Implementing the Orwell ATM protocol over an optical fibre ring

This item was submitted to Loughborough University’s Institutional Repository by the/an author.

Additional Information:

- A Doctoral Thesis. Submitted in partial fulfillment of the requirements for the award of Doctor of Philosophy of Loughborough University.

Metadata Record: [https://dspace.lboro.ac.uk/2134/14122](https://dspace.lboro.ac.uk/2134/14122)

Publisher: © Charles F. Nche

Please cite the published version.
This item was submitted to Loughborough University as a PhD thesis by the author and is made available in the Institutional Repository (https://dspace.lboro.ac.uk/) under the following Creative Commons Licence conditions.

For the full text of this licence, please go to:
http://creativecommons.org/licenses/by-nc-nd/2.5/
<table>
<thead>
<tr>
<th>AUTHOR/FILING TITLE</th>
<th>NČHE, C.F.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCESSION/COPY NO.</td>
<td>NYL052046</td>
</tr>
<tr>
<td>VOL. NO.</td>
<td></td>
</tr>
<tr>
<td>CLASS MARK</td>
<td>LOAN COPY</td>
</tr>
<tr>
<td></td>
<td>28 JUN 1996</td>
</tr>
<tr>
<td></td>
<td>21 MAY 1996</td>
</tr>
<tr>
<td></td>
<td>14 JUN 1996</td>
</tr>
<tr>
<td></td>
<td>27 FEB 1998</td>
</tr>
<tr>
<td></td>
<td>4 OCT 1996</td>
</tr>
<tr>
<td></td>
<td>2 OCT 1998</td>
</tr>
</tbody>
</table>
Implementing the Orwell ATM Protocol over an Optical Fibre Ring

by

Charles F. Nche

A Doctoral Thesis

Submitted in partial fulfilment of the requirement for the award of Doctor of Philosophy of Loughborough University of Technology

June, 1993
Supervisor : Dr. D. J. Parish

Department of Electronic and Electrical Engineering
University of Technology
Loughborough, England

© by Charles F. Nche, 1993
Dedicated to

My Parents

and

Mr. Fowajuh Che
ABSTRACT

Asynchronous Transfer Mode (ATM) has emerged as a possible contender for the next generation of digital data switching system and as the fundamental transport basis for the Broadband Integrated Services Digital Network, (B-ISDN) and telecommunications in the future. ATM networks can provide a high degree of flexibility. Due to the dynamic allocation of transmission and switching resources and the absence of a physical channel structure in such a network, services are not necessarily restricted to a particular bit rate. They may work at any bit rate within the limit of the network and can also support variable bit rate connections.

There is a growing need for networks to carry a greater range of traffic such as wide-band traffic (video), bursty traffic (variable bit rate video), short holding time traffic (facsimile) and low bandwidth traffic (voice). To cope with variations in demand, networks must also be capable of dynamically allocating their total capacity amongst the various traffic types. The traffic requirements can be met through the use of architectures based on Asynchronous Transfer Mode (ATM).

This thesis is concerned with the research, design and assessment of a high speed computer network based on a protocol proposed by British Telecom which is used to implement an ATM-based network referred to as the Orwell protocol. Throughout this thesis, the Orwell ring is discussed with particular emphasis to its bandwidth allocation mechanism. The first part of the thesis deals with the provision of a physical layer that operates at a bit rate of 155.52 Mbits/s. Three independent nodes connected by optical fibre to form the Orwell ring network have been developed with a maximum possible user access bit rate of 90.5 Mbits/s. Subsequent sections concentrate on the transmission of a constant bit rate video service with a capacity of 77.8 Mbits/s over the network.

In the ATM network environment where fixed sized cells are the basic transport unit, a packetiser is required to packetise the data at the transmitter into cells of
fixed length. These cells are then transported independently in the slots of the ring
and are routed by information contained in the cell headers to the appropriate
receiver where they are depacketised. Various packetisers and depacketisers are
discussed and one is described in details which is used in the transfer of the
constant bit rate video. Other PC based services using an alternative
packetiser/depacketiser design were supported to investigate the multimedia aspect
of ATM.

The main aim of the research was to demonstrate that an ATM network could
support high bandwidth services and guarantee this bandwidth irrespective of other
services on the network. The use of the Orwell protocol demonstrated that this was
possible and further showed that ATM could be the most flexible and efficient
network to implement B-ISDN.
ACKNOWLEDGEMENTS

I would like to thank my supervisor, Dr. D. J. Parish for his support, guidance and encouragement and Mr. P. R. Robinson and Ivan Thomas from British Telecom Research Laboratories for their help throughout the course of the research.

I would like to acknowledge the financial support provided to me by the Government of the Republic of Cameroon and British Telecom.

I would also like to extend my special thanks to Mr. C. Rodgers for all his relentless help and discussions throughout the research period and for being a good friend.

My thanks also go to all my friends and colleagues in W242 who have helped me in one way or the other to see me through the research period. Mrs P. Coventry is acknowledged for her valuable contribution toward the completion of this research.

I would like to express my gratitude to all my family and friends in Cameroon who have been waiting and praying for me to finish my studies.

Finally, The "Members", Associates and Rosemarie Dixon are acknowledged for their moral support.
# TABLE OF CONTENTS

List of Symbols and Acronyms viii
Definitions and Terminology xii

Chapter 1: Introduction
1.1 Broadband Integrated Services Digital Network 1
1.2 The Orwell Ring 2
1.3 Aims 3
1.4 Thesis Organisation 4

Chapter 2: Switching Methods
2.1 Introduction 6
2.2 Circuit Switching 6
2.3 Packet Switching 8
   2.3.1 Store and Forward Switching 8
2.4 Hybrid Systems 11
2.5 Integrated Services Digital Network (ISDN) 13
2.6 Broadband - ISDN 17

Chapter 3: ATM Networks
3.1 Introduction 20
3.2 B-ISDN Functional Architecture of an ATM 21
   3.2.1 The ATM Layer and the Cell Header 23
   3.2.2 ATM Adaptation Layer 25
3.3 Distributed Queue Dual Bus (DQDB) 27
   3.3.1 DQDB Network Architecture 28
   3.3.2 The Distributed Queueing Protocol 30
3.4 Fibre Distributed Data Interface (FDDI) 32
   3.4.1 Frame and Token Formats 34
   3.4.2 Physical Layer (PHY) Operation 35
   3.4.3 Token MAC Functional Operation 36
3.5 The Orwell Ring Protocol
3.6 Analysis of HSLAN and MAN Implementations
   3.6.1 The Physical Protocol Block
   3.6.2 The Media Access Block
3.7 Conclusion

Chapter 4 : Orwell Ring ATM based Network
4.1 Introduction
4.2 The Orwell Protocol
4.3 The Slot Structure
   4.3.1 Start Field
   4.3.2 Control Field
   4.3.3 Address Field
   4.3.4 Information Field
4.4 Node Design
   4.4.1 PHIC
   4.4.2 Election
   4.4.3 Monitor
   4.4.4 Communications Control Processor (CCP) Interface Logic
4.5 Physical Layer Tx and Rx Boards
   4.5.1 Coding/Decoding
   4.5.2 Parallel to Serial and Serial to Parallel Converters
   4.5.3 Clock and Slot Synchronisation
   4.5.4 Clock and Data Recovery Circuit
   4.5.5 Optical Fibre
4.6 Summary of the Physical Layer
   4.6.1 Tx Section of the Physical Layer
   4.6.2 Rx Section of the Physical Layer

Chapter 5 : Implementing Real-Time Video Transfer on a Fibre-based ATM Network
5.1 Introduction
5.2 Transmitter Station
   5.2.1 Video Tx Sub-System
REFERENCES

APPENDIX I  
Software controller for IEEE488 interface  133

APPENDIX II  
Software packetiser interface routines  154

APPENDIX III  
Technical details of the PHIC device  172

APPENDIX IV  
Circuit diagrams  187
## List of Symbols and Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Full Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAL</td>
<td>ATM Adaptation Layer</td>
</tr>
<tr>
<td>ADC</td>
<td>Analogue to Digital Converter</td>
</tr>
<tr>
<td>ANSI</td>
<td>American National Standard Institute</td>
</tr>
<tr>
<td>AR</td>
<td>Auto-Reset</td>
</tr>
<tr>
<td>ARPANET</td>
<td>Advanced Research Project Agency NETwork</td>
</tr>
<tr>
<td>ATD</td>
<td>Asynchronous Time Division</td>
</tr>
<tr>
<td>ATM</td>
<td>Asynchronous Transfer Mode</td>
</tr>
<tr>
<td>AU</td>
<td>Access Units</td>
</tr>
<tr>
<td>B-ISDN</td>
<td>Broadband Integrated Services Digital Network</td>
</tr>
<tr>
<td>BTRL</td>
<td>British Telecom Research Laboratories</td>
</tr>
<tr>
<td>CBO</td>
<td>Constant Bitstream Oriented</td>
</tr>
<tr>
<td>CBR</td>
<td>Constant Bit Rate</td>
</tr>
<tr>
<td>CCS</td>
<td>Consultative Committee on International Telephony and Telegraphy</td>
</tr>
<tr>
<td>CCP</td>
<td>Communication Control Processor</td>
</tr>
<tr>
<td>CCS</td>
<td>Common Control Switching</td>
</tr>
<tr>
<td>CID</td>
<td>Connection IDentity</td>
</tr>
<tr>
<td>CLR</td>
<td>Cell Loss Rate</td>
</tr>
<tr>
<td>CLP</td>
<td>Cell Loss Priority</td>
</tr>
<tr>
<td>CODEC</td>
<td>COder and DECoder</td>
</tr>
<tr>
<td>CPCS</td>
<td>Common Part Convergence Sublayer</td>
</tr>
<tr>
<td>CRC</td>
<td>Cyclic Redundancy Check</td>
</tr>
<tr>
<td>CS</td>
<td>Convergence Sublayer</td>
</tr>
<tr>
<td>DA</td>
<td>Destination Address</td>
</tr>
<tr>
<td>DAC</td>
<td>Digital to Analogue Converter</td>
</tr>
<tr>
<td>DCS</td>
<td>Distributed Control Switching</td>
</tr>
<tr>
<td>DIL</td>
<td>Dual In Line</td>
</tr>
<tr>
<td>DLE</td>
<td>Data Link Escape</td>
</tr>
<tr>
<td>DQDB</td>
<td>Distributed Queue Dual Bus</td>
</tr>
<tr>
<td>DRAM</td>
<td>Dual port Random Access Memory</td>
</tr>
<tr>
<td>DTACK</td>
<td>DaTa ACKnowledgement</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>ECL</td>
<td>Emitter Couple Logic</td>
</tr>
<tr>
<td>ETSI</td>
<td>European Telecommunication Standard Institute</td>
</tr>
<tr>
<td>ETX</td>
<td>End of TeXt</td>
</tr>
<tr>
<td>FC</td>
<td>Frame Control</td>
</tr>
<tr>
<td>FCS</td>
<td>Frame Check Sequence</td>
</tr>
<tr>
<td>FDDI</td>
<td>Fibre Distributed Data Interface</td>
</tr>
<tr>
<td>FIFO</td>
<td>First In First Out</td>
</tr>
<tr>
<td>FMBS</td>
<td>Frame Mode Bearer Service</td>
</tr>
<tr>
<td>FPS</td>
<td>Fast Packet Switching</td>
</tr>
<tr>
<td>FS</td>
<td>Frame Status</td>
</tr>
<tr>
<td>FTP</td>
<td>File Transfer Protocol</td>
</tr>
<tr>
<td>GFC</td>
<td>Generic Flow Control</td>
</tr>
<tr>
<td>HDTV</td>
<td>High Definition TeleVision</td>
</tr>
<tr>
<td>HEC</td>
<td>Header Error Control</td>
</tr>
<tr>
<td>HSLAN</td>
<td>High Speed Local Area Network</td>
</tr>
<tr>
<td>IDN</td>
<td>Integrated Digital Network</td>
</tr>
<tr>
<td>IMP</td>
<td>Interface Message Processors</td>
</tr>
<tr>
<td>ISDN</td>
<td>Integrated Services Digital Network</td>
</tr>
<tr>
<td>ISLN</td>
<td>Integrated Services Local Network</td>
</tr>
<tr>
<td>ISO</td>
<td>International Standard Organisation</td>
</tr>
<tr>
<td>Kbits/s</td>
<td>Kilo-bits per second</td>
</tr>
<tr>
<td>LLC</td>
<td>Logical Link Layer</td>
</tr>
<tr>
<td>MAC</td>
<td>Medium Access Control</td>
</tr>
<tr>
<td>MAEN</td>
<td>MAp ENable</td>
</tr>
<tr>
<td>MAN</td>
<td>Metropolitan Area Network</td>
</tr>
<tr>
<td>Mbits/s</td>
<td>Mega-bits per second</td>
</tr>
<tr>
<td>NISDN</td>
<td>Narrowband Integrated Services Digital Network</td>
</tr>
<tr>
<td>NNI</td>
<td>Network Node Interface</td>
</tr>
<tr>
<td>NRZ</td>
<td>Non Return to Zero</td>
</tr>
<tr>
<td>NRZI</td>
<td>Non Return to Zero Invert</td>
</tr>
<tr>
<td>OSI</td>
<td>Open System Interconnection</td>
</tr>
<tr>
<td>PA</td>
<td>Preamble</td>
</tr>
<tr>
<td>PABX</td>
<td>Private Automatic Branch eXchange</td>
</tr>
<tr>
<td>PAL</td>
<td>Phase Alternating Line</td>
</tr>
</tbody>
</table>
PC  Personnal Computer
PCM  Pulse Code Modulation
PDU  Protocol Data Unit
PHIC  Protocol Handler Integrated Circuit
PHY  Physical Layer Protocol
PLL  Phase Locked Loop
PMD  Physical Layer Medium Dependent
PMSI  Periodic Mark Space Insertion
PRM  Protocol Reference Model
PROM  Programmable Read Only Memory
PSTN  Public Switched Telephone Network
PT  Payload Type
PTT  Postal Telephone and Telegraph
QOS  Quality Of Service
QPSX  Queued Packet Synchronous eXchange
RACE  Research for Advance Communication in Europe
RAM  Random Access Memory
RI  Reset Interval
RR  Reset Rate
Rx  Receiver
SA  Source Address
SAR  Segmentation And Reassembly
SD  Starting Delimiter
SDH  Synchronous Digital Hierarchy
SMT  Station Management
SSCF  Service Specific Coordination Function
SSCOP  Service Specific Connection Oriented Protocol
SSCS  Service Specific Convergence Sublayer
STX  Start of TeXt
SYN  SYNonchronisation
TELCO  TELecommunication COmpany
TDM  Time Division Multiplex
TTL  Transistor Transistor Logic
TTR  Time Token Rotation

- x -
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TV</td>
<td>TeleVision</td>
</tr>
<tr>
<td>Tx</td>
<td>Transmitter</td>
</tr>
<tr>
<td>UNI</td>
<td>User Network Interface</td>
</tr>
<tr>
<td>μ</td>
<td>Micro</td>
</tr>
<tr>
<td>VBR</td>
<td>Variable bit rate</td>
</tr>
<tr>
<td>VCI</td>
<td>Virtual Channel Identifier</td>
</tr>
<tr>
<td>VCO</td>
<td>Voltage Control Oscillator</td>
</tr>
<tr>
<td>VPI</td>
<td>Virtual Path Identifier</td>
</tr>
</tbody>
</table>
Definitions and Terminology

Asynchronous Transfer Mode
A packet and connection oriented transfer mode using time division multiplexing technique where the information flow is organised in fixed size data transport units called cells.

Auto-Reset
A mechanism which resets the d-val allocation of the CBO services if no reset is received within a pre-defined interval.

Call Acceptance Threshold
The reset rate below which call blocking commences.

Call Blocking
Refers to the situation when a new call request is rejected because of insufficient bandwidth on the ring to support the new call, without degrading the service performance of the existing connections.

Cell
Is the data transport unit in ATM technique. It consists of a header field, which contains routing and control information, and an information field for carrying user data, both fields are fixed sizes.

d-val
Unit in which bandwidth is allocated on the Orwell Ring.

Load Control
The mechanisms which decide whether a new call request or a request for more bandwidth allocation can be accepted or not. The decision is based upon whether the increased load will cause the ring to overload. E.g. call blocking, dynamic d-val allocation.
MAP
MAP is a name given to two 64 bit RAMs, MAP1 and MAP2, which are used as a routing table. Addresses for MAP1 are taken from the first 6 bits of an incoming slot’s DA, and MAP2 is addressed by the second 6 bits.

Monitoring Interval (of the dynamic d-val allocation scheme)
The interval over which cell arrival rate is monitored.

Orwell Torus
A switch where rings are stacked in parallel. The Orwell protocol operates across the whole Torus making it appears as a single high capacity ring.

Overload Control
The mechanisms which protect the ring against overload conditions.

Packetisation
The process whereby the user data is segmented into packets with a header appended to it to be transmitted over packet switched networks.

Depacketisation
The process whereby the header is stripped off from the receiving slot.

Reset Interval
Time elapsed between two consecutive ring resets. Maximum reset interval is the pre-determined interval which should not be exceeded, e.g. 125 μs

Reset Rate
The number of resets observed on the ring within a given time interval, e.g. 2 ms. This provides a measure of the amount of unused bandwidth on the ring.

Switch
Equipment used to connect and distribute communications between a trunk line or backbone and individual nodes. Sometimes it is used to represent a network.
VBR Video
Video signals which are coded such that the data rate is proportional to the spatial and/or temporal complexity of the picture at any given moment, thus maintaining a constant picture quality.

Video Coding
The application of digital image processing technique to remove redundancy in the video signals, in order to reduce the signal bandwidth for transmission economy.
Chapter 1

Introduction

1.1 Broadband Integrated Services Digital Network

Public networks will continue to evolve in order to provide additional services and to make use of emerging technologies. Not only is there a steady increase in demand for communication over long distances with more and more people wishing to communicate with other people or indeed machines, e.g. computers or databases, but ever larger volumes of information are being exchanged per unit of time. Demand for transmission systems with high transmission capacities along the entire route between subscribers therefore, is growing rapidly. The current proliferation of multiple application-oriented networks is inefficient, expensive and unlikely to satisfy these long term requirements. As the need for new applications grows, the limitation of current systems will become increasingly significant. The ISDN was at best a stop-gap measure that postponed the problems for a few years. An effective solution requires the development of flexible information transport networks, capable of providing connections of arbitrary bandwidth and potentially with multiple end-points.

To support the bearer services mentioned above, a protocol at layer 2 of the Open Systems Interconnection (OSI) Reference Model is required. Many such protocols have been developed for local data networks[85,94] and some have included features for handling delay sensitive services such as voice and video[66,69,88]. However, these are not well suited to the requirements of a public network. The reduced cost of optical fibre links has made possible larger bandwidth and an increase in the maximum distance between nodes. To cope efficiently with variation in demand, networks must have sufficient bandwidth and also be capable of dynamically allocating their total capacity amongst the various traffic types. Realisation of this fact has led to plans for the establishment of a uniform international network for all services entitled Broadband Integrated Services Digital
Network (B-ISDN). The packet-based Asynchronous Transfer Mode (ATM) has been accepted by the CCITT and other international bodies as the basic network to implement B-ISDN. In an ATM environment, fixed size cells are the basic transport unit.[14]

Much has been published on the techniques available to combat the problems associated with handling multimedia services but they have mainly been confined to theoretical exercises based on computer simulation.[18,35,36] This thesis is concerned with the design and operation of an ATM based network using the Orwell protocol.

1.2 The Orwell Ring

The Orwell Ring is a high speed slotted ring implementing the Orwell medium access control protocol. The protocol was developed at British Telecom Research Laboratory (BTRL) as a flexible protocol for carrying mixed services such as voice, data and possibly low bit rate VBR video on a single multi-service network. It was later developed as a high capacity switch suitable for carrying the above mentioned services and variable bit rate video. The ring supports a connection-oriented, cell-based transport mechanism which complies with the definition of ATM. A major throughput improvement has been achieved because of its destination deletion policy and thus doubling the accessible bandwidth or halving the ring speed for a given capacity.

Delays in accessing the ring are kept within pre-determined limits, typically 125μs or 2ms depending on the size of the network, and network management so that with minimal buffering the ring offers a transparent path to constant bit rate traffic. An inherent overload control mechanism allows the ring to behave as a lost call system unlike conventional LANs which are lost packet systems. The load control mechanism is fully distributed and is at the heart of the Orwell protocol, guaranteeing synchronous services small cell delay and high cell security, whilst allowing unused bandwidth to be allocated in a flexible and efficient manner by delay tolerant asynchronous services and other low priority services such as VBR.
video. The load control mechanism operates a bandwidth reservation scheme and provides an estimate of the unused bandwidth on the ring. By controlling access to the ring from the nodes according to their bandwidth reservation, and rejecting call requests that might cause the ring to overload, cell delay can be bounded. The protocol further provides overload control on non-priority services in order to protect the synchronous services. There is also provision for broadcasting and multicasting information.

