, are controlled by the protocols of the network. The protocols are generally implemented in one of two places. The first is on the host itself. A program on the host performs all the necessary manipulations of packets and maintains timeout counters and other state information associated with each transfer which is in progress. Even with DMA to move data from packets to and from the host's memory, the load on the host brought about by performing the protocol operations can be substantial. This is particularly true with networks such as slotted rings, where data arrives frequently and in small quantities.

To reduce this processing load on the host a separate processor to perform parts of the protocol may be used. This is the second place in which protocols are commonly implemented. Such systems are described by Garnett and he gives them the name Network Interface Processors or NIPs [Garnett 83].

The processor of the NIP is entirely dedicated to performing network operations for the host. It can perform DMA operations on the host's memory and this method may be used not only for data transfer but also to pass commands and results between the two processors. The NIP signifies that it requires the host's attention by means of an interrupt.

There is a certain level of protocol above which implementation in the NIP is not worthwhile. Protocols in which the data must be scanned for markers or escape sequences are not worth implementing in the NIP since its speed comes from not having to look at each word which is transmitted.

The NIPs described by Garnett are used on the Cambridge Ring and use bipolar and MOS microprocessors as their processors. The MOS system cannot transfer data as fast as the network because the processor cannot keep up with arriving minipackets. The bipolar system can keep up with the fastest Cambridge Rings which transfer data between hosts at 1.6MBit/S or 100,000 minipackets per second. The MOS NIP implements protocols up to the SSP and BSP level, while the bipolar one only implements the Basic Block protocol.

To do justice to the data rates of the CFR, a NIP will have to be carefully designed and have special facilities for manipulating minipacket data at high speeds. A CFR NIP should implement at least the low level protocol (e.g. Basic Protocol) and may include higher level ones also.

## 9.3 SIMPLE CFR INTERFACES

Before dealing with high speed interfaces for the CFR the design of simple interfaces will be considered. Many hosts will communicate infrequently with the CFR and many will not require high transfer rates. For these hosts a simple slow interface will suffice.

A simple interface is defined to be one in which the host interacts directly with the CFR station. It thus handles individual minipackets and the words contained within them. The registers of the CFR station chip will be mapped into the memory or input/output space of the host and the host may access them by read and write operations.

The host will need to know when minipackets arrive and when they have been transmitted. To do this it can poll a register in the station and test for the appropriate bit changing. Alternatively the interrupt line of the station may be configured to interrupt the host when a minipacket arrives or is transmitted. The polling method uses many of the processors cycles, many of which will be effectively wasted. It is applicable when the host is a server which waits for requests from the network and performs some task as a result of them. The interrupt method is more generally applicable in that the host may execute programs and still be able to use the network on demand. There is a difference in the transfer rates which will be possible with the two systems, given hosts of similar processing power. The polled system is inherently faster than the interrupt system. This is because there is normally a time overhead with interrupt driven systems. The processor will normally change state in some way when an interrupt occurs, typically saving its registers in memory. This takes time which is not necessary in polled systems and thus if the time to enter, execute and leave the interrupt routine is long then the transfer rate will suffer accordingly.

The usefulness of each method depends on the application. If minipackets are expected in rapid succession then the polling method will lead to the higher transfer rate. If the interval between minipacket arrival is longer than the interrupt service time then the interrupt method will be preferable. A combination of the two may be possible in some circumstances. The design of such interfaces for Cambridge Rings is discussed in [Gibbons 80].

## 9.4 HIGH SPEED CFR INTERFACE

This section describes a NIP design for the CFR. A high speed interface may be expected to transfer data between the CFR and a host computer at rates in excess of 10MBit/S. In normal mode a single slot CFR clocked at 100MHz has a maximum point to point bandwidth of 40MBit/S and this falls to 26MBit/S in a two slot ring and to 20MBit/S in a 3 slot ring. Channel mode bandwidths on the same rings are 80MBit/S, 40MBit/S and 26MBit/S respectively. A ring with only one slot will probably contain a normal slot and thus the 80MBit/S rate will not be encountered in most configurations.