The protocol can be extended to operate on systems consisting of a number of slotted rings connected in parallel, making their separate capacities appear as a single high capacity ring. This multiple ring system is called an Orwell Torus. It can provide a means for the addition/removal of a node while maintaining existing connections. It also provides recovery mechanisms to maintain existing connections when fault conditions cause partial loss of the switch capacity. Because of its enhanced capacity and high flexibility for handling different service mixes, the switch can be used for public network applications. More information on the Orwell ring and Torus can be found in references.[1,2,3,34]

1.3 Aims

The purpose of this research was to demonstrate the enormous potential of ATM to support multimedia services. However, the provision of very high bandwidth user connections may be required e.g for broadcast video distribution, and if this could be provided, many lower rate connections would if anything, be easier to support, since most ATM bandwidth will be shared between users.

The initial stage of the research involved the design and implementation of a high-speed ATM network based on the Orwell protocol. Subsequently a system was developed to support a constant bit-rate video transfer at 77.8 Mbits/s, and was meant to provide essentially standard TV quality images, with no loss of quality between scene changes and to demonstrate the potential to provide high guaranteed bandwidth to future multimedia users. It was intended that lower rate services such as computer data would be supported to demonstrate the multimedia aspect of the
network. Experiments were to be carried out to gain some insight into the performance of the network.

1.4 Thesis Organisation

Chapter 2 reviews the existing switching methods adopted over the years with particular emphasis to circuit and packet switching. A brief overview of ISDN and the network evolutionary path toward the broadband integrated services digital network is given.

Chapter 3 describes the current state of Asynchronous Transfer Mode (ATM) development within standard and the basic principles of ATM. It describes three protocols (DQDB, FDDI and Orwell) for high-speed LANs and MANs from an implementation standpoint with emphasis placed on the lowest layers of ISO OSI Reference Model, i.e the physical layer and the media access control sub-layer of the data link layer. A brief survey and classification of the above MAC protocols for HSLANs and MANs are given.

Chapter 4 details the design and implementation of a high speed network operating at a physical layer speed of 155.5 Mbits/s. The Orwell protocol is reviewed with particular attention paid to the slot structure and bandwidth allocation mechanism. A section is included on the synchronisation of the nodes to the line-rate clock.

Chapter 5 describes a functional system that allows real time video images to be transmitted over the network. The main building blocks of the system are described. A CODEC is discussed which performs straightforward pulse code modulation linear sampling of a composite PAL signal.

Chapter 6 describes some of the packetisation and depacketisation methods and details those designed and used in this network.

Chapter 7 presents an overall view of the working system. It describes some experiments conducted to assess the performance of the network.
Chapter 8 contains a discussion of the work and presents the conclusions drawn from the research.
Chapter 2

Switching Methods

2.1 Introduction

Telecommunication networks have long been based on analogue technology dedicated to a telephone service. When a telephone call is made, the switching equipment within the telephone system seeks out a 'physical' path between the caller and the receiver. The interconnection of the subscriber's line is done at a switching centre (a telephone exchange). There are many such exchanges which are interconnected by transmission media (trunks). There have been three principal eras of switching based mostly upon the technology: Manual, Electromechanical and electronic. Each of these eras has been well dealt with in other papers.[5,6,31] In a manual exchange, holding a connection during a call and breaking the connection at the end is carried out by the operator inserting and withdrawing plugs from the switchboard.

2.2 Circuit Switching

Early automatic exchanges involved the used of hard wire logic to establish the programming of exchange operation. These exchanges used analogue switching involving, usually, step-by-step rotary contactors. Typical of such contactor is the two-motion selector where stepping mechanisms are used to move a contactor onto the required contact and it is locked in that position until released at the end of the call. This approach is referred to as Distributed Control Switching and is characterised by the sequential setting of contactors by dialled digits in a rigidly defined manner. The distributed control lasted for some years and became unsuitable for the requirements of what was then a large, modern exchanges. This then called for flexible structuring of exchange operation and efficient use of switching plant. The concept of Common Control Switching was introduced
whereby an incoming directory number was register-stored and used to establish a 'best path' through a switching matrix. Matrix switching systems such as crossbar and reed-relay usually rely on a marking technique to established a path through a series of switching stages. The two ends of the connection are determined by the exchange control and the different potentials applied at these points. A selection is made if more than one through path can be found and the connection is then established through the crosspoints corresponding to the marked path. The connection is held either by mechanical locking (crossbar), by magnetic remanence (latching reeds) or by an additional contact (non-latching reeds).[43]

As electromechanical systems progressed from directly controlled gross motion switches to common control with fine motion switches much progress was made in technology and techniques for the functions. Signalling was one function that improved greatly not only in reliability but also in range. The signalling function took place from station to the signalling system in which case it was known as 'station signalling' or between switching offices also known as 'interoffice signalling'. The transition from an analogue to a digital system (thereby including trunk and international, as well as junction circuits in the PCM transmission of voice signal) was made because of the enormous advantages that the digital system could offer. Some of the advantages included: the reduction of installation size, power requirement, easier setting up and reduced maintenance requirements. Easier coding for transmission reliability, security and privacy. Enhanced versatility in terms of the variety and nature of the information sources that were handled and the manner in which the handling took place. With the introduction of digital transmission facilities it became possible to allocate separate bits for signalling.[5]

The successful application of electronics to reduce the cost and extend transmission distance encouraged the study of switching technology. All previous switching centre networks used separate physical paths in space and each pair of wires or switching crosspoint within the exchange carried only one speech circuit and is called 'space division' switching, whilst the switching of speech samples with electronics that are simultaneously shared by other calls is known as 'time division' switching.
The above switching techniques mentioned so far were first introduced for telephone networks and are now known generally as 'circuit switching'. Circuit switching is regarded as providing through connections for the exchange of messages. It can also be defined as applied to a system with a switching centre network used for two-way transmission with no chance of transmission delay due to passage through the network. The message may be voice, data (including telex, teletypewriter), or video. Circuit switching applies to switching where a continuous two way path is established in space, time, or frequency of the bandwidth suitable to the calling and called stations for the duration of the call. The delay time through the network is constant once the path is established. It is an efficient way to handle voice since a constant amount of bandwidth is required for the duration of the call.[6,116]

2.3 Packet Switching

The early demand for switched data services arose largely from the high cost of computers. Time sharing by remote terminals was accepted by many users to make most effective use of expensive data processing equipment. The Public Switched Telephone Network (PSTN) was pressed into service to provide the necessary switched access. However as computer technology developed it became clear that the PSTN would not be adequate for many of the applications (e.g teletex, electronics funds transfer, electronic mail etc) and that dedicated switched data networks would be required. As time went on, computer-based equipment became cheap and widespread and data communication was motivated more by the need to share information than to share equipment. For many applications the changing balance of costs increasing favoured packet switching.

2.3.1 Store and Forward Switching.

One of the early forms of packet switching was store and forward switching. It was used in a variety of forms to provide message switched services, in which users could exchange complete messages (often very long messages) with the advantage
of delayed delivery and retransmission if the message was lost in transmission. Communication between computers or between computers and terminals usually involves the transfer of 'blocks' of data. Packet switching exploits the fact that data blocks may be transferred between terminals without setting up a continuous end-to-end connection. Instead they are transmitted on a link-by-link basis, being stored temporarily at each switch en route where they are queued for transmission on an appropriate outgoing link. Routing decisions are based on control information contained in a header prefixing each data block. The store and forward nature of packet switching brought a number of important advantages compared with circuit switching. By using redundant coding methods, transmission errors were detected and corrected by retransmission on a link-by-link basis. This increased the store needed at each switch since packets were not removed from the transmit queues until their correct receipt had been acknowledged by the next switch en route. Since transmission capacity was dynamically allocated to each user on a demand basis, very efficient use was made of transmission plant. On inter-switch trunks, packets from different users were interleaved to achieve high utilisation. Similarly on dedicated access circuits the user may interleave packets for different transactions. This feature provided a strong incentive to introduce packet switching where transmission costs were high e.g. North America where circuits tend to be long. [11,31]

The need for further advances in switching came with developments in computers. Even in early computers, a form of switching was employed to permit access to the store from the central processing unit and later, as more peripherals were added, a highway or bus system was employed. A simple polling arrangement was often used and since the equipment was all located in the same area, adequate control could be exercised by providing extra wires. As computer systems came to be distributed firstly within the same premises and later over very wide areas, the switching problem became more important. Although data transmission was possible over the telephone network, the economics and operational problems of this approach lead to a search for new techniques. The result was the development of packet switched networks such as ARPANET. This was the first large scale packet switched network and was developed to link the large number of computer research projects sponsored throughout America by the Advanced Research Project
Agency (ARPANET).[7]

The ARPANET IMP-IMP protocol really corresponds to a mixture of the layer 2 and layer 3 protocol of the ISO OSI Reference Model (Fig. 2.1). When an IMP wants to send a frame to an adjacent IMP, it sets it up in a memory buffer and then starts the transmission hardware. When a complete frame has been sent an interrupt is generated. The transmission hardware sends a SYN (SYNchronise), a DLE (Data Link Escape) and an STX (Start of TeXt) before sending the first character in the buffer. These three characters serve to define the start of frame for the receiver. Finally the hardware transmits a DLE and an ETX (End of TeXt) followed by 24 bit CRC checksum and then another SYN character. If there is no new frame to transmit, the hardware keeps sending SYNs.[98]

<table>
<thead>
<tr>
<th>ISO</th>
<th>ARPANET</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Application</td>
</tr>
<tr>
<td>6</td>
<td>Presentation</td>
</tr>
<tr>
<td>5</td>
<td>Session</td>
</tr>
<tr>
<td>4</td>
<td>Transport</td>
</tr>
<tr>
<td>3</td>
<td>Network</td>
</tr>
<tr>
<td>2</td>
<td>Link</td>
</tr>
<tr>
<td>1</td>
<td>Physical</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Terminal Network</th>
<th>Terminal Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMP - IMP</td>
<td>Physical</td>
</tr>
</tbody>
</table>

Fig. 2.1 Approximate correspondence between ISO and ARPANET.

When a message comes into an IMP from a host, the IMP first builds a 128 bit header and attaches it to the front of the message, forming a packet. The routing algorithm then takes over and decides which output line to use. When the packet has worked its way to the front of the queue, the IMP hardware transmits it,
encased by the frame header and trailer, as shown in Fig 2.2. Control, routing, tracer and other special packets have different formats.

**Fig. 2.2 Format of an ARPANET single frame message while "on the wire".**

### 2.4 Hybrid Systems

Circuit switching has traditionally been an efficient way to handle voice since between the times of starting and finishing a call a constant amount of bandwidth is required. Where only a few simultaneous calls are to be handled on one link, committing one channel to handle each voice stream exclusively at just under half utilisation of the channel has been better than providing the intelligence to switch packets of voice along uncommitted channels. Data traffic however, has some properties that are fundamentally different from telephone conversations. Typical examples of data traffic are a keystroke or a screen update sent between computers and remote terminals, or a file transfer between two filing systems. In each case the
traffic is bursty, so that fixed channel allocation will be very wasteful. Unlike voice traffic where some modest level of error is of no consequence (since it will not be noticed by the listener), data transmission must be error free, implying a requirement for protocol layers quite different from anything needed for voice. Whilst voice requires very small and strict transmission delays, the exact timing of a terminal screen refresh is of little importance. In particular the information is perfectly comprehensible whenever it arrives.

Voice traffic needs continuous use of low bandwidth channels whereas data traffic needs intermittent use of a high bandwidth channel. Packet switching is more suitable for data traffic since packet switching acquires and releases bandwidth as it is needed whilst circuit switching statistically reserves the required bandwidth in advance. In packet switching any unused bandwidth maybe utilised by other packets from unrelated sources going to unrelated destinations since circuits are never dedicated. With circuit switching any unused bandwidth on an allocated circuit is just wasted. Thus packet switching is more attractive for data, since in a packet network it is feasible to have many connections set up simultaneously but idle for much of the time. The available bandwidth will be divided at any instant between those connections actually engaged in transmission bursts. The network may occasionally become overloaded and have to discard traffic. Data transmission protocols are in general designed to cope with loss of packets. It is understood that from time to time the network becomes heavily loaded and each transmitter will obtain very little bandwidth. This approach will be quite unacceptable for voice. A lost call is preferred as far as voice is concerned but once a call is set up it must be allowed to proceed unhindered until the caller terminates it which means that the full call bandwidth must be available on demand. This mode of operation is natural to circuit switching systems.[52,53]

Given the very different requirements of voice and data, the National (Postal) Telephone and Telegraph operators (PTT) have tended to force data into the circuit switched mode because it is the minority traffic by a significant margin. Other networks such as the ARPANET have done the reverse. Several hybrid systems have been produced which have packet based structures. At the hardware level some proportion of the bandwidth is reserved for voice traffic with the rest
allocated to more bursty load which provides a good basis for transmission of integrated media. As computer and communication technology move closer together, hybrid forms of circuit switching and packet switching became possible. Voice and data services differ markedly in bit rate characteristics and the switching delay tolerated.[11,64,66]

The range of multiplexing strategies available for variable bit rate operation led to a corresponding set of switching architectures. Examples include bus (Ethernet) and ring (Token rings) networks.[98]

So far the discussion has concentrated on a basic network that provides a fixed bandwidth channel (or delivers bits) between specified terminals. A telecommunication 'service' such as telephony, picture transmission or data can then be mounted on the basic network. The concept of specific services mounted on a basic network has been aided by developments in the computer field, particularly the OSI Reference Model for data interchange.[94] This models the communication process into a set of hierarchical layers (Fig. 2.1). Each layer interacts with the layer below it down to the physical connection and at the same time, defines the protocols for use between one system and another within the layer.

A network can be regarded in two ways. At the lower levels (mainly levels 1-3) of the OSI Reference Model, the network provides the very basic communication capacity to transfer bits (assuming a digital network) between one point and another. A network at the higher level may comprise the users of a particular telecommunications service. Furthermore, the information may be processed in some way in the higher level network and this may interact with the lower level network.

2.5 Integrated Services Digital Network (ISDN)

A telecommunication network can only realise its full potential if it is available over a wide geographical area and it would be extremely expensive to establish a new network for each telecommunications service. Thus the trend is towards the
establishment of networks that can carry a wide range of telecommunications services.

With the advances in technology, the Plenary Assembly of the CCITT (Consultative Committee on International Telephony and Telegraphy) adopted the I series recommendations dealing with Integrated Services Digital Network (ISDN) matters. CCITT stated that an ISDN "is a network that provides end-to-end digital connectivity to support a wide range of services, including voice and non voice services, to which users have access by a limited set of standard multipurpose user-network interfaces".[8] It can be seen from this general definition that ISDN can be read to mean at least two distinct things. Services which are provided in an integrated manner (i.e. through a single access) over one or several digital networks or an integrated digital network which provides a range of services to the user. Such an ISDN standard interface was defined (and called "basic access"), comprising two 64 Kbits/s B channels and a 16 Kbits/s signalling D channel. Another type of interface, the "primary rate access", with a gross bit rate of 1.5 Mbits/s or 2 Mbits/s, offers the flexibility to allocate high speed H channels or a mixture of B and H channels respectively.[24,37,102]

From the CCITT definition of ISDN it emerges that the basis is a network which evolves from the telephony IDN. The IDN here means the 'Integrated Digital Network' where the term integrated refers to the commonality of digital techniques used in transmission and switching systems in the main part of the network. In particular the telephony IDN, based on 64 Kbits/s adopted for the encoding of speech, has been considered since the 1970s to be a powerful means of conveying other services in addition to voice, such as data and video. When access to this IDN is provided digitally to the user then we have ISDN. The above principle applies to other types of IDN (e.g. a data network). This means that ISDN includes all types of switching capabilities within the concept. The telephony IDN therefore consists of an interconnection of digital switching nodes, where circuits operating at 64Kbits/s are circuit-switched. These digital exchanges are linked together by a signalling network which is based on common channel signalling principles. Digital multiplexing hierarchies have been built up internationally, but unfortunately different schemes are used in different regions of the world. Thus the IDN with its
basic elements of 64Kbits/s circuit switching, common channel signalling, digital transmission hierarchies, together with digital access from the customer’s terminal to the local exchange, forms the basis of the ISDN concept.[79,100]

The main feature of ISDN is the support of a wide range of voice and non-voice services in the same network. A key element of service integration in an ISDN is the provision of a limited set of multi-purpose user-network interfaces as well as a limited set of multi-purpose connection types each of which support a wide range of services. An ISDN may support both non-switched and switched connections; switching techniques include both circuit and packet.

In essence ISDN is a series of protocols agreed by the PTT, by which terminal equipment carrying some communications medium (such as telephony, data file transfer or facsimile) can be connected to a public network and communicate with equipment handling the same medium at another (remote) location served by a public network. Within the ISDN recommendations the concept of a service is understood to mean: 'Services supported by ISDN are the communication capabilities made available to customers by telecommunication service providers’ or network operator and which are accessed by users either at an ISDN interface or within a terminal connected to the ISDN. So in all cases the service is something which is offered, sold, rented etc to the customer. This definition allows a clear distinction to be made between the service itself and what is happening in the network. Because of the wide range of services likely to be offered on an ISDN, the CCITT has developed in recommendation I.210 a hierarchical structure to the definition of services in the ISDN. Firstly there are two main categories of telecommunication services- bearer and teleservices.[4,38]

An ISDN bearer service is defined as ’a provision of such a service that allows him/her the capability for information transfer between ISDN access points according to CCITT standards. The higher (i.e above layer 3) terminal functions are defined by the customer. A teleservice is defined as a service that provides with the complete capability including defined terminal equipment functions, for communication with another user of the service according to protocols established by CCITT. Secondly there is a concept of basic service and supplementary service.
The basic service is what the customer gets when he/she asks for a communication capability. A supplementary service modifies or supplements the basic service, for example, redirection of calls to another number. In dedicated networks these supplementary services are known as user facilities.[62,108]

Most of the work on ISDN has been geared towards the use of PTT networks (which are largely the existing telephone networks plus a little extra infrastructure to handle particular non-telephony loads) for a combination of different media. The ISDN is mainly seen as a wide area network but some ISDN work deals with more local facilities, based around the Private Automatic Branch Exchange (PABX) sited on a customer premises. Whilst the PTTs have been using some of their telephone network capacity to carry data traffic, some experiments have also been done on the use of computer networks to carry voice traffic.[46,102]

Because of the diversity of the interpretations of the ISDN concept a very functional approach has been adopted in producing ISDN international recommendations. The first area which was tackled for the purposes of producing such standards was what is called the user-network interface (UNI). This interface is a point where the physical, electrical and logical characteristics can be defined. Examples of various possible user-network interfaces are shown in Fig 2.3. The

![Fig. 2.3 ISDN user-network interface examples.](image-url)
figure illustrates that what is meant by a user-network interface is the point at which terminal equipment, PABXs, LANs or private networks are connected to the (public) ISDN.[23]

Although the basis of ISDN is the 64 Kbits/s circuit switched connection, the same principle will apply to what is known as Broadband ISDN where bit rates of the order of greater than 100 Mbits/s will apply, and various switching modes will be used.