An interface capable of 40MBit/S transfer rates will cover most practical applications and even one allowing 10MBit/S will be too fast for the vast majority of hosts. Assuming that the Basic Protocol is to be used on the network, the following table gives the time allowed for certain fundamental operations at various data rates. These figures assume that a data minipacket carries 28 bytes of user data.

| P-to-P data rate (MBit/S) | Processing time allowed (uS) | | | |
|---|---|---|---|---|
| | Minipacket | Data (8 bit) | Data (16 bit) | Data (32 bit) |
| 40 | 6.4 | 0.23 | 0.45 | 0.9 |
| 20 | 12.8 | 0.45 | 0.9 | 1.8 |
| 10 | 25.6 | 0.9 | 1.8 | 3.6 |

Transmitting a data minipacket involves placing the Basic Protocol information (4 bytes) and the data (28 bytes) into the minipacket buffer in the CFR station. The destination address may also need to be written to the station. The data bytes will be obtained from the host by DMA, while the protocol information will be supplied by the interface processor.

Receiving a data minipacket is a rather more complex operation. The processor in the NIP will have to read the protocol information in the minipacket to decide whether to accept the data. It must check that the channel number is a valid one, interpret the function code, determine if the sequence number is valid, update a sequence counter and initiate a DMA operation if all else is correct. Additionally it may have to send an acknowledgement minipacket if the end of the block has been reached.

There is quite a lot of processing to be done at the minipacket level. The operations are quite simple but must be done very quickly. At a slightly higher level, but still within the Basic Protocol, a number of more complicated tasks must be performed. Dealing with channel allocation and deallocation, maintaining timeout counters and communicating with the host are quite complex operations by comparison with those performed on minipackets.

There is a case for having two processors within the CFR NIP. A simple fast one, the **minipacket processor**, for dealing with minipackets and a more sophisticated and probably slower one, the **protocol processor**, for dealing with those protocol operations which occur less frequently than minipacket arrival and departure. The minipacket processor can deal with the Transfer protocol and will refer any exceptional events to the protocol processor. It would, for instance, inform the protocol processor of the arrival of a negative acknowledgement. The protocol processor will deal with all minipackets arriving on channel zero. These should be minipackets of the Exchange protocol and will be dealt with on an individual basis. Similarly the protocol processor will initiate the transmission of such minipackets by the minipacket processor.

### 9.4.1 Minipacket Processor Requirements

The minipacket processor must do two things. First, it must receive minipackets of the Transfer protocol and copy the data that they contain to the host's memory. Second, it must take data from the host's memory and transmit that in minipackets. While doing these tasks it must be updating sequence numbers, validating channel numbers and performing other housekeeping operations. A certain amount of information must be stored within the NIP for each active channel. This includes the address of the other station involved in the transfer, its channel number, sequence numbers for transmission and reception and other status information. In addition there will also be information relating to the DMA operations which are associated with the channel and timeout counters for the current block. The protocol processor will

need to set up some of this information and so it must be accessible by both processors. The amount of information needed for each channel is quite small, 32 bytes will be more than enough. A table of the necessary pieces of information is given below.

| Name | Size (bytes) | Purpose |
|---|---|---|
| Status | 2 | Flags for channel (in use, retransmitting, awaiting protocol processor action, etc) |
| Address | 2 | Address of other station |
| Channel | 1 | Channel used at other station |
| Rx Seq no | 1 | Receive sequence number |
| Rx Timer | 2 | Receive block timeout counter |
| Rx DMA addr | 4 | Receive DMA address |
| Rx DMA count | 4 | Receive DMA byte count |
| Tx Seq no | 1 | Transmit sequence number |
| Tx Timer | 2 | Transmit block timeout counter |
| Tx DMA addr | 4 | Transmit DMA address |
| Tx DMA count | 4 | Transmit DMA byte count |

One of the status bits must say if the channel is in use or not and the collection of information for each channel can be kept in a memory set aside for the purpose. The size of this memory, the **channel state RAM**, will determine how many channels are possible at the station.

The actual minipacket processor has a number of simple operations to perform at high speed. This suggests two possible implementations. A dedicated hardware state machine would enable very high speeds to be achieved but would be expensive and take a long time to design. A system built around a bit-slice microprocessor would be rather slower but much more flexible. It would also be much easier to implement and this approach is the more sensible one to take at this stage.