2.6 Broadband ISDN

The evolution of common carrier networks to provide in the metropolitan and long distance areas high speed multimedia communication and information services is represented by the Broadband Integrated Services Digital Network (B-ISDN). Enhancement of service capability is directed in the first place to make progressively available very high speed transmission and switching facilities by deployment of optical fibre links and very high speed and capacity switching fabrics. By fairly sharing the broadband network resources among the various media (voice, data, image and video) applications, the B-ISDN is directed towards the provision of an integrated services transport mechanism. The true integration of services in the B-ISDN is then guaranteed by an integrated network management and operation capability and by the offering via network data-bases of "intelligent network" features to allow customer control to be added to the basic service facilities.[47,115]

Not only is there a steady increase in demand for communication over long distances and more and more people wishing to communicate with other people or indeed machines, e.g. computers or databases, but ever larger volumes of information are also being exchanged per unit of time. Demand for transmission systems with large transmission capacities along the entire route between subscribers, is therefore growing rapidly. Realisation of this fact has led to plans for the establishment of a uniform international network for all services entitled Broadband Integrated Services Digital Network (B-ISDN).
The primary triggers for evolving toward the B-ISDN include the demand for new high bit-rate services, and a potential reduction in the costs of supporting current services on the telephone network. These needs are made possible by the availability of high speed semiconductor and lightwave technologies and by the availability of advanced communication techniques. This network is viewed as the communications umbrella which will stimulate the need for switches, capable of handling data rates well in excess of 64Kbits/s, the limit of today narrow band ISDN switches.[96]

Spurred by the high transmission capacity of optical fibres, many applications which require much higher bandwidth than possible in present networks have emerged[48,95]. Some of these applications are computer related (e.g. communications among remote supercomputers jointly performing a task), others involve the transmission of images and video signals. While fibre optic technology provides the necessary bandwidth for transmission purposes, the creation of a network that can provide bandwidth services to the users remain a challenge. One of the difficulties encountered comes from switching. The traffic requirements can be met through the use of architectures based on Asynchronous Transfer Mode (ATM). ATM has been widely accepted as the basic network of the future used to implement broadband integrated services digital network.[86,99]

The basic idea of the ATM is simplified connection-oriented packet switching to enable an enhanced throughput. A connection-oriented mode allows the use of fixed routing during data phase, resulting in the possibility of applying special purpose hardware switching fabrics to route the packets, and greatly facilitates the load control in the network. This definition places the switching in the nodes at the bottom of layer 2 of the OSI Reference Model stack. It also allows the use of packets with an enhanced functionality in the packet header such as e.g., time stamping and priorities. This enhanced functionality is required to limit the end-to-end delay of services with high delay constraints because these systems work at a moderate bit rate.

Another approach to define the ATM is to start from time division multiplexing (TDM). This approach is called Asynchronous Time Division (ATD). Information
flows are multiplexed and switched using short and fixed entities called cells. Each cell is labelled with a short header, logically identifying a connection. The label has only local significance and is established at call set-up. In the switching nodes incoming labels are mapped to outgoing labels. It is therefore reasonable to place the switching on top of the layer 1 of the OSI Reference Model stack. Due to the very small complexity, these systems can run at a very high bit rate.[28,42,101]

The CCITT describes telecommunications data rates (bandwidths) in the following way;[95]

<table>
<thead>
<tr>
<th>ISDN network</th>
<th>Interface bit rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Narrowband</td>
<td>up to 64 Kbits/s</td>
</tr>
<tr>
<td>Wideband</td>
<td>64 Kbits/s to 100 Mbits/s</td>
</tr>
<tr>
<td>Broadband</td>
<td>over 100 Mbits/s</td>
</tr>
</tbody>
</table>

However, this does not mean that a system with broadband data rates can be regarded as a Broadband-ISDN. The whole concept of ISDN is that the services are integrated. Therefore, a B-ISDN must be able to support Narrow and Wideband services as well as Broadband services.
Chapter 3

ATM Networks

3.1 Introduction

This chapter gives a brief overview of the current state of ATM development within standards and the basic principles of ATM, which is the B-ISDN solution defined by CCITT, ETSI and ANSI. Emerging multimedia, high speed data and imaging applications are generating a demand for the public network to be able to multiplex and switch simultaneously a wide spectrum of data rates. To support this need, national and international standards bodies such as the Consultative Committee for International Telephony and Telegraphy (CCITT) are formulating new telecommunication standards such as the Asynchronous Transfer Mode (ATM). The objective of ATM is to offer a fully integrated flexible switching and multiplexing mechanism by which existing circuit-switched and packet-switched traffic can be processed simultaneously. The existing circuit type and packet type traffic handling capabilities in ATM are called continuous bit stream oriented (CBO) and burst oriented service, respectively. Real time media such as voice and video can be handled through the burst oriented service as well as the CBO service. ATM, a statistical multiplexing and switching method which is based on fast packet switching concepts, is a radical departure from the circuit switching techniques that are now used by today digital switches. A wide variety of high- and low-bandwidth networking services can be supported simultaneously in an ATM switch.[26,41,54]

ATM represents convergence of two switch concepts developed in the 1980s. Asynchronous Time Division (ATD) multiplexing and Fast Packet Switching (FPS). The first of these was developed to increase the flexibility of circuit switching and the second from technology and protocol enhancements arising from packet switching techniques. Several factors have influenced the definition of broadband ATM switching architectures:[68,77]
- The need to accommodate a wide range of traffic types from voice to video to data;
- The high speed at which the switch has to operate (from 155.52 Mbits/s to over 1.2 Gbits/s);
- The bursty nature of the information streams passing through the ATM switching systems;
- The definition of ATM with a small fixed size cell and a limited header functionality has an important influence on the definition of optimal ATM switching architectures.

3.2 B-ISDN Functional Architecture of an ATM

The Protocol Reference Model (PRM) for the B-ISDN has evolved through several phases during the standardisation process. Unlike the Narrowband ISDN (N-ISDN), for which the user plane contains generally only a layer 1 physical layer

![Fig. 3.1 B-ISDN Protocol Reference Model.](image)
functionality, with higher layer network functions contained only in the control plane, B-ISDN has a significantly richer protocol stack for the support of all services. Although the relationship between ATM and the Open Systems Interconnection Reference Model was not made clear, the layers of the B-ISDN PRM are shown in Fig. 3.1 above. The three layers shown are physical, ATM, and Adaptation.

In this model the ATM layer is service independent and transmission medium independent. It supplies data transmission capabilities for signalling or network management as well. This layer service remains basically packet-oriented: it maintains clock and data service independence at the access, but on the other hand it does not perform synchronisation and it introduces disturbances such as cell jitter and cell loss or intrusion.[21,25,105] See Fig 3.2 below.

Below the ATM layer is the physical layer which consists of the physical medium sublayer and the transmission convergence sublayer. The physical medium sublayer provides bit transmission capability and includes functions that are dependent on

<table>
<thead>
<tr>
<th>Cell - Oriented</th>
</tr>
</thead>
<tbody>
<tr>
<td>. Clock Independence</td>
</tr>
<tr>
<td>. Semantic Independence</td>
</tr>
<tr>
<td>. Cell Sequence Integrity</td>
</tr>
<tr>
<td>. No Error Monitoring on User Data</td>
</tr>
<tr>
<td>. No Flow Control at Cell Level</td>
</tr>
<tr>
<td>. Variable Propagation Delays : Cell Jitter</td>
</tr>
<tr>
<td>. Cell Loss or Intrusion</td>
</tr>
</tbody>
</table>

Fig. 3.2 ATM Layer Services
the physical medium used. In principle, ATM is operable on any transmission medium (fibre optic, copper cable, microwave, etc) and on any physical topology. Therefore, that layer should perform transmission network termination functions, including a Medium Access Control (MAC) protocol. It might also include some medium dependent error monitoring capability, so as to provide the ATM layer with a given quality of service (QOS). Above the ATM layer is the adaptation layer. It provides extra functions that may be needed to fulfil the exact service requirements. As an example, Continuous Bitstream Oriented (CBO) services need an adaptation layer to emulate a "circuit-like" connection on top of the ATM layer.

3.2.1 The ATM Layer and the Cell Header

The ATM layer provides the carriage of all services on the B-ISDN using small fixed size cells. In principle, the protocol data unit (PDU), i.e. the cell, reflects that functional architecture. The cell basically consists of two fields, the header field and the information field, respectively allocated to the ATM and to the adaptation layer. The header, used by the ATM layer to route the cell from one access point to another, only includes routing informations (GFC, VPI, VCI). (Fig. 3.3). This is

<table>
<thead>
<tr>
<th>GFC</th>
<th>VPI</th>
<th>VCI</th>
<th>VCI</th>
<th>HEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPI</td>
<td>VCI</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Key:
- PT: Payload Type
- RES: Reserved
- CLP: Cell Loss Priority
- GFC: Generic Flow Control (at UNI only)
- VPI: Virtual Path Identifier
- VCI: Virtual Channel Identifier
- HEC: Header Error Check

Routing and Addressing

**Fig. 3.3 BISDN ATM cell structure.**
basically regarded as a logical channel identifier (or virtual channel identifier, VCI), the aim of which is to identify a communication channel on a given transmission entity.

The standard cell size of ATM is a 53-octet fixed length cell (or packet) consisting of a 48-octet information field to carry the user or network information and a 5-octet header that contains information used to route the cell to the destination. For some services, the information field may be left partially empty. The term 'octet' is used instead of 'byte' to emphasise that the bits are independent of one another and that they are always 8 bits wide. A key element in the standardisation of ATM is the header functionalities associated with each cell.[105] CCITT Recommendation I.361 details the following main header functions (see Fig. 3.3) each reflecting an ATM layer protocol function:[15,65]

- A routing or address field to carry the Virtual Channel (VC) or Virtual Path (VP) identification of each ATM cell.
- A Generic Flow Control (GFC) field that may be used at the User Network Interface (UNI) to control the flow of user information within the customer premises network to the public network.
- A payload type field
- A Cell Loss Priority (CLP) indicator
- A Header Error Control (HEC) field to protect the integrity of the header information.

The ATM layer is common to all services, but varies in header functionality between the UNI and the Network Node Interface (NNI). ATM cells may be transferred from one user to the other in different ways. For example, at the UNI, ATM cells may be carried in an externally framed transmission system (such as Synchronous Digital Hierarchy (SDH)) or in a cell-based transmission structure.[61] Within the network, ATM cells will also be mapped into the payload of existing plesiochronous transmission systems as well as SDH. The ATM layer supports the Adaptation Layer (AAL) which provides different functionality to accommodate various services.[22]
3.2.2 ATM Adaptation Layer

The ATM Adaptation Layer (AAL) provides the link between the services to be carried on B-ISDN and the generic ATM cells used by the B-ISDN. It offers an OSI layer 2 type service to the layers above plus some functions usually seen in layer 1. For some applications, the AAL will be minimal or non-existent. The AAL is placed in the information field of the cell, and as such it is carried transparently from end to end across the ATM network. Functions of the adaptation layer includes adapting the data stream structure to the transfer structure (i.e to map the user data flow into ATM cells: cell assembly and disassembly), in restoring when needed the timing properties (end-to-end synchronisation) or end-to-end error handling of the source data stream. It thus provides the functionality that will generally depend on the class of service concerned.

The AAL has been divided into two sub-layers; Convergence Sub-layer (CS) which is service dependent and copes with the special requirements of various higher protocol layers, and Segmentation And Reassembly sub-layer (SAR) which breaks down large messages or streams into cells and put them back again. For data or message based services, this has been further subdivided. The CCITT recommendation I.362 and ETSI outlines four AAL types that have been standardised. Type 1 is used for the support of constant bit rate, connection-oriented services that have a timing requirement between source and destination. It will also be used to support existing voice services, new constant bit rate video services and provides circuit emulation when required. This AAL Type 1 allows the full flexibility of ATM to be used, but does not utilise the potential ATM efficiencies. Type 2 AAL is used to support variable bit rate, connection-oriented services that also have timing requirement between source and destination. Examples include variable bit rate video for interactive or distributive services. This Type 2 AAL allows the full flexibility and efficiency of ATM to be utilised.[16,65]

Type 3 AAL is used to support connection-oriented variable bit rate services with no timing relationship. This includes data services such as X.25 and Frame Mode Bearer services (FMBSs), signalling, and future high speed data services. Type 4 is
used to support connectionless data services such as those supported by LANs and MANs. The newest AAL on the scene is the so-called Simple, Efficient Adaptation Layer (SEAL), usually known as Type 5. This is expected to be accepted within the standardisation very soon. It was proposed as being simple to implement and more bandwidth efficient on reliable links. The internal structure of the AAL Type 5 is shown in the same diagram as Type 3/4 (Fig. 3.4). The current trend of standardisation is to have Type 1/2, Type 3/4 and Type 5. The use of Type 1/2 has been standardised for class A services, Type 3/4 for class D services and Type 5 for class C and some of class D services.[57]

![Diagram of AAL Service Specific Convergence Sublayer](image)

**Fig. 3.4 Framework of AAL Service Specific Convergence Sublayer.**

SSCF  Service Specific Coordination Function
SSCOP  Service Specific Connection Oriented Protocol
CS  Convergence Sublayer
SAR  Segmentation And Reassembly Sublayer
SSCS  Service Specific Convergence Sublayer
CPCS  Common Part Convergence Sublayer
Thus the AAL consists of a sublayer which provides cell segmentation and reassembly to interface to the ATM layer and also a more service-specific convergence function to interface to the bearer services being carried. The Quality Of Service needed at the service level on the basis of the ATM network layer service can be provided in this layer. In addition to supporting the carriage of different services, the AAL has a key role in the interworking of B-ISDN with different networks and services. For example, the interworking of telephony services would use AAL Type 1.

3.3 Distributed Queue Dual Bus (DQDB)

The CCITT has already defined the integrated services digital network (ISDN) and is defining the broadband ISDN (B-ISDN). The International Standard Organisation (ISO) is standardising protocols and procedures so that different equipments can communicate with each other. The IEEE in 802 is bridging the gap between private and public networks. It is defining local area network (LANs) suitable for customer premises or campuses and Metropolitan Area Networks (MANs) that will interconnect LANs and provide other integrated services as part of the public network. The IEEE defined MAN was an evolutionary path to a full B-ISDN. A technology proposal that has TELCO support is one put forward from Australia. It is the dual bus Queued Packet Synchronous Exchange (QPSX) MAN. It is now widely known as Distributed Queue Dual Bus (DQDB).[60,78]

DQDB is a distributed switch/network that will fulfil the requirement of a public MAN. The switch architecture of DQDB is based on two contra-directional buses and a multiplicity of intermediate nodes as shown in Fig. 3.5. This allows full duplex communications between each pair of nodes of the switch. All nodes can read and write to both buses. At the end of each bus is a frame generator. Each frame consists of a header followed by a number of slots. The communications on DQDB are integrated, providing asynchronous, packet-oriented traffic of computer networks (packet switch) and the isochronous traffic (circuit switch) services. The two switch modes function independently and the total capacity of the switch is shared flexibly and completely between them. The access of packets to the buses of
QPSX is controlled by a unique media access control (MAC) protocol referred to as distributed queueing protocol. The distributed queueing protocol provides access characteristics independent of network size and speed.[109,110]

3.3.1 DQDB Network Architecture

The principal components of the DQDB architecture are a head station, two unidirectional buses and a multiplicity of access units (AUs). These can be configured as a point-to-point link (with two nodes positioned at either end of the bus) as in Fig. 3.5 or as a looped bus, where the frame generators at the ends of the bus are located in the same piece of equipment. A looped bus structure have the advantage that the bus can continue functioning in the event of a break. All nodes on the bus have the ability to generate frames. If a break occurs, the bus is made continuous through the old head-of-the-bus node, and the node on either side of the break will take over as frame generator. See Fig. 3.6. The head station generates frame synchronisation on the forward bus and the end station generates the frame pattern at the same rate on the reverse bus. The maximum length of a looped or open bus is a function of the bus bit rate and the number of nodes. For a
point-to-point bus, the maximum length is purely a function of the transmission system.[27]

Access units are attached to both buses via read and write connections. The writing onto the bus is by logical OR between the data arriving from upstream and the data from the unit. The read connection is placed ahead of the write connection and allows all data to be copied from the bus prior to modification by the unit's own writing. The bus architecture of QPSX was chosen for its reliability. The data bus passes directly by each node in the network. Hence, in QPSX, nodes can fail or even be removed from the network with no consequence on the operation on the rest of the network.

Communications on the DQDB buses are within the frame format shown in Fig. 3.7. The frame repetition period is 125 microseconds, matching that of digital telephony. The frame is subdivided into a fixed number of equal size units called slots. Slots provide the subdivision between isochronous and non-isochronous
traffic. A slot may be allocated to contain either type of communication type. Isochronous traffic, which requires a fixed, pre-determined bandwidth is carried in permanently assigned slots. Each octet within a slot provides a 64 Kbits/s isochronous channel. Asynchronous data packets carried on the DQDB are first segmented into fixed length segments of 53 bytes, comprising a 48 byte information field and a 5 byte header. These segments are written into non-assigned slots on the DQDB, access to these slots being controlled via the distributed queue MAC protocol. For non-isochronous communications, a slot once claim for access by a node, is used to transfer a single segment of a packet. The segmentation of packets into slots and the reassembly of packets at the destination is by a source-identifier-linked segmentation and reassembly protocol. The frame structure for DQDB is very similar to that for B-ISDN as standardised by CCITT. The slot size and the slot header size are aligned with that adopted by CCITT for B-ISDN cell size and cell header size.

3.3.2 The Distributed Queueing Protocol

Distributed Queueing is a media access control packet switching scheme that controls the access of fixed length data segments to the slots on the QPSX bus. The maximum throughput of the network is not constrained by the network speed or size and the protocol operates on the highly reliable dual bus architecture. The operation of the Distributed Queueing is fundamentally different from all existing MAC protocols. In all other access schemes, no continuous record is kept in the nodes that explicitly indicate the state of the network. Hence, when in these protocols a node has a packet to transmit, it must first derive the information from
the network as to when to access. This leads directly to a sensitivity in almost all protocols to the size of the network. The larger the network, the longer the delay in obtaining the access information. This limits the size over which the protocols may operate efficiently.

By contrast, with Distributed Queueing a current state record is kept in every node which holds the number of segments awaiting access to the bus. When a node has a segment for transmission, it uses this count to determine its position in the distributed queue. If no segments are waiting, access is immediate, otherwise deference is given only to those segments that queued first. With the queueing of segments for the bus, slots are never wasted. This guarantees minimum access delay at all levels of loading. This performance is achieved with negligible control overhead, in fact only two bit per slot and is effectively independent of network size and speed.

The operation of the basic distributed queueing algorithm is illustrated in Fig. 3.8. Each node is either idle, when there is nothing to transmit or otherwise count-down. The protocol uses two control bits, busy and request (REQ). The busy control bit

![Fig. 3.8 Distributed Queueing Algorithm.](image-url)
indicates whether or not the slot is occupied by a segment, while the REQ bit is used for sending requests for future segment transmission. When a node has a segment for transmission on the forward bus, it sends a single request on the reverse bus by setting \( \text{REQ} = 1 \) in the first slot with \( \text{REQ} = 0 \). This bit will pass to all upstream nodes, where upstream is defined in relation to flow on the forward bus. This REQ bit serves as an indicator to the upstream nodes that an additional segment is now queued for access. The request counter (RQ_CTR) increases by one for each REQ received on the reverse bus and decreases by one for each empty slot in the forward bus.

When a node becomes active, it transfers the contents of the RQ_CTR to a count_down counter (CD_CTR) and resets RQ_CTR to zero. The CD_CTR is decreased by one for every empty slot in the forward bus until it reaches zero. Immediately afterwards, the node transmits the segment into the first empty slot of the forward bus. In the meantime, the RQ_CTR increases by one for each new REQ received in the reverse bus. Thus with the use of two counters in each node, one counting outstanding access requests and the other counting down before access, a first-in-first-out queue is established for access to the forward bus. The queue formation is such that a slot is never wasted on the network if there is a segment queue for it. This is guaranteed since the CD_CTR count in the queue nodes represents the number of segments queued ahead.[17,20,27,74]

### 3.4 Fibre Distributed Data Interface (FDDI)

The Fibre Distributed Data Interface (FDDI) is a proposed ANSI for a 100 Mbits/s token-passing ring that uses optical fibre for transmission between stations and has dual counterrotating rings to provide redundant data paths for reliability. FDDI was originally proposed as a packet switching network with two primary areas of application: first as a high performance interconnection among Mainframes, and among Mainframes and their associated mass storage subsystems and other peripheral equipment and second, as a backbone network for use with lower speed LANs such as IEEE 802.3, 802.4 and 802.5. An enhancement to FDDI called FDDI-II would add a circuit switching capability to the existing packet capability,
expanding the field of application of FDDI to include those requiring the integration of voice, video and sensor data streams.[60,87] A packet service is a service where the elements of data to be transferred are placed in frames. Packets may vary in length and self-defining in that each contains delimiters that mark its beginning and end and address that specifies the target station. FDDI packets are called frames.

FDDI is emerging from ANSI as a standard for the lowest two layers (the physical and media access control levels) of the ISO Reference Model for Open Systems Interconnection, and is gaining wide support within the computer and communications industries. Fig. 3.9 shows the component entities necessary for an FDDI station. Identified components, conforming to both the IEEE 802 structure and the OSI concept of layering are: Station Management (SMT), which specifies the local portion of the network management application process, including the control required for the proper internal configuration and operation of a station in an FDDI ring; Media Access Control (MAC), which specifies the lower sublayer of the data link layer, including the access to the medium, addressing, data checking, and data framing; Physical layer protocol (PHY), which specifies the upper sublayer of the physical layer, including the encode/decode, clocking and framing

![Fig. 3.9 FDDI relationship to OSI model.](image-url)
for transmission; and Physical Layer Medium Dependent (PMD), which specifies the lower sublayer of the physical layer, including power levels and characteristics of the optical transmitter and receiver, interface optical signal requirements, the connector receptacle footprint, the requirements of conforming optical fibre cable plants, and the permissible bit error rates.[88,112,114]

3.4.1 Frame and Token Formats

Information is transmitted on the FDDI ring in frames which are variable in length. Tokens are special short fixed-length "frames" that are used to signify the right to transmit data. Fig. 3.10 shows the frame and token formats. The Preamble field, consisting nominally of IDLE symbols (a maximum frequency signal that is used for establishing and maintaining clock synchronisation), precedes every transmission. The Starting Delimiter (SD) field consists of two symbols sequence (JK) that is uniquely recognizable independent of previously established symbol boundaries. The SD establishes the symbol boundaries for the content that follows.

The Frame Control (FC) is a two-symbol element that defines the type of frame and its characteristics. It distinguishes between synchronous and asynchronous frames, the length of the address field (16 or 48 bits), and the kinds of frame (e.g., LLC or SMT). One set of FC values is reserved for implementer frames that have no defined format and are to be repeated unchanged by all conforming FDDI stations.

![Fig. 3.10 Frame and Token Formats of FDDI.](image)
The FC field also provides for two kinds of tokens, restricted and non-restricted. The latter is used in a special class of service which provides for extended dialogues among a limited set of cooperating stations. Two Ending Delimiter (ED) symbols (TT) complete a token.

The Destination Address (DA) and Source Address (SA) fields may be either 16 or 48 bits long, depending on the FC value. DA may be either an individual or a group address, the latter of which has the potential to be recognized by more than one station. The 32-bit Frame Check Sequence (FCS) field is a cyclic redundancy check using the standard polynomial used in the IEEE 802 protocols. The information field of a frame, like the other fields covered by the FCS check, consists only of data symbols. The ED field of a frame is one delimiter symbol (T). It is followed by the Frame Status (FS) field that has a minimum of three control indicators symbols that are modified by the station as it repeats the frame. These indicate, when set, that an error has been detected in the frame by the station, that the addressed station has recognized its address and that the frame has been copied by the station.

3.4.2 Physical Layer (PHY) Operation

PHY provides the protocol and the optical fibre hardware components that support a link from one FDDI station to another. PHY simultaneously receives and transmits. The transmitter accepts symbols from the MAC, converts these to five-bit code groups, and transmits the encoded serial data stream on the medium. The receiver recovers the encoded serial data stream from the medium, establishes symbol boundaries based on the recognition of a start delimiter, and forwards decoded symbols to the MAC. Additional symbols (QUIT, IDLE and HALT) are interpreted by PHY and used to support Station Management (SMT) functions.

PHY also provides the bit clock for each station. The total ring including all stations and links, must remain the same apparent bit length (i.e., no bit may be created or deleted) during the transmission of a frame around the ring. In the face of jitter, temperature, voltage and component aging effects, such stability can only...
be realised through special provision. PHY provides an elasticity buffer which is always inserted between the receiver and the transmitter. The receiver employs a variable frequency clock, using standard techniques such as a phase-locked loop oscillator, to recover the clock of the previous transmitting station from the received data. The transmitter, in contrast, uses a local fixed-frequency clock. The elasticity buffer in each station compensates for the difference in frequency between the local clock and that of the upstream station by adjusting the bit delay through the station. The elasticity buffer in each station is re-initialised to its centre position during the preamble (PA) that precedes each frame or token. This has the effect of increasing or decreasing the length of the PA as it proceeds around the ring.

3.4.3 Token MAC Functional Operation

In FDDI, media access is provided by a token which is passed from station to station, signifying the right to transmit information frames on the ring. Although a transmitting station must remove its own frames from the ring, pipelining is achieved because a station immediately releases the token after transmitting its frames. MAC schedules and performs all data transfers on the ring.

The basic concept of a ring is that each station repeats the frame that it has received from its upstream neighbour to its downstream neighbour. If the destination address (DA) of the frame matches that MAC's address and there is no error indicated, then the frame is copied into a local buffer with MAC notifying LLC (or SMT) of the frame's arrival. MAC modifies the indicator symbols in the FS field of the frame as it repeats it to indicate the detection of an error in the frame, the recognition of its own address, and the copying of the frame. The frame propagates around the ring to the station that originally placed it on the ring. The MAC of this transmitting station is responsible for removing from the ring all of the frames that it has placed on the ring (a process termed stripping). MAC recognises these frames for stripping by the fact that the Source Address (SA) contained in them is its own address. IDLE symbols are placed on the medium during stripping.
If the MAC has a frame from LLC (or SMT) to transmit, it may do so only after a token has been captured. A token is a special frame that indicates that the medium is available for use. Priority requirements, necessary to assure the proper handling of frames, are implemented in the rules of token capture. Under these rules, if a given station is not allowed to capture the token, then it must repeat it (or in certain cases reissue a token) to the next station in the ring. Only after having captured a token and stripping it from the ring is MAC allowed to transmit a frame or frames. When finished, MAC issues a new token to signify that the medium is available for use by other stations.

The FDDI MAC uses a Timed Token Rotation (TTR) protocol to control access to the medium. Under this protocol, each station measures the time that has elapsed since a token was last received. The initialisation procedures establish the Target Token Rotation Time (TTRT) equal to the lowest value that is bid by any of the stations. Two classes of service are defined. Synchronous service allows use of a token whenever MAC has synchronous frames queued for transmission. Asynchronous service allows use of a token only when the time since a token last was received has not exceeded the established TTRT. Multiple levels of priority for asynchronous frames may be provided within a station by specifying additional time thresholds for token rotation. The use of TTR protocol allows stations to request and establish (via SMT procedures) guaranteed bandwidth and station response time for synchronous frames. It establishes a guaranteed minimum response time for the ring because, in the worst case, the time between the arrival of two successive tokens will never exceed twice the value of TTRT. [12, 87, 88, 114]

3.5 The Orwell Ring Protocol

The Orwell ring was the end result of the search for a new Integrated Service Local Network (ISLN) protocol. The ring consists of a fixed number of slots established at switch-on and maintained continuously. The ring delay is increased if necessary to accommodate an integer number of slots. The size and structure of a slot is describe later in another Chapter. Nodes connected to the ring concentrate traffic from several terminals, supporting a wide range of services. All slots are potentially
available to all nodes: there is no restriction imposed whereby a node is barred from seizing empty slots if it already has a packet travelling round the ring (as is the case in the Cambridge ring protocol [75]). Nor is a node barred from seizing a slot adjacent to one it has just released. Seized slots travel to the destination node where they are released and become available to other downstream nodes.

With the destination release policy, a slot can deliver more than one cell during a single rotation on the ring. The gain from destination deletion is proportional to how far the destination is around the ring. Simulation results show that with substantial two-way traffic the gain will be close to 2.[18] However there is always the possibility of a high usage node demanding a large segment of the ring to send, for example, a high speed file transfer. Thus time variation in the achieved gain can be expected. The major problem that the protocol must overcome is how to prevent nodes being starved of required bandwidth, or incurring excessive delay in frame transmission. The problem is particularly acute for a node downstream from a bursty node seizing all slots for short periods. Without control, transmissions in progress could be pre-empted by a node hogging all available slots.

To overcome hogging the protocol operates in the following way. Nodes are permitted to seize as much bandwidth as they require for a maximum of d-val slots. A counter at each node is used to store the current total number of slots transmitted. However, the counter may be frequently reset during idle ring condition in a way which will become apparent later. In this case a count of d-val will never be reached and the node will continue to seize empty slots in a totally unrestricted way. On reaching d-val, the counter enters a new state known as the PAUSE state. In this state the node is prevented from filling any empty slot. This allows more empty slots to pass downstream to other nodes and, hence, overcome the hogging problem. As more and more nodes enter the PAUSE state or otherwise become IDLE a point is eventually reached where no slots are being filled. Thus a mechanism is needed which can restart activity again.

To achieve a return to full activity every idle node and every node in a PAUSE state conducts a TRIAL using an empty slot to see if it can be passed all the way round the ring without being used. To do this it loads its own address into the
destination area and marks the control field of the slot to indicate that it is performing a TRIAL. However, the control field still indicates that the slot is empty. Hence, if a downstream node remains active, then that node will seize the empty slot and terminate the TRIAL. This procedure carries on until a stage is reached where there are no more nodes waiting to transmit and an empty slot is returned which originated the TRIAL. A successful TRIAL slot is converted to a RESET slot by a suitable change in the control field. This RESET slot then circulates round the ring resetting the d-val counters of every node. The counters are reset regardless of whether they have reached their maximum value or not. Clearly resets will be more frequent on lightly loaded rings.

Call requests are initially directed to the node control. A mechanism is desired whereby the ring has sufficient spare capacity to carry the new request or else reject the call. This control has the desirable effect that potential overload results in call blocking not lost packets spread randomly across all nodes. The interval between two consecutive resets is known as \( \text{reset interval (RI)} \). The reset interval is influenced by the load on the ring: it is short when the load on the ring is light and vice versa. However, when the ring is heavily loaded, the RI is only allowed to increase up to a pre-defined maximum value. In order to maintain the RI below this maximum, each node uses a load monitor to measure the average RI over a period of time, or alternatively, to keep a count of the number of reset occurrences over the same period (reset rate, RR). Based on these measurements, call requests are rejected if they may cause the maximum RI to be exceeded (or the RR to fall below a minimum). There are call acceptance thresholds associated with the different services, and the threshold levels depend on the bandwidth requirements of the services. When the RR falls below any of these thresholds, call requests of the corresponding service will be rejected. With the RI maintained below the maximum value, every node gets its d-val allocation within a specific period and thus its guaranteed bandwidth allocation.[39,40]
3.6 Analysis of HSLAN and MAN Implementations

In this section, implementations of HSLAN and MAN network nodes are analysed and compared in terms of complexity, technology and performance. The analysis focuses on the three media access protocols described briefly in the previous section. In order to present the implementations in a unified manner, the station hardware and architecture used in the various HSLAN and MANs are described in terms of the generic architecture shown in Fig. 3.11. The fundamental blocks of the generic architecture do not exactly correspond to the layers of the OSI Reference Model, because implementation architectures are often chosen according to the implementation technologies available, not according to the abstract OSI Reference Model.[93]

The transmission block provides the baseband point-to-point communication between stations. Typically, this block contains a fibre optic transmitter and

Fig. 3.11 Generic HSLAN/MAN node Architecture.
receiver, which incorporates clock and data recovery circuits. Thus, a recovery bit clock and a serial bit stream are delivered to the Physical Protocol block. This block is capable of performing clock synchronisation, line coding, frame synchronisation, bit manipulation, serial/parallel conversion, and fault isolation. The Media Access block operates on parallel data words. It may consists of access protocol state machines, address look-up tables, frame check sequence generators, timers, counters, and word inspection and word modification logic. Data is buffered for transmission and reception in the Buffering block. This block operates on wider data words than the Media Access block and includes memory components.

The upper layer block contains one or more microprocessors implementing the OSI layers above the MAC sub-layer. Typically, a "node processor" with its own memory executes the LLC and the Network Layer software, while a "host processor" implements the rest of the OSI layers[58]. However, hardware to implement MAC level bridges between networks or interfaces to fast real time traffic sources are also a part of this block. Finally, the management block contains a microprocessor responsible for initialisation, supervision and reconfiguration of the other blocks.

3.6.1 The Physical Protocol Block

The key mechanism of the Physical Protocol block handles clock synchronisation, line encoding/decoding, frame synchronisation, bit manipulation and fault isolation. A special synchronisation problem in HSLANs and MANs is the synchronisation of the high-speed clock oscillators in the Transmission blocks attached. The problem arises because all transmission blocks must either use a common master clock, or each individual Transmission block must use its own local clock. One of two basic methods is employed to synchronise the clocks: synchronous clock synchronisation or plesiochronous clock synchronisation.

Frame synchronisation is executed in the Physical Protocol block to detect the start of a data frame, and to enable a serial-to-parallel conversion with data aligned at parallel word boundaries. Special bit patterns, called delimiters are used to obtain
frame synchronisation. These delimiters are either made unique by use of line code violations, or they are non-unique, i.e., they can occur in the actual data of a data frame. A "flash" detection of one unique delimiter is simple to implement with a shift register, but the hardware complexity increases if two or more unique bit patterns have to be recognised in this way. If non-unique delimiters are used, some kind of state machine is needed making the implementation complex. If bit manipulation is required, this mechanism is more complicated to implement at the bit level than at the word level because the operating speed is higher.

Finally, mechanisms for improving network reliability are a part of the Physical Protocol block. These fault isolation mechanisms handle station and fibre optic cable failures. A faulty station can be by-passed by activating a simple by-pass switch, while more complicated, so called "self-healing", mechanisms are necessary to handle breaks in the fibre. Thus, the self-healing mechanisms require that two-way signalling must be possible in each Transmission block, and that switching from one transmission path to another must be performed in case of fibre or station failure.[9]

3.6.2 The Media Access Block

The state machines of the Media Access block must execute or control the following key mechanisms: start of data transmission, transmission of mixed traffic, termination of data transmission, reception of data, transmission integrity checking, removal of data, and recovery from failures. If a network node has to reserve a free token or an empty slot before transmitting, the Media Access block must ask the Physical Protocol block to manipulate words or bits in the passing data stream.

All the networks incorporate mechanisms to support transmission of traffic with more than one priority. The priority access is allocated in either a static or a dynamic way. Static mechanisms are simple to implement, but they are not fair because only certain selected stations have high priority access. Dynamic mechanisms are fair, but they require priority indications in slots and tokens combined with time-out or round generation mechanisms. Furthermore, the
implementation complexity increases with the number of priorities supported. Regarding the reception of data, protocols where the destination station generates an acknowledgment to the source station are more complicated than protocols without acknowledgment. The size and complexity of address look-up hardware increases with the length of the address fields. Moreover, networks with centrally allocated high priority data channels, like DQDB, require an extra mechanism to handle channel recognition. To ensure transmission integrity, Cyclic Redundancy Check (CRC) sequence generators and comparators are needed. The longer the CRC sequence, the more complex the generators/comparators. Finally, removal of data and recovery procedures are more simple to implement in networks with master stations or end stations than in networks based on distributed control.

3.7 Conclusion

This section give a brief survey and classification of the above mentioned media access protocol for high speed LANs and MANs.

Fibre Distributed Data Interface (FDDI) supports traffic with different bandwidth and transmission delay requirements by use of a time-out mechanism and a priority mechanism. The amount of time a station is allowed to hold a token for transmitting frames is limited by the time-out mechanism. All stations attached to the ring negotiate the length of this token holding time during a distributed ring initialisation process. The priority mechanism offers two classes of services, where the lowest priority class is subdivided into eight priority levels.

Distributed Queue Dual Bus (DQDB) MAN also known as QPSX for Queued Packet and Synchronous Exchange employs a reservation access scheme. DQDB is based on a pair of contra-flowing unidirectional fibre optic buses. The buses originate and terminate at a central master station that performs generation of rounds. Each round is subdivided into a number of fixed length slots and each slot contains data priority information. Therefore, DQDB is able to integrate traffic with two different priorities, where the lowest priority class is subdivided into four priority levels. The high priority slots are allocated by the master station to be
shared between a number of selected stations, while the low priority slots can be used by all stations. The reservation access scheme can only be used to access low priority slots, into which both data and reservations are written by the station.

The Orwell ring operation is controlled by a central master station, which divides the ring into fixed length slots that continuously circulate around the ring. Orwell supports traffic with different bandwidth and transmission delay requirements. The Orwell protocol can integrate traffic with different priorities. The protocol provides overload control on non-priority services in order to protect the synchronous services. It allows dynamic allocation of bandwidth to the VBR queue, with unused bandwidth given to the data queue. Thus CBR services should be unaffected by VBR activity whilst data throughput will be inversely related to VBR activity.
Chapter 4

Orwell Ring ATM based network

4.1 Introduction

The provision of sufficiently high transmission capacity along the entire route between subscribers is a key precondition for a future network. At the same time, however, it must also be possible to subdivide this capacity finely enough so that the bit rate provided exactly corresponds to that required for a particular connection. This has been achieved with an ATM based network implemented using the Orwell protocol.[76,83,115]

This chapter describes the realisation of a 155.52 Mbits/s fibre optic physical layer from the Orwell protocol based on an ATM technology, and also discusses how Orwell can offer broadband transport capabilities. It also discusses the design, development and implementation of a high speed Orwell network that can provide true ATM services with significant advantages over other ATM networks - ease of network and node synchronisation, simplicity of node design, very high node access bandwidth, and dynamic control and allocation of total network bandwidth. A brief description of the Orwell ring is given with emphasis on some parameters which are used in the implementation of the network such as the slot structure, the d-val value and the PHIC.

4.2 The Orwell Protocol

The switching technology discussed in this work uses the Orwell protocol with a non-standard packet size. A full description of Orwell has been published already[34,40]. The Orwell protocol implemented here is based on a slotted ring
concept with an integral number of slots circulating around a communication network. A network will comprise a number of nodes that allow data to be placed on or received from the ring, interconnected by a serial communication medium such as optical fibre cable. A typical Orwell ring network is shown in Fig. 4.1. Each slot contains a header field and a user information field. The header field contains control and routing information. Each slot occupies a total of 176 bits of which 40 bits comprise the header and 128 bits comprise the information field to carry user or network information. The remaining 8 bits are used as start-of-slot identifier for synchronisation purposes.

Data to be placed onto the ring are formatted into data packets of precisely the same length and structure as a slot. These are placed in a queue for access onto the ring. The packets are then substituted onto the ring as and when the node sees a valid empty slot referred to as a TRIAL slot. When the receiving node recognises its own address in the slot header it extracts the data, empties the slot and marks it
as a TRIAL slot with its own address in the destination field. Thus there is
destination deletion of the slot rather than source deletion and consequently a large
potential increase in bandwidth or a reduction in delay for the same load. This can
be contrasted with the more usually encountered source deletion mechanism, which
provides for a piggyback acknowledgement at the expense of maximum potential
throughput.[51]

With the destination deletion policy, a node with a lot of data to transmit can hog
the ring, depriving other nodes on the ring of their own share of the bandwidth. To
limit such occurrence, every node on the network is given an initial value called a
d-val value which corresponds to the number of slots a node can transmit between
resets. Each node contains a counter (d-val) which is decremented for each packet
sent. Eventually the d-val associated with a node will reach zero or the node will
have nothing to transmit. When either of these conditions occurs, transmission of
data from the node ceases, associated transmission hardware is disabled, and the
node enters a PAUSE state. In this state a node will transmit a TRIAL to itself
which may be picked up and used by other nodes for sending data, in which case
the TRIAL is cancelled. If the TRIAL slot is not taken by another node and returns
successfully to its sender it implies that no other node on the ring is demanding
bandwidth. When this occurs, the TRIAL is modified into a RESET slot and travels
around the ring causing the d-val counter on each node to reset. Thus all nodes can
restart transmission and the cycle starts again. This mechanism prevents a node
from monopolising the ring bandwidth, yet allows a highly asymmetric distribution
of bandwidth between nodes to be guaranteed if required for a particular network
traffic profile.

The rate at which RESET slots occur is a measure of ring occupancy and hence the
available bandwidth. This reset rate is continuously monitored by all nodes on the
ring so they each have a measure of ring occupancy. The reset interval is used to
govern the maximum time delay any packet has to wait for access to the ring. It is
also an indicator of how long it takes the ring to clear active nodes (which defines
when the next reset will occur). This mechanism is used to determine whether a
ring has sufficient spare capacity to support a new request, hence an overload will result in call blocking and not lost packets.

The protocol provides overload control on non-priority services in order to protect the synchronous services. Signalling is given the highest priority, followed by Constant Bit Rate (CBR) services, Variable Bit Rate (VBR) services, with data services given the lowest priority. All the service queues are allocated a guaranteed minimum amount of bandwidth but may receive more if network load allows. The Orwell protocol allows dynamic allocation of bandwidth to the VBR queue, with unused bandwidth given to the data queue. Hence, in a situation where all service types are supported, CBR services should be unaffected by VBR activity whilst data throughput will be inversely related to VBR activity.[82]

4.3 The Slot Structure

The actual size and format of the slots is fundamental to the operation and performance of the protocol. If the user data field is small in relation to the overall packet size, much of the ring bandwidth is unavailable due to the large header overhead. If it is made significantly larger it becomes increasingly difficult for delay-sensitive services such as speech to use the ring as access times become larger. To run an efficient ATM network protocol, the slot size and structure must accommodate the needs of all likely services by providing adequate bandwidth combined with an acceptable ring access delay.