The AMD 2900 series of bit-slice microprocessors are widely used in the sort of application considered here. A processor with a word size which is a multiple of four bits may be constructed using the basic four bit processing element. Processors with a 16 bit word can perform a microinstruction in less than 150nS. Since the CFR interface is 8 bits wide an 8 bit processor would be suitable. An interface which is capable of 40MBit/S needs to process a minipacket in under 6.4uS. This is time to perform 42 microinstructions which is probably just enough to perform all the operations required in minipacket reception. The DMA operations will be performed by

one of the available LSI controller chips and simply initiated by the minipacket processor.

### 9.4.2 Protocol Processor Requirements

The protocol processor must oversee the operation of the minipacket processor and deal with higher level operations. It can expect stimuli from three places. First, minipackets arriving on channel zero will be passed to it. These will be channel requests or similar control minipackets. Second, there will be requests from the host asking it to do various tasks and finally, there will be requests from the minipacket processor when it detects some exception it cannot deal with. Additionally, there will be routine tasks such as maintaining channel timeout counters.

The protocol processor may also implement higher level protocols than the Basic Protocol and therefore have to maintain state information concerned with these. These tasks are more suited to a conventional microprocessor than one with a bit-slice architecture. A fast microprocessor with a good interrupt facility is needed. Extensive arithmetic capability and a large addressing space are not needed per se.

The Motorola 68000 microprocessor would be a good choice. This is a 16 bit microprocessor with 32 bit internal registers and byte addressing capability. It has a wide range of useful peripheral chips including a DMA controller and timer/counter chips.

### 9.4.3 NIP Summary

The NIP contains two processors with associated control and scratchpad memory, the DMA system, the channel state RAM and the CFR station chips. The minipacket processor deals with all minipacket transfers. Channel zero minipackets are passed to the protocol processor and all others go via the DMA system. The channel state RAM is accessible by both processors, the minipacket processor maintains the data which is initially placed there by the protocol processor.

## 9.5 ERROR DETECTION

The checking of the correctness of data sent over a network is best done on an end-to-end basis. This implies that when a message is copied from one host's memory to another's, some method of comparing the two messages actually in memory should

be used to check that the transfer was successful. This can be done conveniently by computing a checksum of the message at both source and destination and sending the one from the source to the destination for comparison.

The checksum may be formed by the source NIP as words are transferred from the host's memory. At the destination the checksum can be formed as the message is written to the host. This method is less than perfect since it does not detect errors occurring while words are read from and written to the host's memory. A more rigorous approach is to have the hosts compute the checksum of the messages in situ. This involves much more work since in the previous method the checksum may be calculated on the fly in the NIPs.

To incorporate the former scheme into the Basic Protocol some changes will have to be made. The last minipacket of the message must contain the checksum and if this does not agree with that calculated by the receiving NIP, then a request to have the message retransmitted must be sent. Checksumming might alternatively be done at the block level. This is more efficient if a message is many blocks long but inefficient if a block consists of a small number of minipackets. The size of the checksum would typically be 16 or 32 bits.

## 9.6 DISCUSSION

The NIP design described allows the Basic Protocol to be implemented in a way which does not significantly load the host. By using two processors, each with different properties, sufficient speed and sophistication can be included to allow a variety of higher level protocols to be implemented while keeping transfer rates high. The hardware of the NIP will be standard with the exception of the DMA circuitry. This will have to be tailored to suit individual hosts and perform operations such as the packing and unpacking of bytes to and from words. The inclusion of a relatively powerful microprocessor as the protocol processor means that the NIP could be used in its own right as a server or processor on the network.

Chapter 10

# CONCLUSION

This final chapter reviews the Cambridge Fast Ring design and gives some ideas for related work which could be done in the near future.

## 10.1 DISCUSSION

In its early stages, the design of the CFR was influenced by that of the Cambridge Ring. A gradual divergence took place, however, and while the Mark 1 version of the CFR resembled the Cambridge Ring in many ways, the present design is far removed from its ancestor.

It is pertinent to ask if the slotted ring is a good architecture for a high speed network. Recent implementations of fast networks such as Macrolan and Hubnet [Lee-ES 83] have used star and tree architectures. The aim of the CFR project was to produce a general purpose network without special emphasis on ultra-high reliability. This being the case, any of the ring systems would have been adequate as would a star or tree. The requirement that the bandwidth be partitionable led naturally to the slotted ring and this has proved to be a highly suitable architecture for this application.

Given the imminent arrival of local networks standards, especially that of the IEEE, one might also ask if the CFR could and should be made to conform to this or some other standard. The IEEE standard specifies 48 bit addresses. It insists that rings use the token access method and does not specify clocking rates above 20MHz for rings. In short, the standard applies to some current implementations of local networks and does not cater for new network designs.

The CFR could be modified to have 48 bit addresses but that is the closest one could get without changing it into a token ring and thereby into a totally different network. Little would be gained by such an exercise and only the emergence of a more wide-ranging standard would make changes worthwhile.

### 10.1.1 Bandwidth Partitioning

There are two features of the CFR which are new and set it apart from current networks. One is the addressing scheme and routing of minipackets through bridges and the other is the method of bandwidth partitioning by slot differentiation. This latter feature has only been exploited to a limited degree in the CFR. Many schemes of slot marking are possible and the one chosen keeps the network hardware and software simple.

The channel transmission mode allows a transmitter to obtain a transmission path to another station which is unaffected by other traffic on the network. The bandwidth available is higher than can be achieved with the normal (Cambridge Ring style) transmission method and is also constant. A possible drawback with this mode of operation is that the transmitter is obliged to supply data at a certain rate. If it fails to do this or if the destination fails to receive a minipacket, then the "channel" is lost and may take some time to regain. If there is competition for channel slots, then a competing station may take the channel which has been lost and its original owner will not be able to get it back.

Certain applications, such as video transmissions, might find an alternative method preferable. One in which the failure to provide minipacket data or the failure to receive a minipacket did not cause the loss of the channel could be provided. Implementing such a scheme, in addition to the present one, would add to the complexity of the network hardware.

### 10.1.2 Bridges

Minipacket bridges considerably enhance the flexibility of the CFR. In addition to increasing network reliability and allowing large numbers of devices to be connected, they allow much larger networks to be constructed with a wide range of topological configurations. One scheme which has been suggested is the linking of two distant CFR networks by high speed duplex land lines, such as Megastream links running at 1MBit/S. The land lines can be used to form a ring with bridges onto the CFR networks at each end. If both networks use compatible protocols then the two will be able to communicate directly via the bridge link and the whole is now one large network. There must, of course, be no duplication of station addresses on the network.

One problem, common to many ring networks, which is particularly apparent with the CFR, is the ability of the host devices to cope with variations in the speed of the network and in the other hosts. The Cambridge Ring has a response which tells the source of a minipacket if the destination host is still busy processing the last minipacket. This forms a flow control mechanism which is not possible with the CFR since minipackets may now cross bridges. A related problem is that retransmissions of minipackets on the CFR are made at a rate dependent on the clocking rate of the ring. This will lead to problems at bridges when rings of widely differing clocking rates are connected. A range of retransmission rates have been provided for but the range is small and extra minipacket buffering may be needed at some bridges. The size of the buffers will be dependent both on the expected traffic through the bridge and the clocking rates of the two rings. In addition, the low level protocol may have to take ring speed and host processing speed into account.

## 10.2 FURTHER WORK

The present status of the CFR is that the logic design of both the CMOS and ECL chips is complete and samples of both devices are expected in the second half of 1984. There is a great deal of further work involved in producing networks using the CFR. Ideas for protocols and interfaces were given in this thesis but these have not been demonstrated in practice. The transmission of voice by local network has been done on other networks but will require further effort to implement on the CFR. Video transmission is relatively unexplored as far as LANs are concerned; there is much work to be done here.

The integration of computer, voice and video data onto a single network provides much opportunity for research, not only in terms of network considerations, but also the user interface. The presentation of voice and video with the computer data at a user's terminal provides plenty of scope for experimentation.

The Universe experiment has shown that wide area networks can be built which have many of the characteristics of LANs. The distinction between wide and local area networks is becoming one of size alone. With the CFR it is possible to construct much larger networks than was possible with an Ethernet or Cambridge Ring, for example.

The IEEE standards committee has expressed interest in metropolitan area networks (MANs) and the CFR could be used to network a small town to the same extent that telephone systems do at present. Research into making "local" networks larger is likely to continue and some of the design considerations may be different to those of existing networks. Networks which connect different organisations will have to be designed with privacy in mind and suitable encryption techniques will have to be found. Suitable addressing systems must be devised. Linked MANs will enable country-wide or even world-wide communication.