As discussed in Chapter 3, the standardized cell size for a Broadband Integrated Services Digital Network (B-ISDN) is a 53-octet fixed length cell consisting of a 48-octet information field to carry the user or network information and a 5-octet header that contains information used to route the cells to their destination. For this work, a non-standard packet size is used. However, the Orwell protocol is capable of conforming to current ATM standard specifically with regard to the cell size. The slot structure used on the Orwell ring is shown in Fig. 4.2. All traffic types share this common slot structure.[81]
4.3.1 Start Field

This is made up of an 8-bit J/K symbol pair. This field enables each node to identify the start of each slot. It is also used by the physical layer hardware and each PHIC for slot synchronisation. The PHIC and the physical layer detects this and locks onto the slot boundary.

4.3.2 Control Field

The half octet field C1 contains the Data (D), Signal (S), Monitor (M) and the Broadcast (B) bits. The different slot types indicated by these bits are shown in Table 4.1 below. The monitor station sets the monitor bit whenever a slot passes through. The monitor bit is cancelled by the destination node as it deletes the slot. If the monitor node subsequently recognises a slot with the monitor bit set, it converts it to a TRIAL (empty) slot, thus removing any erroneous slot from the ring. If the broadcast bit (B bit) is set, every node on the network will copy the contents of the slot into its receive FIFO. The Broadcast bit is cancelled by the node which originated the broadcast.

The next half octet control field contains the parities (P1, P2) field, Synch/Asynch (S/A) field, and the header (H) field. The parity bit P1 covers the control field C1,
the destination address (DA) and the parity bit P1, whilst the parity bit P2 covers
the P2 field, S/A, H bits and the Connection IDentity (CID) field. The remaining
control field bits are used to signify the slot type, as shown in Table 4.1.

<table>
<thead>
<tr>
<th>C1</th>
<th>C2</th>
<th>SLOT TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>D S B M</td>
<td>Hex</td>
<td>S/A H</td>
</tr>
<tr>
<td>0 0 0 -</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>0 0 1 -</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>0 1 - -</td>
<td>4</td>
<td>-</td>
</tr>
<tr>
<td>1 0 - -</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>1 1 - -</td>
<td>C</td>
<td>1</td>
</tr>
<tr>
<td>1 0 - -</td>
<td>8</td>
<td>0 0</td>
</tr>
<tr>
<td>1 0 - -</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>1 1 - -</td>
<td>C</td>
<td>0 1</td>
</tr>
<tr>
<td>1 1 - -</td>
<td>C</td>
<td>0 0</td>
</tr>
<tr>
<td>1 1 - -</td>
<td>C</td>
<td>0 1</td>
</tr>
</tbody>
</table>

4.3.3 Address Fields

There are 32 bits of address information of which 16 bits are used for source
address (CID) and the second 16 bits are used for the Destination Address (DA).
The destination address field enables the data to be routed to the appropriate device
attached to the station. It is possible to sub address nodes within a node. This is
achieved by using the MAP address. The MAP is a name given to two 64 bit
RAMs situated inside the PHIC which is used as a routing table.
The MAP is split up into MAP1 and MAP2. Addresses for MAP1 are taken from
the first six bits of an incoming slot's destination address (DA15-DA10), while the
sub-field of the destination address field (DA9-DA4) is used to address the MAP2
memory. Programming of both MAPs is controlled by the Communication Control
Processor (CCP) via the control bus. When a slot arrives at a node, the destination
address field will be compared with the node's address (NA). The slot will be
received if MAP1 contains a '1' in the location accessed by the first six bits of the
DA, or the first six bits of DA matches the first six bits of NA and MAP2 contains
a '1' in the location accessed by the second six bits of the DA. The node will also
received a slot if the node's address matches the destination address. The address
field should always be represented as data symbols.

4.3.4 Information Field

The information or user data occupies the remaining 128 bits of the slot. The slot
structure used in the PHIC is summarised in Fig. 4.2.

4.4 Node Design

Each node on the Orwell ring contains a Protocol Handling Integrated Circuit
(PHIC), a Communications Control Processor (CCP) and independent physical
layer Transmit (Tx) and Receive (Rx) boards, as shown in Fig. 4.3. The function of
the PHIC device[81] is to control the initialisation of either a single Orwell ring or
a multiple ring Orwell Torus, then to control the communications between nodes.
The PHIC implements the full Orwell protocol and is responsible for handling all
tasks associated with switching data in and out of the node. The physical layer Rx
and Tx boards provide the interface between the optical fibre layer and the PHIC.
The CCP communicates with the PHIC via an 8 bit control bus.
4.4.1 PHIC

The Protocol Handler Integrated Chip (PHIC) implements the basic Orwell protocol, enabling reception and transmission of asynchronous and synchronous data, constant and variable bit rate services, on a slotted ring. It generates the network slot structure, synchronises the network and provides a range of error detection facilities. The PHIC is capable of operating either as a single ring or multiple rings Torus. Torus operation can be found in references [106,107].

The PHIC has a 17-bit wide bidirectional bus for receiving and transmitting information to the attached communication equipment. Cells or slots to be transmitted and received are passed to and from the PHIC over the 17-bit data bus as eleven 16-bit words (i.e one complete slot). The 17th bit is used as a start-of-slot
indicator; it is set to one (‘1’) for the first 16 bit word and zero (‘0’) for the subsequent 10 words. Thus the user must present a complete slot to the PHIC for transmission over the network. The PHIC will perform parity checks on the header field and will modify the parity fields accordingly before placing the slot onto the network.

The complete data path within the PHIC spans 3 blocks: FIFO, DATIN and DATOUT. The PHIC introduces a total delay of 36 bits to the ring data due to nine internal 5-bit registers. These registers are labelled as A to H inclusive plus an additional output register, as shown in Fig. 4.4. These are clocked by the symbol clock (SCK). Logic surrounding the A register detects the J/K symbol sequence for the start of slot and then identifies the slot type from the first header control field C1.

When a slot from the network enters the PHIC the C1 field of the header is checked to determine the cell type. The incoming cell is converted into 16-bit parallel words by taking the output of register C to F on every fourth symbol clock cycle. The first word will contain the J/K symbol pair, the C1 header field and the first four bits of the destination address. A 4-bit magnitude comparator is used to compare the destination address (DA) in the received slot with the node’s own

![Fig. 4.4 PHIC internal register used for loading and receiving ring data.](image-url)
address. If the incoming slot is destined for that node then the whole cell is passed out of the chip via the RT bus.

A TRIAL cell entering the PHIC can only be used to transmit information if a cell is ready for transmission and the node d-val is greater than zero. When a queued cell is ready for transmission, a transmit request is sent to the PHIC. An empty slot entering the PHIC is identified from the C1 header field and causes the RT bus outputs to be tri-stated if such a transmit request has been received. The slot is clocked into the PHIC as eleven 16-bit words and loaded into registers E, F, G and H overwriting the incoming slot. A fifth non-data bit is added to each of these registers and the slot is passed onto the ring.

The PHIC has seven received slot control pins which consist of an ERROR pin, RTP(0), and six type pins RTP(1:6). If ERROR equals zero then the type pins duplicate the S, B and A/S bits and indicate whether the slot was received because of a node address match (NAQD = 1) or a routing table match (MAPI = 1). If however, ERROR is one, the first five pins indicate the error type as summarised below;

<table>
<thead>
<tr>
<th>ERROR</th>
<th>RTP(1)</th>
<th>RTP(2)</th>
<th>RTP(3)</th>
<th>RTP(4)</th>
<th>RTP(5)</th>
<th>TP(6)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>S</td>
<td>B</td>
<td>A/S</td>
<td>MAP1</td>
<td>MAP2</td>
<td>NAQD</td>
</tr>
<tr>
<td>1</td>
<td>P1</td>
<td>P2</td>
<td>NOD</td>
<td>MPE</td>
<td>MOF</td>
<td>0</td>
</tr>
</tbody>
</table>

P1 and P2 are parity error indicators, while NOD is the no data bit which indicates whether the information in the data field is either control or user data. MPE and MOF stand for monitor pass error and monitor failure respectively.

4.4.2 ELECTION

Every node on the network is capable of being a monitor station. The selection of a monitor for Orwell is done by employing a qualification test. The node itself takes
an autonomous decision on its eligibility to compete in the election process, by looking at a reference 2MHz clock input and grading itself according to the following rules:-

a) When a 2MHz clock is detected and REF = 1 in status register 1, then that node can compete.

b) When a 2MHz clock is present and REF = 0, the node can compete but will later be eliminated by any other competing node which has REF = 1.

c) When there is no 2MHz clock, the node cannot compete.

The REFerence (REF) flag (1 if there is PSTN clock source) is set by a Communications Control Processor (CCP) if the 2MHz clock is locked to the national network. In a small local network, interface to the national network may not be required. The final selection is based on each unique node address, and by convention the winner will have the smallest numerical address in the network.

To implement this procedure, the initiating node opens the ring, creates its own slot structure and makes its address known by transmitting a special election slot (S-slot), with its own unique node address in the destination address (DA) field. The node containing the master network clock starts by continuously sending out S-slots. The node immediately 'downstream' receives these S-slots and examines the destination address. If it is able to be a monitor and possesses a lower address, it overwrites the destination address with its own address, opens the ring data path and transmits its own S-slot over the network to the next node. These S-slots may in turn be overwritten by another lower-address node further downstream. Any station receiving S-slots with an address lower than its own will simply pass them on. Eventually the station with the lowest ring address will receive its own S-slots, indicating that it has won the election.
4.4.3 MONITOR

The newly elected monitor now enters a SYNCH state in which it resynchronises the ring with its own reference clock signal and establishes a new slot structure. During this state, all other nodes cease communications, lock themselves onto the monitor's signals and passively repeat them. For Torus operation, the phase of the slot structure on the rings of a Torus can be set to avoid reset clashes and to smooth out the bandwidth demands of the attached terminal equipment. The monitor node monitors the ring and takes corrective action necessary to maintain the operation of the ring.

After the SYNCH state, the monitor enters a start sequence within which it measures the ring size, and adds shift register bits to pad out the ring delay length to an integral number of whole slots. Closing the ring data path at its node completes the ring closure and the monitor begins counting S slots in a delay counter whilst converting returning S-slots into TRIAL slots. When the first TRIAL returns, the counter now contains the number of slots in the ring which will be used later to time outstanding resets.

When in full operation there is always an integral number of slots circulating on the ring irrespective of network size. This is achieved by the inclusion of a variable length First-In-First-Out (FIFO) buffer in the data stream, contained in the elected monitor station, as shown in Fig. 4.5. The padding scheme begins by initialising the FIFO, enabling it to operate in dynamic equilibrium, with the FILL rate equalling the UNLOAD rate, whilst maintaining a sufficient residual data storage of 6 nibbles to allow for jitter. An input start delimiter (J/K) will quickly fall to the bottom of the FIFO and when detected, the unload clock is stopped, causing the FIFO to fill up with the remainder of the slot.

The node meanwhile, has been transmitting a continuous stream of T-slots (a special slot used in the padding scheme). If the UNLOAD clock is enabled, coincident with the start of a new slot transmission, the FIFO will contain the required padding necessary to pad out the ring to an integer number of whole slots.
If a node loses the election, the FIFO is disabled, causing a disturbance in slot length but only as far as the next downstream node still bidding for monitor status. During the election stage and subsequent FIFO adjustment stage the ring is essentially operating in an open-loop mode, being discontinuous at the monitor node. After the successful election stage and FIFO padding the ring data path is closed in the monitor station and communication may then commence between all nodes on the network (referred to as comms mode).

The monitor node is the first node to transmit a TRIAL slot. This is to ensure that already synchronised nodes or stations do not attempt to transmit until all stations are synchronised. On entering the communications mode the monitor continuously examines the integrity of the ring. Error conditions occurring are logged and corrective action is automatically invoked. The monitor is responsible for maintaining the ring slot structure during normal communications mode. It also performs performance monitoring, error detection and a status reporting role. The monitor node is also responsible for deleting full or data slots which attempt to circulate around the network more than once.

Fig. 4.5 Variable length FIFO buffer internal to each PHIC device.
4.4.4 Communications Control Processor (CCP) Interface Logic

The Communications Control Processor (CCP) Interface Logic contains all the logic and timing circuits necessary for the PHIC to communicate with a CCP (processor) running at 10MHz. The operation of each PHIC is controlled via a control interface. The CCP interface to the PHIC constitutes an 8-bit control bus, 8-bit data bus and 5-bit address bus. For the network described here a CCP operating at 6MHz communicates with each PHIC over the control bus (see Fig. 4.6), however this would be performed as a system management task were the system to incorporate any level of network management. Apart from the monitor node which uses the IEEE488 interface, all other nodes on the network have an 'on-board' CCP. The CCP is programmed with operational information such as node address and d-val as well as register values controlling node functionality.

Fig. 4.6 Communication Control Processor.

The PHIC uses the delayed negative edge of lower data strobe (LDS) and upper data strobe (UDS) to strobe data off the data bus and issues an acknowledgement to indicate acceptance of data into the PHIC's lower byte and upper byte respectively. LDS and UDS are output signals from the PHIC. The flow diagram of the programme shown in Fig. 4.7 starts by initialising the registers and ports of the CCP. The registers of the CCP are used in a display routine so that error
monitoring can be observed. The next routine writes information necessary for the operation of the PHIC. The address of the node is then programmed into the appropriate PHIC register. This is followed by a routine to write the d-val into the appropriate PHIC register. The program for the IEEE488 interface to the PC is included in Appendix I.

The routing table addresses are programmed using the sub-address field of the destination address field. The MAP ENable (MAEN) bit in the PHIC is set to '0' before programming the MAP entries. The table in Appendix III shows the correlation between DA(15 - 4) and the CCP address and data. Programming multiple entries into the same row address requires the data parts of the table to be logically ORed. CCP address bit 4 indicates a MAP address, with MAP2 addressed by the Lower Data Strobe (LDS) and MAP1 addressed by the Upper Data Strobe (UDS). The lower three bits of the address bus provide the row address for CCP byte wide Read/Write access to the selected MAP. The CCP sets the MAEN bit when programming is completed, which enables ring accesses to commence. A display routine is included in the program to display the content of each PHIC register, hence indicating states of the PHIC. DaTa ACKnowledgement (DTACK) issued by the PHIC signifies to the CCP the presence of valid data on the bus. After
the CCP negates the addresses and data strobes, the PHIC returns first D(7:0) to the high impedance state, then DTACK is released.

The flags in the lower byte of register 0 of the PHIC are set by the state machine whenever a system error is detected by the node. Any flag set in this register will force an interrupt, alerting the communication control processor to the interrupt condition. Whenever a parity or slot related error is detected by a node, a flag in the high byte of Register 0 is set. An interrupt condition is not raised since these flags do not necessarily indicate a system error. Register 0 thus contains error information for the benefit of the CCP and therefore should only be cleared by the CCP.

The PHIC chip control interface has been designed to be compatible with the 68000 processor family. The obvious control device would therefore seem to be a 68000 development system. However, the PCB complexity and cost of a 68000 system is significant. The development board acquired from British Telecom Uses an IEEE488 interface to communicate with the PHIC. It was therefore easier to incorporate this into our design. Furthermore, the amount of logic required to convert from the IEEE488 to the PHIC control bus is modest. The PC also provide a user-friendly environment for writing and modifying the software programme. In the experimental system developed, only the monitor node uses the IEEE488 interface to communicate with the PHIC.

An additional 64x16 bit FIFO known as the CAMERA was added to the monitor node to take 'snap shots' of the passing slots. This allows us to examine the slots for correct structure especially during ELECTION, SYNCH and START mode of operation. The CAMERA copies five consecutive slots and displays them in the structure shown in Fig. 4.8. The Device Clear (DC) function is used to clear the CAMERA FIFOs. Group Execute Trigger (GET) is used to trigger the CAMERA for a snap shot of ring activity. Without affecting network operation it effectively copies and displays groups of slots travelling around the ring and allows detailed examination of the control and data content of each slot, providing a powerful debugging tool.
4.5 Physical Layer Tx and Rx Boards

Packetised data enters or leaves the PHIC as a series of 16-bit words, but this data is passed from or to the physical layer as 5-bit words. In transmission, the PHIC converts the 16-bit data words to 4-bit words and then adds a fifth bit to uniquely code each as control or data information before being directed to the transmitter side of the physical layer (Fig. 4.9) and subsequently over the optical fibre cable. In the coded symbol, bit_5 is a non-data flag which is set to ZERO when bits 4 to 1 represent binary data. When the non-data flag is set to ONE, bits 4 to 1 represent control and slot framing codes as shown in Appendix III. In reception the reverse process takes place where 5-bit words are received from the physical layer receiver and are converted to 16-bit words and directed to the 17-bit data interface. If the physical layer data packet is not intended for the node, ie the destination address does not correspond with the node address, then the data is effectively passed through from receiver section to transmitter section.

The physical layer provides the digital baseband point-to-point communication between nodes at a physical layer speed of 155.52MHz. Typically it contains a fibre optic transmitter and receiver. The physical layer performs line coding/decoding, parallel to serial/serial to parallel conversion, clock and data...
Packetised data

11 x 16-bit words

16-bit
4-bit
5-bit encoded

Fig. 4.9 Schematic of internal operation of the PHIC during data transmission

To physical layer Tx

5-bit

recovery circuit, and clock and frame synchronisation. A schematic diagram of the Tx and Rx stages is shown in Fig. 4.10. Both stages employ Emitter Coupled Logic (ECL) technology to cope with the very high switching speeds in the serial data path. One important feature of the network is that with the exception of a master node, each node extracts clock and data signals directly from the serial data stream and this has important consequences both for the physical layer design and the method of ring synchronisation to be discussed in a later section.

4.5.1 Coding/Decoding

Line coding mechanisms introduce redundancy into the serial data stream on the media. This redundancy is required for reliable clock recovery and for the coding of special control symbols. Regarding clock recovery, long periods of consecutive identical bits are not desirable because timing information for regenerating an incoming signal cannot be maintained correctly during a long sequence of identical bits. Therefore, the line codes are designed to generate frequent transitions, ensuring that the maximum number of consecutive bits is small. This also solve the problem of AC saturation associated with long sequences of 1’s. Special control symbols are used to obtain frame (slot) synchronisation. The line codes can be
The function of a scrambler is to generate transitions in the line encoded data stream when the original data stream contains long sequences of identical bits. Furthermore, strings of repetitive-pattern data are broken. A scrambler is simple to implement with a shift register and a few exclusive OR gates. Scramblers are often combined with other types of line codes because they introduce bit error propagation, and bit error detection cannot be performed by the scrambler alone.
Examples of bit insertion codes are the 5B6B-PMSI, 8B1C, and 9B1C line codes. In the 5B6B-PMSI (Periodic Mark Space Insertion) scheme, the encoder alternately inserts a mark (one) and a space (zero) for every five information bits. In the 8B1C (8 Binary with 1 Complement insertion) scheme, every eight bits are preceded by one extra bit, which is the complement of the first of the eight bits. Such bit insertion codes are simple to implement, guarantee that a maximum number of consecutive identical bits is not exceeded, and do not introduce propagation of bit errors.

Finally, the 4B5B code which is employed in this project is an example of a block code. The 4B5B code translates each source 4-bit block into a constrained 5-bit sequence, which minimizes the difference between the numbers of marks and spaces in the data stream. As a result, the maximum number of consecutive identical bits can be kept low. Special non-data characters are defined as zero, and transmission bit-errors can be detected. However, drawbacks are that bit error propagation is possible, and that sophisticated electronic circuits are required to perform the 4B5B code conversion. The 4B5B line code is combined with a simple NRZI (Non-Return-to-Zero-Invert-on-ones) code. This latter operation is not strictly required, but the sponsor wanted the coding to be compatible with FDDI technology. This allows the use of 5-bit code blocks containing many consecutive marks (or ones), since the NRZI encoder generates a transition for every mark. The same coding technique is used in FDDI technology.