The materials with which to construct very high speed networks are rapidly becoming available. Fibre-optic transmission systems can be obtained, at a price, which are capable of 400MHz and more. ECL logic can currently be clocked at up to 500MHz and the speeds of both these technologies are likely to increase. The speed of "slower" logic families, such as CMOS, is increasing as manufacturing methods become more advanced. The construction of a 500MHz network similar to the CFR is likely to be possible in the next few years.

# REFERENCES


**Adams 82**
"The Interconnection of Local Area Networks via a Satellite Network"
Adams, G.C., Burren, J.W., Cooper, C.S. and Girard, P.M.
New Advances in Computer Systems,
ed. K.G. Beauchamp,
Reidel, 1982. (pp201-210)

**Banerjee 83**
"Aspects of Fast Local Area Networks"
Banerjee, R.
PhD thesis, Computer Laboratory, University of Cambridge. (in preparation)

**Biba 81**
"Packet Communication Networks for Broadband Coaxial Cable"
Biba, K.J.
Local Networks and Distributed Office Systems, Online, May 1981. (pp611-625)

**Binns 82**
"Further Developments on the Cambridge Ring Network at the University of Kent"
Binns, S.E., Dallas, I.N. and Spratt. E.B.
Proc. IFIP TC6 Symposium on Local Area Networks, Florence 1982,
ed. Ravasio, Hopkins and Naffah,
North Holland Publishing Company. (pp183-204)

**Boggs 80**
"PUP: An Internetwork Architecture"
Boggs, D.R., Shoch, J.F., Taft, E.A. and Metcalfe, R.M.
IEEE Trans. on Communications, vol COM-28, no 4, April 1980. (pp612-624)

**Bux 82**
"A Local Area Communication Network based on a Reliable Token Ring System"
Bux, W., Closs, F., Janson, P.A., Kummerle, K., Muller, H.R. and Rothauser, E.H.
Proc. IFIP TC6 Symposium on Local Area Networks, Florence 1982,
ed. Ravasio, Hopkins and Naffah,
North Holland Publishing Company. (pp69-82)

**Croker 72**
"An Experimental Interconnection of Computers Through a Loop Transmission System"
Croker, C.H.
Bell System Technical Journal, vol 51, no 6, July 1972. (pp1167-1175)

**Dallas 80**
"A Cambridge Ring Local Area Network Realisation of a Transport Service"
Dallas, I.N.
Proc. IFIP WG6.4 Workshop on Local Networks, Zurich 1980,
ed. West and Janson,
North Holland Publishing Company. (pp245-269)

**Dellar 80**
"Removing Backing Store Administration from the CAP Operating System"
Dellar, C.N.R.
Operating Systems Review, vol 14, no 4, Oct 1980. (pp41-49)

**Farmer 69**
"An Experimental Distributed Switching System to Handle Bursty Computer Traffic"
Farmer, W.D. and Newhall, E.E.
Proceedings of ACM Symposium on Problems in the Optimisation of Data Communications Systems, Oct 1969. (pp1-33)

**Garnett 83**
"Intelligent Network Interfaces"
Garnett, N.H.
PhD thesis, Computer Laboratory, University of Cambridge, Sept 1983.

**Gibbons 80**
"The Design of Interfaces for the Cambridge Ring"
Gibbons, J.J.
PhD thesis, Computer Labortatory, University of Cambridge, Sept 1980.

**Hafner 74**
"A Digital Loop Transmission System"
Hafner, E.R., Nenendal, Z. and Tschanz, M.
IEEE Trans. on Communications, vol COM-22, no 6, June 1974. (pp877-881)

**Hopper 79**
"Binary Routing Networks"
Hopper, A. and Wheeler, D.J.
IEEE Trans. on Computers, vol COM-28, no 10, Oct 1979. (pp699-703)

**Hopper 81**
"Cambridge Ring LSI System Specification 1.2"
Hopper, A.
Project Note, Computer Laboratory, University of Cambridge, Oct 1981.

**Hopper 83**
"Design and Use of an Integrated Cambridge Ring"
Hopper, A. and Williamson, R.C.
IEEE Trans. on Communications, Special Issue on Local Networks, Nov 1983.