4.5.2 Parallel to Serial and Serial to Parallel Converters

Data is transmitted serially over the fibre, but enters the PHIC in 5-bit parallel form. Therefore a major task of each Rx and Tx section is to perform an appropriate parallel/serial data conversion at the basic ring rate of 155.52MHz. At the transmitter, the parallel data after being converted to ECL signals, is serialised and clocked through to the optical transmitter. While at the receiver, the serial data is converted back into parallel 5-bit form which is then directed to the PHIC in a 5-bit data stream.
4.5.3 Clock and Slot Synchronisation

One of two basic methods is employed to synchronise the clocks: synchronous clock synchronisation or plesiochronous clock synchronisation. When a synchronous clock synchronisation technique is applied, all Transmitter sections receive and transmit data with a common master station clock. When a plesiochronous synchronisation technique is used, each individual Transmission section receives data with a clock locked to the clock of the preceding Transmission section, and transmits data with its own local clock. Due to the clock oscillator tolerance, this method implies that the number of transmitted bits can differ from the number of received bits. Therefore, the Physical Protocol block must include an elastic buffer to accumulate these bits. Furthermore, the data stream must contain a special idle bit field, where the elastic buffer is allowed to be recentered, which has the effect of adding or deleting idle bits. So from an implementation standpoint, plesiochronous clock synchronisation is more complex to implement than synchronous clock synchronisation.

The synchronous clock synchronisation is the method employed in this research project where one node on the network contains a clock oscillator running at 155.5MHz and the rest of the nodes extract their clock from the data stream. Ring synchronisation involves two stages. Initially all nodes must be synchronised to the serial data stream and also to each slot boundary. When all nodes on the network are synchronised, all the nodes on the network operate at the same frequency but with a phase shift. The monitor station then synchronises the operation of the entire ring.

Synchronisation of each network node is achieved during the election stage. One node on the network contains a master clock generator. In practice, this would be locked to a national clock source. At start-up all other nodes are free-running. The master node transmits a synchronisation slot (S-slot) onto the ring. The node immediately downstream extracts clock and data signals and on detecting the start-of-slot delimiter locks on to the slot boundary. The node regenerates the slot which is then forwarded downstream to subsequent nodes which also extract their clock and data signals and similarly lock onto the slot boundary. Finally the slot
returns to the master node. At this point all the nodes on the network will be operating at precisely the same frequency, but each will exhibit a phase difference. Since all nodes on the network will be operating at the same master clock frequency, all clock signals within each node that are derived from this base frequency will also be frequency-locked.

Word synchronisation is executed in the Physical layer to detect the start of a data slot, and to enable a serial-to-parallel conversion with data aligned at parallel word boundaries. Special bit patterns, called delimiters, are used to obtain slot synchronisation. These delimiters are either made unique by use of line code violations, or they are non-unique, i.e., they can occur in the actual data of a data frame. A "flash" detection of one unique delimiter is simple to implement with a shift register, but the hardware complexity increases if two or more unique bit patterns have to be recognised in this way. If non-unique delimiters are used, some kind of state machine is needed making the implementation complex. If bit manipulation is required, this mechanism is more complicated to implement at the bit level than at the word level because the operating speed is higher.[44,45]

Since data enters each PHIC device as a parallel data word, it is necessary to synchronise each receiver not only at the bit level (i.e., frequency synchronisation) but also at the 5-bit word boundary (i.e., word synchronisation). This is accomplished by using high speed hardware to detect the start-of-slot delimiter (known as the J/K symbol pair). The serial to parallel conversion section within each receiver constantly examines the incoming serial data stream, checking for the occurrence of the J/K symbols. When these are detected the receiver node initialises its clocking circuitry at the appropriate time, ensuring that the PHIC device receives valid data words.

**4.5.4 Clock and Data Recovery Circuit**

The Clock and Data Recovery chip used in this project is a complete Phase Locked Loop (PLL) clock recovery and data returning/regenerating subsystem for applications requiring high data rates, and small size. The chip integrates PLL clock
and data recovery together with a high-performance loop filter and other components to realise a complete 3-terminal (data in, clock and data out) PLL clock recovery subsystem. No external components are required. Internally, it is made up of a comparator whose decision threshold value is set internally but can be adjusted externally since the value of the resistors are chosen such that the input resistance is less than one kilo ohms. It contains a phase frequency comparator, data regenerator, Voltage Control Oscillator (VCO) and a Loop Filter. The operating centre frequency is factory set to 155.52MHz. The PLL design tracks input data frequency drift of +/- 1MHz and retains lock for long constant data run lengths.

Unlike SAW filter clock recovery circuits which filter the clock signal from incoming data, the PLL clock and data recovery device used in this project is capable of synchronising an internal VCO directly to an incoming digital data stream, while simultaneously retiming and regenerating the data stream. The on-chip VCO will generate a clock output even in the absence of incoming data. This might be critical if the recovered clock drives a state machine or any other logic circuitry that might go into a metastable state in the absence of a clock. It is also capable of unaided frequency acquisition eliminating the need for special circuits to "pull" the loop into lock when the incoming data rate differs from the initial VCO (clock) rate.

4.5.5 Optical Fibre

Currently available silica glass fibre optic waveguides offer dramatic cost and performance advantages over electrical waveguides. In long-haul communication networks optical fibre has long since supplanted coaxial cable as the medium of choice for high bandwidth, long distance data links. It is also widely used for high-speed networks.[10,12]

The optical devices used in this project are the DLT1000 Data Link Transmitter and the DLR1000 Data Link Receiver. The DLT1000 Data Link Transmitter converts serial ECL signals to lightwaves in the 1300 nanometer band. It is capable of data rates from DC to 220 Mbits/s. Transmit Disable signals are received
through complementary differential inputs to comparators. Logical 1 to transmit data results in high optical output. Logical 1 to Transmit_Disable disables the transmitter. The DLR1000 Data Link Receiver converts lightwaves in the 1300 nanometer band to serial ECL signals. Data rates from 1 to 170 Mbits/s can be used. Data and Signal_Detect signals come from complementary, ECL-compatible differential outputs. In the logic 1 state, the positive output is high and the negative output low; in the logic 0 state, the positive output is low and the negative output high. The optical device can only operate well if the ratio of ‘1’ to ‘0’ or ‘0’ to ‘1’ does not exceed 60 to 40 percent for long runs of constant symbol.

4.6 Summary of the Physical Layer

4.6.1 Tx Section of the Physical Layer

Referring to Fig. 4.10(a), in a physical layer transmitter, parallel data is latched from the PHIC and this is encoded using 4B/5B coding to ensure that a roughly 60/40 distribution of '1s' and '0s' is obtained in the data stream. The parallel data is then serialised and is encoded a second time using a Non-Return to Zero (NRZ) coding technique which has the effect of alternating any long sequences of '1s' in the data stream. The serial data is then transmitted optically over the fibre using a commercially available optical transmitter device. The coding techniques are necessary for the synchronisation of each physical layer receiver, and are also required for the operation of all optical devices which would tend to saturate if they did not receive a roughly equal distribution of '1s' and '0s' in the data stream. This latter consideration can be avoided, but only by the use of more expensive optical devices.

4.6.2 Rx Section of the Physical Layer

At the receiver, Fig. 4.10(b), data is received by an optical receiver device and the signal from this is immediately fed to a data and clock extraction module. This
module employs internal phase locked loop circuitry to lock on to the 155.52MHz fundamental frequency of the incoming serial data stream and generates a clock at precisely this frequency (which corresponds to the master network clock frequency). Using this clock the device also extracts a stable data signal from the data stream. The device can only operate if the incoming data stream has a sufficiently varying bit pattern - a long series of '1s' or '0s' will cause the device to lose synchronisation, hence the need at the transmitter for coding which introduces a degree of alternation in the data. The receiver then performs the reverse of the process performed in the transmitter. Data from the extractor module is NRZI decoded, converted to 5 bit parallel form and then 4B/5B decoded to generate a 5-bit data stream to the PHIC. This data will be identical to that transmitted by the PHIC in the node immediately 'upstream', implying that the physical layer Rx and Tx boards and the fibre optic cable form an effectively transparent communication link. The effective error rate of this part of the circuit is negligible.
Chapter 5

Implementing Real-Time Video Transfer on a Fibre-based ATM Network

5.1 Introduction

After the development of the high-speed network discussed in Chapter 4, an application was developed to demonstrate the capability of the network to carry a high-bandwidth service. It would have been possible to add a series of different applications to show both the multimedia aspect of the network and the potential to allocate high bandwidth to a particular service or application. Cost and the availability of equipment was a limitation. Instead an application was selected that would offer a high-bandwidth service with the understanding that many applications with a similar collective bandwidth would behave in the same way. Hence a constant bit-rate video service was used with an output bit-rate of 77.8 Mbits/s.

This chapter describes a functional system that allows uncompressed colour video images at 25 frames of 625 lines per second to be transmitted at full television resolution and frame rate over a fibre-based 155.52MHz ATM network supporting the Orwell 'A' protocol. Emphasis here is placed upon a discussion of the operation of the video system at the hardware level. The system has been developed primarily to demonstrate that an ATM network running the Orwell protocol can offer a very high bandwidth guaranteed service which could be used for example to support multiple highly-compressed video streams. No attempt is made to define any level of network management.

Conventional systems that allow video signals to be distributed over a digital network must invariably be adapted to an environment of insufficient point-to-point bandwidth and variability of this bandwidth due to network loading. Many methods exist whereby the video stream bandwidth is reduced sufficiently to allow the transfer of a moving video image, but in all cases the quality of the received image
is compromised[82,103,104]. When a communication network offers the video user
a sufficiently large guaranteed bandwidth, there exists the potential to distribute
very high quality image sequences without the need for the compression techniques
or framestores typically associated with networked video. When real-time video
transfer is attempted, many of the problems encountered in conventional slow-scan
video systems are removed, but other problems arise which are uniquely associated
with this type of video transfer.

It is important to realise that the system has been developed to demonstrate the
ability of an ATM network to carry very large bandwidth calls in real-time. It could
in practice be employed to carry high-quality services such as video communication
but in reality dedicating such large bandwidth to just one video channel is
extremely wasteful and may not be commercially viable. It would be much better
employed to carry a large number of lower-rate streams, for example compressed
videophone communications or compressed HDTV. Either traffic flows could easily
be supported by the Orwell protocol.

A diagram of the complete networked video system is shown in Fig. 5.1. Two
nodes on the Orwell ring are used for video transfer, one for transmission and one
for reception of video data. Each node has a ring access sub-system, and an
associated application sub-system. The nodes are shown as separate Tx and Rx
stations, but each can perform both operations, allowing two-way communication.
The ring access sub-system gives the user an access path to and from the ring and
is not specific to an application. The application (video) sub-system is designed
specifically to allow the transmission of real-time video over the network. The
operation of each sub-system is described below. To achieve such a high-speed
application, all of the components in the video data path were developed in
dedicated hardware. Although this provides little flexibility, it was considered the
easiest route to demonstrate the principle of the network to support a very high
bandwidth service. Other solutions were considered giving considerable greater
flexibility, but would have been more complex and would have generated little
advantages in this application.[84]
5.2 Transmitter Station

5.2.1 Video Tx Sub-System

The video sub-system in the transmitter station accepts a standard colour PAL video signal as an input. For testing purposes and to generate a very high quality images a laser disc player has been used as the source signal but any PAL video source is suitable. The video signal is passed to a Transmitter (Tx) CODEC which digitises it to generate a serial PCM data stream at a bit rate of 77.8MHz. Since the CODEC contains only very limited storage, the PCM output represents the real-time video image encoded at 25 frames of 625 lines per second. The CODEC[29] assigns special codes to horizontal and vertical sync periods for use in the receiver codec. The sample rate at which the CODEC operates is set to be half the master network frequency of 155.52MHz and derives this clock from the network. The PCM video signal generated by the Tx CODEC is then converted to parallel data and is directed to the ring access sub-system for transmission over the network.
5.2.1.1 Transmitter CODEC

With the development of high-speed Local Area Networks (LANs) and transmission systems, a need for cheap quality television CODECs are essential. Some multiplex applications and the non-standardisation of LAN line rates have necessitated a flexible CODEC to operate over a wide range of transmission rates. This section discusses the CODEC used in producing a Constant Bit Rate video service that is transmitted over the Orwell ring. The CODEC described here was designed for use with a high-speed dedicated link. In this instance the CODECs are required as demonstrators of the communication system. They offer "domestic" quality video, audio and a data port. They have an inbuilt flexibility to work over a wide range of transmission bit-rates between 63 Mbits/s and 90 Mbits/s. Although this is not the author's work, it is briefly discussed here for completeness.

The CODEC performs straightforward 6-bit Pulse Code Modulation (PCM) linear sampling of the composite PAL input signal, resulting in a sample rate of 11.65 Megasample per second (Mfps). The whole of the colour burst signal is included in the video sampling range. The transmitter is arranged to be driven by an external clock source. The clock in this case is 77.8Mhz which is exactly half the network frequency. An alarm indicator is raised when loss of this external clock occurs. An internal crystal oscillator can be switched-in for testing purposes. Whichever clock source is driving the transmitter, it is also fed to external equipment through line drivers and can be used to drive a stand alone receiver unit.

The CODEC was designed to accept video input signals of one volt peak-to-peak (1v p/p) with an input impedance of 75 ohms. The adaptive sync separator has a slicing level at 50% of sync amplitude. The clamping voltage is arranged such that the negative half of the colour burst signal falls within the input range of the Analogue to Digital Converter (ADC). The analogue to digital conversion is carried out by a flash converter, offering 6-bit resolution up to 30Msps. The digital to analogue conversion is performed by a DAC capable of working up to 20Msps.

Good clean sync separation from a signal is essential for clamping the video signal and the overall transmission frame synchronisation requirements. The transmission
framing structure is designed to be synchronous to the incoming video line sync pulses. Each frame sync interval contains 55 6-bit words. The first two words of each frame sync interval are arranged to contain a frame alignment word which is a 7-bit Barker sequence for subsequent video line sync alignment at the receiver, and a 5-bit word for video frame alignment at the receiver, as shown in Fig. 5.2.

\[
F = 1 \text{ at start of video line number 4, otherwise } F = 0
\]

Fig. 5.2 Alignment words at start of transmission frames.

The frame alignment word is the reference point from which all other multiplex signals are referred. This, together with the video frame alignment word, forms the beginning of each transmission frame. Due to the windowing technique employed for synchronisation at the receiver, no other data are permitted to emulate the frame alignment word within approximately \(+0.7\ \mu\text{sec}\) of the frame alignment word. To ensure this the 14 words following the alignment pair are transmitted as the video frame alignment word and after a total of 731 words have been sent to line, all further data is sent as video black level 001011 until the new frame alignment word has been sent. Thus 14 or 15 video black level words precede each frame alignment word. In the transmitter, a timing PROM has a switchable input, allowing each audio, data and video transmitted word to be set to video black level or ones according to the preprogrammed sequence.[29]

The twice video line frequency of 31.25KHz provides the sampling rate of the audio signal. This together with 12-bit linear PCM format provides acceptable quality audio transmission. Anti-aliasing filters are included at the audio input and output stages. These are 7th order lowpass active elliptical filters, adjusted to give a
stopband edge at 15.625KHz, with stopband attenuation of 60 db, and ripple in the passband less than 0.5 db p/p. Since the successive approximation conversion time occupies an appreciable portion of the audio sampling period, it was necessary to include the sample and hold circuit before the ADC.

Approximately 3.3 Mbps spare capacity still remains during the video line sync interval after the transmission frame synchronisation and audio information have been included. Part of this spare capacity is used to transmit data at 281.25Kbps. Standard TTL interfaces are employed, and control signals are provided to facilitate latching of the 18-bit words at video line rate.

5.2.1.2 Serial to Parallel Converter

The serial to parallel conversion is performed by a card utilising a custom designed chip that employs both ECL and TTL technology, ECL technology being required to accommodate the high frequency serial bit stream. Serial data and clock at 77.8MHz are input to the serial to parallel card, and this card performs a serial to parallel process feeding parallel output data to the input stage of the packetiser. A write strobe is generated by the converter which shifts the information into the FIFOs in the packetiser. (Fig. 5.3). The data rate from the serial to parallel

Fig. 5.3 Serial to Parallel Converter.
converter is 16 bits every 200ns. Hence the write signal is a 5MHz clock with a 200ns clock period. This will satisfy the setup and hold times for the FIFOs. No acknowledgement is generated for the data transfer to the packetiser.

5.2.2 Access Sub-System

Data must be presented to the PHIC in the form of data packets of precisely the same length and structure as a slot, containing a start-of-slot delimiter, control and address information, and a user data field. The access sub-system provides the interface between the protocol handling device (PHIC) which controls the node operation, and the user application hardware (Fig. 5.1). Two main tasks are performed by the access circuitry: packetisation and queueing (in the Tx node) and queueing and de-packetisation (in the Rx node).

5.2.2.1 Packetiser

Data and address information generated at the video transmitter must be in a specific format before it may be distributed over the network and the process which formats user data, destination address and control information is termed packetisation. On an Orwell network this process occurs outside the PHIC. This process is performed purely in hardware by a board known as a packetiser. The video packetiser takes in 16-bit digitised video words from the serial to parallel card and combines these with control information and a destination address to form network slots (the address field is hard-wired since there is no network management). An element of programmability could however be added. This task is performed synchronously with the Tx CODEC.

A packetiser is necessary for every transmission node on the network and a corresponding de-packetiser is required for every receiver node (these may co-exist within the same node). The removal of the packetisation function from the tasks performed by the PHIC gives the user a greater bandwidth data channel onto and from the network than would otherwise be possible.
Data from the video subsystem is passed to the hardware packetiser as a continuous stream of 16-bits words. When there are eight words stored in the packetiser’s FIFO, a header which constitutes the source and destination addresses and control bits is appended to form a complete slot. This is then passed to the queue card. The slot is of the same size and structure required by the PHIC. Fig. 5.4 summarises the function of a packetiser. A control mechanism in the packetiser board monitors the number of words in the packetiser FIFO, and a counter is incremented for every word written into the FIFO. If there are more than eight words in the FIFO, the packetiser adds the header to the eight words and passes the complete slot to the transmitter queue card.

![Fig. 5.4 Function of a Packetiser.](image)

5.2.2.2 Transmitter Queue

The queue card provides an interface between the packetiser and the PHIC card, taking complete packets of data (equivalent to slots) from the packetiser and storing them before the PHIC accepts them for transmission over the network. This ensures that complete data slots are available before the PHIC attempts transmission. The queue card must be capable of operating at least as fast as the packetiser. Practical multimedia systems require a range of bandwidths and connection
characteristics to be available, and the Orwell system has been developed to cope with this by providing multiple connections per station, geared to the requirement of each connection. Fig. 5.5 shows the outline architecture of the multi-user interface with each queue capable of supporting multiple packetisers carrying different services.

![Fig. 5.5 Multi-user Orwell Interface.](image)

The queue was also implemented to cope with the jitter variation caused by the activities of the network. Thus the queue acts as an elastic buffer between the synchronous operation of the packetiser and asynchronous nature of the network. See Fig. 5.6. The handshaking mechanism between the packetiser and the queue guarantees that only complete cells are stored in the queue. Incorporated in the queue is a counter that counts the number of slots in the queue FIFO. The counter is incremented for every complete slot written into the FIFO and decremented for every cell read from the FIFO. When the counter reaches a certain number, in this case 15, the queue stops accepting data from the packetiser. This is done to maintain slot boundaries.
The packetiser provides a cell-ready strobe to the queue when a cell is ready to be passed to the queue. The queue checks the counter and returns a ready signal which initiates the transfer of a cell to the queue. A shift-in signal is generated by the packetiser which writes the slot into the queue. A cell-sent strobe is passed to the queue when the transfer of a slot is completed.

The transmit queue will generate a transmit-request strobe when a cell is ready to be transmitted onto the network. When a TRIAL or an empty slot arrives a node, the PHIC provides shift-out pulses to read the slot from the queue, and load the data into its internal FIFO. The cell is shifted in as eleven 16-bit words over the data bus of the PHIC. This bus is used for transmitting and receiving slots to and from the PHIC. The generation of the transmit-request strobe by the queue only takes place after a complete cell has been passed to the queue from the packetiser.

When real-time video transfer is in operation, the number of slots in the transmit queue can only be 0, or 1 assuming that the node with the video service is given a large d-val and the video service is the only application on the network. This is because the ring stops taking data from the transmit queue when there is no data in the transmit queue or the node has run out of d-val value. With the above situation,
the queue temporarily runs out of data to transmit as the ring operates slightly faster than the packetiser. Hence a RESET slot is produce as a result of there being no data in the queue. During this time, the queue will built up to one slots and the process continues. It was therefore not found necessary to make the queue size bigger than 15 slots.

5.3 Receiver station

The operation of the receiver station is essentially a reversal of the transmission station, and contains appropriate access and receiver sub-systems. Because of the method of clock distribution over the Orwell network the receiver section will operate at precisely the same frequency as the transmitter, although data packets will be arriving asynchronously, i.e in an intermittent or 'bursty' manner. The receiver PHIC detects data packets addressed for its receiver station and removes these from the ring, passing them to a receive queue. The receive queue accepts data packets from the PHIC as they arrive over the ring and allows a depacketiser to remove these synchronously. Thus the queue will be expanding and shrinking depending upon the arrival of data. The depacketiser strips the header information from the slot and passes the resultant user data to the parallel to serial converter then to the receiver CODEC and ultimately to a colour monitor.

5.3.1 Access Sub-System

5.3.1.1 Receive Queue

The receive queue card provides an interface between the PHIC and the depacketiser. When a node detects its address in the Destination Address field, the PHIC generates eleven shift-in strobes to shift the complete slot into the receive queue FIFO. The data is presented as 16 parallel bits and a 17th bit is provided to indicate the first word of every cell received. The receive queue on the other hand detects the start of a slot using the 17th bit before accepting the data from the PHIC. A counter is incorporated in the design which is incremented for every
successful slot received from the PHIC and decremented for every slot transferred to the depacketiser.

The receive queue is designed such that the depacketiser starts taking data from the Rx queue when there are eight slots in its FIFO. The receive queue generates a cell-ready line when a cell has been read into its FIFO from the ring. This signal causes the depacketiser to generate a series of shift-out pulses to read information from the queue FIFO. On the completion of this data transfer from the queue, a cell-received strobe is generated by the depacketiser. This signal is used to decrement the counter incorporated in the receive queue design. This mechanism maintains the slot boundary.

5.3.1.2 Depacketiser

The Depacketiser in effect performs the reverse function of the Packetiser. The Depacketiser takes in eleven 16-bit words from the receive queue and strips off the header, leaving only eight words of information to be transferred to the Parallel to Serial converter. The Parallel to Serial converter generates shift-out pulses which are applied to the Depacketiser FIFO. The data from the Depacketiser is Parallel to Serial converted and directed to the Decoder which produces a PAL signal and this is fed into a monitor. The rate at which data leaves the Depacketiser is equal to the rate at which data enters the Packetiser.

For many network applications, uncertainties in data packet delays are not important. For the real-time video system discussed in this chapter, Tx and Rx synchronisation is crucial since the receiver CODEC contains no frame store and hence must operate at the bit rate at which data leaves the transmitter CODEC. Delay uncertainties are overcome by ensuring that the Tx and Rx queues always have sufficient stored data packets to compensate for the maximum delay that will ever be encountered when attempting to access the network.
5.3.2 Video Rx Sub-System

5.3.2.1 Parallel to Serial Converter

The parallel to serial conversion is also performed by a custom designed chip designed by BT that employs both ECL and TTL technology. Data from the depacketiser are transferred to the parallel to serial converter, one 16-bit word every 200ns. This is then latched through to the custom chip device eight bits at a time. See Fig. 5.7. The serial clock at 77.8MHz which is derived from the network clock of 155.52MHz is input to the parallel to serial card. The card performs a parallel to serial conversion and passes this serial data along with a clocking signal at 77.8MHz to the input stage of the receiver CODEC.

![Fig. 5.7 Parallel to Serial Converter.](image)

5.3.2.2 Receiver CODEC

The data from the Rx queue are directed to the receiver CODEC after being converted to a serial data stream. This data arrives synchronously with that sent from the transmitter but will be delayed due to the average transition time over the network. The principle task of the CODEC is to regenerate a PAL video signal
from the serial data and this is achieved by utilising the special data words inserted by the transmitter CODEC to represent horizontal and vertical sync periods. By locking onto these video data words, an analogue PAL video signal is generated containing all of the blanking and sync signals necessary to drive a video monitor.

The receiver must achieve word synchronisation prior to anything else. An initial word synchronisation phase is established on detection of the frame alignment word in the 77.8 Mbps serial data stream. This initial word synchronisation is followed by a search operation for two further error-free frame alignment words separated by video line period intervals. If the two error-free frame alignment words are not found, it is assumed either that the original frame alignment word was data emulating the word or a high error rate was encountered. In either instance, the receiver reverts back to the initial situation of searching through the serial data stream for a new frame alignment word. For the case where the two error free frame alignment words were successfully detected, word and frame synchronisation is taken as established. Subsequently a "flywheel" mechanism maintains frame alignment until three consecutive frame alignment words are found to each contain more than one error. Word and frame synchronisation is then taken as being lost, and the receiver reverts back to re-establishing word and frame synchronisation.

Due to asynchronism between the network clock and video line rate, a windowing technique is necessary to extract the frame alignment words at the receiver. A frame alignment word can be expected either at 745 or 746 video sample periods after the previous one. To overcome the possibility of jitter on the incoming video line syncs forcing the frame alignment word out of the narrow window, the window has been opened up to 16 video sampling periods wide, enabling the system to tolerate up to +/- 0.7 μsec of jitter. Transmission frame synchronisation is normally achieved within a few video line periods. Alarm signals are provided to indicate errors in frame alignment reception, and overall loss of transmission frame synchronisation[29].

The 5-bit video frame alignment word is normally zero, but is set to all ones during the line sync interval at the beginning of video line number 4. Valid detection of the video frame alignment word is assumed when 4 or 5 ones are detected within
the word at the receiver. During the initial video frame lock-up at the receiver, a video line counter locks onto the first valid detection of the video frame alignment word, and the word is checked thereafter in "flywheel" fashion every 625 video lines. A running total is kept of valid detections. If an invalid detection occurs, the total is decremented by one. Valid detection increments the total to a maximum of three. When the total is zero, loss of video frame synchronisation is assumed, and the receiver reverts back to the initial state of locking onto any valid detection of the video frame alignment word. Video frame synchronisation is normally achieved in less than 50msec[29]. The transmission and video frame alignment words are used to regenerate video mixed sync at the receiver.

The video black level is held in the receiver during video blanking intervals. Immediately following the window, 39 words of information can be transmitted before a line sync interval has elapsed. The two 12-bit audio samples and the 18-bit data word are placed at the front end of this section as shown in Fig. 5.8. Some flexibility exists here since the timings for putting information out to line during this time interval are controlled by a PROM. The remaining data during the line sync period is set to the video frame alignment word, apart from the last few words which are returned to video black level. Video data are output to line during all other periods.

![Fig. 5.8 Multiplex configuration.](image-url)
The receiver unit contains clock and data failure alarms. It may be that the clock and data lines into the receiver are of equal phase, such that the receiver system clock edges are too close to the data edges for reliable clocking. To overcome this possibility, a clock phase reversal switch has been included in the receiver. It is important to note that the clock input to the receiver CODEC is half the recovered frequency. This provides an end-to-end synchronous system.

5.4 Conclusion

Although data passes over the ring asynchronously due to the potentially bursty nature of an ATM network, the rate at which video data leaves the transmitter CODEC will exactly match the rate at which it arrives at the receiver CODEC. This is achieved by utilising the inherent synchronisation of the end to end system and the characteristics of the Tx and Rx cell queues, shown conceptually in Fig. 5.6. Since the transmitter and receiver CODECs operate at the same frequency there is no build-up or loss of data that is typical of networks where local and remote clock frequencies differ. The asynchronous Orwell network exists within a rigid synchronous video transfer environment. Provided the allocated bandwidth is sufficient to carry the required amount of data, and transmit and receive queues have sufficient capacity, transmitter and receiver CODECs can be considered to be synchronised, since any uncertainty and jitter caused by the asynchronous nature of the network is completely removed by the Tx and Rx queues. These queues need not be large - up to fifteen complete cells can be stored but typically each queue contains only three or four cells during operation.
Chapter 6
Packetisation and Depacketisation

6.1 Introduction

The adaptation layer allows higher layer information from either the user plane or the control plane to be mapped on to the ATM layer and vis versa. The adaptation layer is service dependent. User data must be presented to the network in the same format as that of the network cell or slot, and the process whereby user data or control information is assembled into fixed sized chunks and the header appended to it to form protocol data units or slots can be termed packetisation or segmentation. In networks with varying packet sizes, a trailer is added to indicate the end of a message. In an ATM network, a trailer is not necessary since all services are carried by a fixed-length transport unit called a cell.[105]

The packetisation process can either be done within the layer 2 protocol operation as is the case with the Cambridge ring[51,70], or outside the layer 2 protocol operation as is the case with the Orwell ring discussed in this project. The advantage gained in separating the task is a reduction in the amount of processing overhead involved. A packetiser is necessary for every transmission node on the network and a corresponding de-packetiser is required for every receiver node (these may co-exist within the same node). Packetisation can be performed as either a software or a hardware task, and each has advantages in terms of flexibility and speed of operation. In hardware packetisation the task of segmenting the data and adding the header is performed purely in hardware. In software packetisation the task of formatting the data into slots is done by a software program which is then transferred into a temporarily storage medium.

Hardware packetisation is used in applications where high data rates are necessary. With continuing high-speed network development, demand for high-speed packetisation is increasing. Hardware packetisation can assemble data into protocol
data units (slots or cells) at very high speed often approaching the network data rate. This operational speed is essential for video communications where data rates can be very high. Hardware packetisation is limited because of its lack of flexibility (the header field may be controlled by a processor to introduce some flexibility). Different services will require a different hardware packetiser design suited to a particular service.

Software packetisation is limited by the speed at which the processor can assemble the data into network cells. The software process allows control signals to be easily modified along with addressing information, providing great flexibility to the user. The software packetisation process is also sufficiently flexible to support a different slot structure providing the network interface remains the same. The drawback with this level of flexibility is a compromise in the data rate the packetiser can support.

This Chapter looks at two packetisation methods developed and used in this project. The first method is hardware-based and permits extremely high data rates to be achieved. The second method is software-based, allowing considerable flexibility but limited data rates. A third method of packetisation is discussed which combines the advantages of both hardware and software designs, and would also eliminate a number of the drawbacks encountered with each design.

6.2 Hardware Packetisation

Hardware packetisation is attracting a lot of interest in the light of recent development in high-speed computer networks as a possible means of supporting high bandwidth services (e.g. video). The hardware packetisation implemented in this work was developed in collaboration with British Telecom and is used for the transfer of constant bit-rate video at a data rate of 77.8 Mbits/s. This is done to show the ability of the network to support very high bandwidth services. A block diagram of the packetiser is shown in Fig. 6.1. Programmable Array Logic (PAL) are used for the implementation of most of the logic. Hence, this enables a reduction in cost and space.
Data from the video subsystem is passed to the hardware packetiser as a continuous stream of 16-bit words. When there are eight words stored in the packetiser's FIFO, the header which constitutes the source and destination addresses and control bits is appended to form a complete slot. This is then passed to the queue card. The slot is of the same size and structure required by the PHIC. Fig. 6.2 summarises the function of a packetiser. A control mechanism in the packetiser board monitors the number of words in the packetiser FIFO, and a counter is incremented for every word written into the FIFO. If there are more than eight words in the FIFO, the packetiser adds the header to the eight words and passes the complete slot to the Queue board.

The handshaking mechanism between the packetiser and the queue card guarantees that only a complete cell is transferred to the queue. The control counter is also decremented for every word read from the FIFO. At the end of a cell transfer, if there are still more than eight words in the FIFO, the header is again added and another cell is transferred to the queue. This process continues until there are no more cells to send.
6.2.1 Depacketiser

The Depacketiser in effect performs the reverse function of the Packetiser. The Depacketiser takes in eleven 16 bit words from the receive queue and strips off the header, leaving only eight words of information to be transferred to the Parallel to Serial converter. The Parallel to Serial converter generates shift out pulses which are applied to the Depacketiser FIFOs. The data from the Depacketiser is Parallel to Serial converted and directed to the Decoder which produces a PAL signal and this is fed into a monitor.

6.3 Software Packetisation

A software-based packetiser was developed to provide multimedia services for the Orwell network. This software-based approach is limited in its ability to support services with high data rate but provides a flexible connection for PC based multimedia services. The 'C' language was used in the development of the packetiser (all the software programmes are included in Appendix II). Using the software packetiser, data can be read from an application board on the PC bus and
transmitted over the network. The functionality of the software packetisation allows easy access to the control and addressing fields within a slot.

The software packetiser is made up of a number of software routines written in 'C' and a hardware interfacing card. The interfacing card constitutes three major sections as shown in Fig. 6.3. The FIFO section contains both transmit (Tx) and receive (Rx) FIFOs which are used temporarily to store data from the PC and from the network respectively. The control section controls the functionality of the card allowing data to be written and read to and from the FIFOs. The address decoder provides control signals for use in different parts of the board. The d-val section is used to check for the validity of a node to transmit.

The board is designed with a level of flexibility that allows its functions to be memory-mapped within a PC. The memory address of registers that control the board functionality are selectable to avoid contention with other boards resident within the PC and this gives the potential for a number of packetiser boards to be co-resident within the same PC.

Fig. 6.3 Software Packetiser interface card.
The packetisation function is controlled by accessing on-board registers. The decoded addresses and their functions are shown in Table 6.1 below. Each board covers a range of addresses, and the base address is selectable. In the example given, address 300H is the base address and it holds the board’s code word used for its identification. The same base address is used for the writing of the d-val for that particular service. Address 302H is used to latch control signals which controls the board operation. The status of the FIFOs (e.g full, half-full and empty flags) may be checked using address 304H. Finally, address 306H is used to read data from the Rx-FIFO and also to write data into the Tx-FIFO.

The software packetiser can be used to identify the base address of a particular board and reset the board ready for operation, i.e. ready to receive or transmit data from and to the network. Network transmission is accomplished by writing 11 words into the Tx-FIFO with the first three words representing the header field and the rest representing the information field. After a complete slot has been written into the Tx-FIFO, one of the control signals is enabled to indicate its readiness to transmit data over the network. If the d-val is greater than zero, then a request to transmit is then presented to the PHIC. When an empty (TRIAL) slot arrives at the node, the PHIC will then provides a series of clocking signals to shift the data from the Tx-FIFO and subsequently over the network. This 'one shot' mechanism tends to be inefficient, and the software is written such that any number of slots up to a maximum of 23 slots can be prewritten into the Tx-FIFO and transmitted in sequence, at the maximum ring determined rate. The number of slots

### Table 6.1 Register Mapping.

<table>
<thead>
<tr>
<th>Address Mapping</th>
<th>Register Mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link 1</td>
<td>Link 2</td>
</tr>
<tr>
<td>Down</td>
<td>Down</td>
</tr>
<tr>
<td>Down</td>
<td>Up</td>
</tr>
<tr>
<td>Up</td>
<td>Down</td>
</tr>
<tr>
<td>Up</td>
<td>Up</td>
</tr>
</tbody>
</table>
is limited to 23 because of the maximum size of the FIFOs. The half-full flags of both FIFOs are used as a control mechanism to maintain transmit and receive slot boundaries.

The software depacketisation is very similar in operation to the software packetisation. When data arrives from the ring, the depacketiser strips part of the header (the start delimiter and the destination address) before downloading it into the Rx-FIFO. As a result, only nine words per slot are written into the Rx-FIFO, i.e. one word for the slot's source address followed by eight words of data. In a situation where sub-addressing is applicable, the data received by the node is directed to the various subnodes according to the destination address field.

6.4 An alternative Hardware Packetiser based on Dual port RAM

One possible alternative to the utilisation of FIFO devices in the packetiser design is to employ random access memory (RAM) rather than FIFOs as the temporary data storage medium. This packetiser is described with reference to the present Orwell slot structure. Data from the external device is written into the dual port RAM and stored until a signal is received requesting data for the ring. Read and write operations occur on separate ports. The dual port RAM consists of an input address port (left) and an output address port (right). A schematic diagram of the system is shown in Fig. 6.4.

On initialisation a microcontroller is used to load the dual port RAM with a three word header starting at every eleventh address. This header is specified on a set of DIL switches although subsequent development work will allow the header to be downloaded from a microcomputer. The data from the data source is written to the eight memory locations between each set of three words. Hence, the required Orwell slot format of three header words followed by eight data words is generated. Each word stored in this RAM consists of eighteen bits: sixteen are data bits and two are control bits used in the internal operation of the packetiser.
The parallel output of an eight bit counter is fed directly into the first eight bits of the output address port. Starting with an initial count of zero, the counter increments until it reaches a predetermined maximum address and is then reset to zero. Hence the data in consecutive memory locations appears on the output port of the dual port RAM. A slot counter circuit ensures that no read operation may occur unless there is sufficient valid data to produce a data slot.

Writing to the dual port RAM is slightly more complex since only eight out of every eleven memory locations may be written to. The single port RAM contains the sequence of allowed addresses in ascending order, stored in consecutive memory locations. These are loaded into the single port RAM on initialisation by the microcontroller. The addresses for the single port RAM are generated by an
eight bit counter. The counter steps through the single port RAM addresses making the data appear on the output port. This data is fed directly into the input address port of the dual port RAM. This ensures that the data written into the dual port RAM is always sent to the next available allowed memory location so that the present header words are not overwritten.

### 6.5 Comparison of the two hardware based Packetisers

The FIFO based hardware packetiser uses a FIFO memory to store data. The packetiser outputs the header and then eight words of data to the transmit queue. The slots are stored in a second FIFO before being sent onto the network. An extra write signal may be erroneously generated when data is being sent from the packetiser to the transmit queue by, for example, noise. An error of this nature would pose a serious problem since a slot with the wrong format would be transmitted onto the ring, which will corrupt the network slot structure. This also leads to error extension characteristics, as the cell boundaries for the following cells will be corrupted. Lost of synchronisation is unlikely to occur in the dual port RAM based system, since if an extra write to RAM was generated, there would simply be a word of erroneous data in the data area of the dual port RAM. The format of the slot would remain unchanged. The two cases are illustrated in Fig. 6.5 where H, D, and E represent header words, data words and erroneous data words respectively.

Another possible failure mode of the FIFO based packetiser described previously is FIFO skew. Two eight-bit wide FIFOs are used in parallel to make up a 16-bit wide fifo. This then allows a situation to occur where one of the FIFO pair contains either fewer or more words than the other due to noise on the reading or writing signals. The resultant effect is that a corrupted slot will be shifted onto the network. When this happens and the header field is corrupted, the slot will circulate around the ring until removed by the monitor node. If skewing occurs at the start of a series of slots stored in the FIFO, the entire group of slots will be corrupted and will subsequently be lost. In consequence the performance of the network is greatly reduced.
A slot with one erroneous word

A slot with one erroneous word

The next slot is not of the Orwell protocol slot structure

A, FIFO based system

The next slot is still of the Orwell protocol slot structure

B, Dual port RAM based system

Fig. 6.5 The effect of an error on the slot structure in the FIFO and dual port RAM based packetisers.

An additional feature of the dual port RAM based packetiser which was not incorporated in the FIFO based system is the detection of output errors. Each data word in the dual port RAM has two control bits; one of these bits indicate if the word is the first word of a slot. If the first word being sent did not have the correct control bit then an error flag is set. This could be used to initiate a course of remedial action. If the address of the dual port RAM was corrupted, synchronisation is still not lost since the data being written into the dual port RAM may go to the wrong address but not one which contains a header word. Therefore the slot format is preserved and the network is not disrupted.

Furthermore the lack of a mechanism that locks onto the slot boundary at the packetiser level is another problem. This is because once there is a slip in the slot boundary, only corrupted slots are produced until the packetiser is reset. However,
if there was a mechanism to detect the start of a slot on every slot, and there was a
slip in the slot boundary, only one slot will be corrupted since the system will lock
onto the next slot boundary. This means that with the mechanism introduced, the
packetisers will become self-healing.
Chapter 7

Performance Analysis

7.1 Introduction

This chapter briefly describes the work carried out in this research project that deals with the practical aspects of transmitting Constant Bit Rate (CBR) video over an experimental Orwell ring. The objective is to provide a very high bandwidth link for users with high bandwidth services. Other services were added to expand the network to include multimedia applications. From this, an insight into the performance of the system as a whole, and particularly of the operation of the Orwell protocol in sharing bandwidth among nodes (d-val allocation) is obtained.

7.2 Ring Configuration

The setup of the experimental Orwell ring is shown diagrammatically in Fig. 7.1. The ring operates at a physical layer speed of 155.5 Mbits/s with two slots circulating around three nodes A, B and C. The cell (slot) structure used conformed to the latest specification for the protocol at the time when the Protocol Handler Integrated Circuit version A (PHIC A) was developed i.e. 22-byte cell with 6-byte header. On this experimental ring, not all the functions specified in the Orwell protocol have been implemented. These are not necessary since the number of connections on the ring has been restricted to a one-way 77.8 Mbits/s CBO video and data connection.