**Jackson 81**
"SILK: An Integrated Voice and Data System"
Jackson, B., Nicholson, J., Roberts, B. and Snelling M.
Local Networks and Distributed Office Systems, Online, May 1981. (pp47-63)

**Johnson 80**
"Ring Byte Stream Protocol Specification"
Johnson, M.A.
Project Note, Computer Laboratory, University of Cambridge, Apr 1980.

**JNT 82**
"Cambridge Ring 82 Interface Specification"
"Cambridge Ring 82 Protocol Specification"
Joint Network Team.
Computer Board and Research Councils, Department of Education and Science, Nov 1982.

**Kropfl 72**
"An Experimental Data Block Switching System"
Kropfl, W.J.
Bell System Technical Journal, vol 51, no 6, July 1972. (pp1147-1165)

**Lee-ES 83**
"The Principles and Performance of HUBNET"
Lee, E.S. and Boulton, P.I.P.
IEEE Trans. on Communications, Special Issue on Local Networks, Nov 1983.

**Lee-R 83**
"Local Area Network Applications"
Lee, R.
MSc Project Report, London School of Economics, Sept 1983.

**Leslie 83**
"Extending the Local Area Network"
Leslie, I.M.
PhD thesis, Computer Laboratory, University of Cambridge, Feb 1983.

**Metcalfe 76**
"Ethernet: Distributed Packet Switching for Local Computer Networks"
Metcalfe, R.M. and Boggs, D.R.
CACM, vol 19, no 7, July 1976. (pp395-403)

**Needham 82**
"The Cambridge Distributed Computing System"
Needham, R.M. and Herbert, A.J.
Addison Wesley, London, 1982.

**Nelson 83**
"802: A Progress Report"
Nelson, J.
Datamation, vol 29, no 9, Sept 1983. (pp136-152)

**Ody 80**
"Logging of Ring Transactions at the Basic Block Level"
Ody, N.J.
Systems Research Group Note, Computer Laboratory,
University of Cambridge, Sept 1980.

**Penney 78**
"Survey of Computer Communication Loop Networks"
Penney, B.K. and Baghdadi, A.A.
Research Report 78/42, Dept of Computing and Control,
Imperial College, Sept 1978.

**Pierce 72**
"Network for Block Switching of Data"
Pierce, J.R.
Bell Systems Technical Journal, vol 51, no 6, July 1971. (pp1133-1145)

**Roberts 73**
"The ARPA Network"
Roberts, L.G. and Wessler, B.D.
Computer Communication Networks,
ed. Abramson and Kuo,
Prentice Hall, 1973. (pp485-500)

**Saltzer 80**
"Source Routing for Campus-Wide Internet Transport"
Saltzer, J.H., Reed, D.P. and Clark, D.D.
Proc. IFIP WG6.4 Workshop on Local Networks, Zurich 1980,
ed. West and Janson,
North Holland Publishing Company. (pp1-24)

**Shoch 79**
"Measured Performance of an Ethernet Local Network"
Shoch, J.F. and Hupp, J.A.
Local Area Communication Network Symposium, Boston, May 1979.

**Stevens 83**
"MACROLAN: A High Performance Network"
Stevens, R.W.
ICL Technical Journal, vol 3, no 3, May 1983. (pp289-296)

**Sumerling 82**
"A Low Cost Fibre Optic Link for Inter-rack Communication"
Sumerling, G.W. and Morgan, P.J.
Plessey Research (Caswell) Ltd., 1982.

**Walker 78**
"Basic Ring Transport Protocol"
Walker, R.D.H.
Project Note, Computer Laboratory, University of Cambridge, Nov 1978.

**Wilkes 79**
"The Cambridge Digital Communication Ring"
Wilkes, M.V.
Local Area Communication Network Symposium, Boston, May 1979.

**Whitehead 82**
"An Investigation into Cambridge Ring Traffic from File Server to 68000 Stations"
Whitehead, J.R.
Diploma Dissertation, Computer Laboratory,
University of Cambridge, July 1982.

**Zimmerman 80**
"OSI Reference Model - The ISO Model of Architecture for Open Systems Interconnection"
Zimmerman, H.
IEEE Trans. on Communications, vol COM-28, no 4, Apr 1980. (pp425-432)