A block diagram of node B layout is shown in Fig. 7.2. Data from a laser disc player are fed into a CODEC which produces a constant bit rate signal at 77.8 Mbits/s. This signal is serial-to-parallel converted and loaded into a packetiser which packetises the data and passes it to a transmit queue. The transmit queue has a maximum buffer size of 15 packets which corresponds to about a line of the
Fig. 7.1 Orwell Ring Configuration
Fig. 7.2 Transmitter Node Configuration
video signal. Hence there is no framestore involved. The small buffer is included to compensate for any jitter introduced by the asynchronous nature of the ATM network. The use of a microprocessor as the controller provides the flexibility for those functions not implemented to be added as and when required. Node B is set up as the transmitter of a constant bit rate source at 77.8 Mbits/s. Other services could be added if required.

A block diagram of node A is shown in Fig. 7.3. Node A is set up as the monitor node which is responsible for the generation of the first TRIAL slot to start the communication process. Node A is also configured as the receiver of the constant bit rate video from node B. Data destined for node A are taken out of the ring and temporarily stored in the receive queue. The buffer size of the receive queue is also made equal to 15 packets. This again compensates for the jitter introduced by the characteristics of the network. The depacketiser takes in the data and strips off the header and passes it onto the CODEC to be decoded and sent to the monitor (TV screen) for the reconstruction of the images. Initially the depacketiser can only take data from the receive queue when there are eight slots in the buffer.

Node C as shown in Fig. 7.4 was developed in the early stages of the project primarily to act as a repeater station (node) and to confirm that the network can perform true ATM switching; i.e high bandwidth data could be transmitted over the network via node C which would switch the data on the basis of the contents of the destination address in each data slot. The node could in practice receive data if the data slot was addressed to it but was essentially idle as it contained no attached transmit and receive application system. This slave node concept was found to work entirely as expected.

In the later stages of the project an interface was developed that allowed four PCs (subnodes) to share the same node, each having full transmission and reception capabilities. The four PCs were addressed using the MAP address which has already been fully described in Chapter 4 but will be briefly mentioned here. The destination address is split up into sections, with the six most significant bits of the destination address used for addressing MAP1 and the next six bits used for addressing MAP2. It is possible by making an entry in each MAP to address up to
Fig. 7.3 Receiver Node Configuration
Fig. 7.4 Node C Configuration
64 subnodes. The node will receive the data slot if there is a MAP address match. The data slot is then directed to the subnode with its address in the destination field. The inclusion of transmission and reception applications on node C also added a lot of flexibility to the network making it possible to assess the performance of the network by using one of the subnodes as a measuring tool.

7.3 Bandwidth Allocation Mechanism

In an Orwell network, a cell is launched into an empty slot when the slot arrives at an active node (a node with data to transmit). It is retrieved by a node or group of nodes in the case of multicasting that recognises their addresses in the cell header field. The slot is then converted back into an empty slot which can be used by any node downstream. Cells are delayed at a node whilst waiting for an empty slot, but the protocol limits this cell delay to an upper bound for delay-sensitive services like voice. With the destination release policy, a slot can deliver more than one cell during a single rotation on the ring. This policy may, however, result in 'hogging', where nodes downstream from a high usage node may be completely deprived of their share of the empty slots and hence bandwidth. This could lead to unacceptable cell delay and eventually cell loss as a result of buffer overflow. In order to prevent such an occurrence, the Orwell protocol limits the number of cells that a node may launch onto the ring before being re-initialised. This number is termed the d-val of the node and it may be dynamically adjusted to meet the demands of the node subject to bandwidth availability.

Each node has a d-val counter which is set to an initial value and decremented for each packet sent. Each d-val is allocated depending upon the node's bandwidth requirement. When a node has sent d-val packets or has run out of packets to send, it is forced to a PAUSE state and other nodes are allowed to transmit until they too PAUSE. When this happens, a reset operation is needed to restart activities again. Nodes in the IDLE or PAUSE states are unable to fill passing empty slots. However, they still perform an essential role by converting all such slots into TRIAL slots.
A TRIAL slot is an empty slot with a special bit set in the slot control field and the destination address set to that of the node originating the TRIAL. As the slot is still marked as empty, it may be filled by any other node, in which case the TRIAL is terminated. If a TRIAL is not taken by any other node, it returns successfully to its sender. Provided the sender still has no packets to transmit, it is then converted into a RESET slot by a suitable change in the control field and travels round the ring causing each nodes d-val counter to be reset to the d-val for that node. This reset mechanism is fully distributed.

Any node can perform this role which leads to a greater robustness of the system. However because several nodes may have successful TRIALs on a multi-slotted ring the problem of 'ghost' resets occurs. This means that one node issues a reset when there is already another on the ring. If both reset slots were allowed to go fully round the ring each node would experience a reset followed very rapidly by a second 'ghost' reset. This must be avoided since the time between successive resets is taken to be a measure of the current ring utilisation. To avoid 'ghost' resets the protocol operates with two boolean variables OUTSTANDING TRIAL and OUTSTANDING RESET. Basically, when a node issues a RESET slot it marks itself as an OUTSTANDING RESET node. RESET slots travel round the ring resetting d-val counters as already described. However when they reach any OUTSTANDING RESET node they are returned to ordinary empty slots. In this way a number of RESET slots actually cover different arcs of the ring resetting the nodes. OUTSTANDING TRIAL guards against a single node issuing more than one RESET slot.

On the experimental network, node C can support four queues carrying different services with the highest priority service connected to port 1 and the lowest priority service connected to port 4 as shown in Fig. 7.4. Each queue is given a d-val depending on their bandwidth requirement. Before a TRIAL or an empty slot arrives at Node C, a check will have been conducted to determine which queue has data to transmit. Every time a slot is transmitted, a check is carried out starting from port 1, then port 2, port 3, port 4 and then back to port 1. Hence, if all the queues have data to transmit, then the highest priority queue i.e port 1 will gain access to the empty slot. Its d-val will be decremented by one for every slot
transmitted. The highest priority queue will carry on transmitting data until it either runs out of data to transmit or its d-val is zero (hence no bandwidth available for that service). The next highest priority queue will then gain access and the process continues until the node’s d-val is zero or there is no more data to transmit. When this happens, a reset process occurs which re-initialises the d-val for all the nodes, and the subnodes within each node and the process starts again.

7.4 System Performance

Cells or slots to be placed on the ring are formatted (packetised) outside the PHIC and this task can be performed as a hardware or software function. One significant advantage of this technique is that the PHIC is freed of the intensive task of packetisation and depacketisation and this allows exceptionally high access bandwidths to be achieved. The maximum data access rate possible through an Orwell node running at a physical layer speed of 155.52MHz can be calculated thus:

\[
\text{access rate} = \left( \text{physical layer frequency} \times \frac{\text{data payload}}{\text{slot size}} \right) \times \frac{\text{coding ratio}}{1} - \text{reset overhead}
\]

Let \( X = \left( \text{physical layer frequency} \times \frac{\text{data payload}}{\text{slot size}} \right) \times \frac{\text{coding ratio}}{1} \)

where

\begin{align*}
\text{physical layer frequency} &= 155.52 \text{ Mbits/s} \\
\text{data payload} &= 8 \times 16\text{-bit words} \\
\text{slot size} &= 11 \times 16\text{-bit words} \\
\text{coding ratio} &= 4/5 \quad (\text{due to 4B/5B coding}) \\
\text{Reset Overhead} &= \frac{2}{(d-\text{val}+2)} \times X
\end{align*}

\( d-\text{val} \) is the value allocated to every node on the network to be used during transmission of data.

Within the physical layer, each 4-bit word is actually transmitted as a 5-bit symbol, making the physical layer cell size to be 220 bits, hence the coding ratio. Two slots are used for every reset process, hence the 2 in the equation. For large value of \( d-\text{val} \) (maximum of 65535), reset overhead is negligible.
Thus this gives an achievable peak node access rate of 90.48 Mbits/s with no other traffic on the network.

A number of slot types are used to transmit either user data or control information over the network:

- **T-slot**: Used by the monitor station to assess the ring size
- **S-slot**: Used during ring synchronisation
- **TRIAL slot**: Available for use by any node wishing to transmit data.
- **RESET slot**: Control slot used to reset every node’s d-val
- **EMPTY slot**: Used to carry user data

The throughput of the ring is ultimately limited by the single path. While a single ring speed of 155.5 Mbits/s, achievable with today’s technology (and possibly 1 gigabit/s in the next decade), may be sufficient for private network needs, it is inadequate for public network use. To meet the requirements of a public network switch, rings can be stacked in parallel and their separate capacities made to appear as a single capacity Orwell ring as shown in Fig. 7.5. This multiple ring system is called an Orwell Torus and besides offering increased capacity, the Torus could potentially provide the reliability, ease of maintenance and expansion that is required for a public switch. The throughput of the Torus will still be limited by the maximum number of rings that can be stacked together. For more details of the Orwell ring and Torus, see references.[106,107]

Measurements were taken to determine the video buffer size when real time video transfer was in operation. The first test was carried out when the video service was the only application on the network, i. e. when Node B was transmitting to Node A. It was discovered that when Node B was given a large d-val, say 255, the number of slots in the transmit queue was either 0 or 1. This is because a RESET slot occurred only as a result of data running out in the transmit queue. If the d-val was reduced to the point where the occurrence of a RESET slot was due to the lack of a d-val, the maximum number of slots in the transmit queue would increase to 2. With this situation a state of equilibrium exists where the bandwidth allocated to the video service exactly matches the bandwidth required by the video service. Any
further reduction of d-val would cause the video service transmit queue to increase, ultimately leading to queue overflow and a corresponding loss of cells.

A further experiment involved adding another application onto the network and observing the effect on the real time video service. Hence while Node B was transmitting constant bit rate video data to Node A, Node C was transmitting computer data to itself. The result showed that with the video service allocated a constant d-val of 25, the peak number of slots in its transmit queue increased as the d-val for Node C (computer data) was increased. When this d-val was increased to 4, the video service queue became full and caused cell losses and the video received by Node A was ultimately affected in the sense that there were break-ups in the images on the monitor.

This cell loss is illustrated in principle in Fig. 7.6. For small values of d-val given to the computer data service, the video service adapts to the sudden reduction of available bandwidth by temporarily storing untransmitted cells in its queue, hence the video queue will increase. This increase will be in relation to the size of d-val
allocated to the computer data service. If this service is given a sufficiently large d-val (which corresponds to bandwidth), the video service is unable to adapt since its queue becomes full and further video data cells will be lost as illustrated by the shaded areas.

Increasing the d-val of Node C also affected the receive queue buffer in the sense that the receive queue FIFOs would temporarily go empty and the images on the monitor would be distorted and hence unacceptable. The system was designed such that the depacketiser starts taking data from the receive queue only after an initial eight slots have been stored.

### 7.4.1 Cell Delay and Jitter

Cells from the packetiser are passed onto the transmit queue before gaining access onto the ring. Cell delay is caused by cells having to wait in the transmit queues for
access onto the network and associated component propagation delays. The asynchronous nature of the ATM network and the variation in network load, means that the waiting times in the transmit queues are not constant, and the varying cell delay results in cell jitter. In order to make the cell transport mechanism transparent to the services, particularly the CBO services, this jitter must be compensated for at the receive queue to restore synchronisation of the bit stream. The size of the buffer required in the receive queue is dependent on the magnitude of the jitter. This will also affect the end-to-end cell delay.

The total end-to-end cell delay is subject to network loading conditions and will incur both fixed (ie known) delays and variable (uncertain) delays due to the characteristics of the ATM network. Fixed delays will be predictable and will be introduced by all physical components of the network (ie electronic components and optical fibre), whereas variable delays and uncertainties are generated by the basic operation and asynchronous nature of the network.

The fixed delay that a data packet will experience will relate to the number of nodes on the routing path and the fibre distance between source and destination nodes. Additional delay will be introduced by the user application hardware and node access hardware, but all delays of this type are inherently fixed and are therefore predictable. However, the time that a data packet takes to travel from source to destination will be uncertain as the overall delay also depends upon network access times and bandwidth availability.

When a data packet is ready for transmission it must wait at the transmission node for a TRIAL or an EMPTY slot to arrive before being placed on the network. If a slot is unavailable it has either been taken by another node or it is being used for network protocol purposes. Thus there will be an element of uncertainty in the time it takes the packet to access and subsequently traverse the ring which will clearly vary with network traffic load.

Delays in accessing the ring must be kept within predetermined limits, typically 125μs for a LAN and 2ms for a MAN. Research shows [39] that for a maximum reset interval of 125μs, an Orwell ring/torus can be configured with 30 nodes and a
3Km ring length limit. To allow Orwell to compete with DQDB and FDDI2, the maximum reset interval can be increased to 2ms, since there may be more than 30 nodes and the ring may be several kilometers in length (hence greater ring latency). The PHIC design used allows the Maximum Reset Interval (MRI) to be set to approximately eight values between 62.5μs and 2ms.

For the Orwell network discussed in this project, only one node is transmitting thus eliminating competition from other network users for empty slots and therefore much of the delay uncertainty is removed. Even in a multi-user network, the Orwell protocol will limit this uncertainty. Variable delays will still occur as the transmitting station encounters non-empty slots used for network protocol purposes. Clearly the Tx station cannot use these slots, and must wait for an empty slot, causing uncertainty (jitter) in the rate at which data packets are placed onto the network and then subsequently received.

Measurements indicate that each network node introduces a ring-path delay of typically 500ns, with the optical fibre introducing a further 500ns per 100 metre length. These delays are fixed and are not dependant upon network loading or other factors. The overall data route from Tx to Rx, i.e through the Tx video packetiser, over the network, and on to the Rx video depacketiser encounters a delay of typically 23μs with a maximum of 2.8μs of timing jitter in the received data, although this first figure is dependant upon the state of both Tx and Rx queue lengths. Clearly the level of timing jitter will increase with network loading.

7.4.2 Cell Loss

Cell loss can be caused either by bit error in the cell header or buffer overflow due to network overload. ATM networks should have a design goal of a cell loss rate of no greater than the bit error rate[102] and effective load control should enable the CBO services to experience cell loss rate of the same order. It is unclear what this loss rate should be although figures in the order of 10^{-6} to 10^{-9} have been quoted[26,99]. Cell loss has a more serious impact on picture quality than bit errors
because it introduces discontinuity into the data stream and causes loss of synchronisation.

7.5 Conclusion

The system in operation at Loughborough University is experimental but has proved highly reliable during prolonged tests where uninterrupted video transmission over periods of up to ten hours has been achieved. The network comprises three nodes interconnected by 5 metre lengths of optical fibre cable, with each node existing within a separate 19" rack. Each rack houses cards to implement the basic node function - i.e physical layer Tx and Rx repeater cards that provide the interface between the high-speed fibre level and the PHIC, and the PHIC card itself which handles all protocol tasks (Fig. 7.7). Further cards are contained in the racks designated as transmitter or receiver nodes, and these are application specific, implementing the Rx and Tx queueing and packetisation functions discussed previously.

A photograph of the full system is shown in Fig. 7.8. Further elements of the system can be identified - at the transmitter node the laser disc player used as the video source, and the Tx CODEC, and at the receiver node the corresponding Rx CODEC and the video monitor. An IBM AT controls and monitors the status of the network, allowing the user to set up and synchronise the network, and also provides a unique real-time 'snapshot' of the slot structure and data passing around the network.
Fig. 7.7 Photograph showing prototype PHIC board (left), and Physical layer Rx and Tx development boards. The large heat sink covers the data and clock recovery module on the receiver.
Fig. 7.8 The complete Orwell network and video system showing transmitter and receiver equipment and three network nodes.
Chapter 8

Discussion and Conclusion

8.1 Discussion

An ATM switch broadband network provides many advantages over current dedicated networks. It provides greater flexibility and is typically more cost effective because many different types of application can be accommodated. This thesis has discussed the design, development and implementation of a high speed Orwell network that can provide true ATM services with significant advantages over other ATM networks - ease of network and node synchronisation, simplicity of node design, very high node access bandwidth, and dynamic control and allocation of total network bandwidth.

The Orwell protocol implemented here is based on a slotted ring concept with an integral number of slots circulating around a communication network. A network will comprise a number of nodes that allow data to be placed onto or received from the ring, interconnected by a serial communication medium such as optical fibre cable. ATM as the future carrier of high bandwidth real-time services is still at the standardisation stage. Some standards, including the cell structure, have been agreed. The Orwell network described in this thesis operates at a physical layer speed of 155.52 Mbits/s and does not adhere to the current standards, but nevertheless demonstrates the tremendous potential of ATM. The described system meets all of the aims of ATM, namely the ability to provide guaranteed bandwidth on demand with limited variation in transmission delay (jitter) by carrying all types of traffic in a fixed size cell. The cells are routed to their destination by the information in the header field.

The network developed comprised three nodes (although more could have been added) which allowed data to be placed onto or received from the ring. The nodes were interconnected by optical fibre cable. For a ring network to operate correctly
every node must be capable of receiving slots, extracting or inserting information, and maintaining and retransmitting the slot structure to the next node on the network. To accomplish this every node on the Orwell ring contained a Protocol Handler Integrated Circuit (PHIC) and independent physical layer Transmit (Tx) and Receive (Rx) boards. The PHIC device implemented the full Orwell protocol and was responsible for handling all tasks associated with switching data onto and from the ring. The physical Tx and Rx boards provided the interface between the optical fibre layer and the PHIC.

The network described in this thesis could support a large number of nodes that could have a wide geographical distribution due to the inherent synchronisation method used and the optical fibre cable linking each station. Typically, node separations of up to 2 kilometres are possible. There is no constraint upon either the position of nodes within the network, or within the spacing between individual nodes, provided the maximum ring delay does not become excessive. The network could therefore be precisely tailored to an environment such as a commercial site or metropolitan network, and could be extended or reconfigured without difficulty.

It is inherent because of the method of network synchronisation that all nodes on the Orwell network would be running at precisely the same frequency (but with a phase shift), and consequently circuitry associated with a node can be synchronised to circuitry on any other node on the network. In a larger system, the master frequency would be generated from a national source. Because this is regenerated at each node, all service connections can be synchronised to the national clock.

The Orwell network developed has demonstrated the network potential of ATM. Orwell is a protocol rather than a fixed communication system. The implementation discussed here used a non-standard cell size, but the protocol itself is not tied to any specific cell size. The development of new PHIC devices and a modification of the physical layer hardware would permit the Orwell network to conform with current standards, specifically with regard to the cell size. Thus there would be no requirement to convert cells from this implementation to standard ATM cells since the Orwell protocol could support this directly.
Another feature of the work was the provision of high speed network access. Each node on the Orwell network was capable of an achievable peak user data access rate of 90.48 Mbits/s. Thus a network user with a very high bandwidth application could generate data rates of this magnitude over the network. This peak figure was subject to network loading conditions and will incur both fixed (ie known) delays and variable (uncertain) delays due to the characteristics of the ATM network. Fixed delays are predictable and are introduced by all physical components of the network (ie electronic components and optical fibre), whereas variable delays and uncertainties are generated by the basic operation and asynchronous nature of the network.

Experiments were performed involving the transmission of a constant bit rate video signal that allowed uncompressed colour video images to be transmitted at full television resolution and frame rate over the network. These experiments were highly successful, achieving an access rate of 77.8 Mbits/s (ie half the physical layer speed of the network), giving high quality pictures as well as an audio channel. The system was developed primarily to demonstrate that an ATM network running the Orwell protocol could offer a very high bandwidth guaranteed service which could be used for example to support multiple highly-compressed video streams. No attempt was made to define any level of network management.

On the Orwell network data was presented to the PHIC in the form of data packets of precisely the same length and structure as a slot, and the process which formats user data, destination address and control information is termed packetisation. For the video application, the process was performed purely in hardware to accommodate the very high data rate involved. A packetiser was necessary for every transmission node on the network and a corresponding depacketiser is required for every receiver node. The removal of the packetisation function from the tasks performed by the PHIC gave the user a greater bandwidth channel onto and from the network than would otherwise have been possible.

A queue card was developed that provided an interface between the packetiser and the PHIC card, taking complete packets of data (equivalent to slots) from the packetiser and temporarily storing them before the PHIC accepted them for
transmission over the network. This ensured that complete slots were available before the PHIC attempted ring transmission. The queue card had to be capable of operating at least as fast as the packetiser. Each queue was capable of supporting multiple packetisers so that a number of different services could all use the same node. Different priorities could also be assigned to each service. For the video system implemented, only one service was operational so only one packetiser was required.

For many network applications, uncertainties in data packet delays are not important. For the real-time video system discussed in this thesis, Tx and Rx synchronisation was crucial since the receiver CODEC contained no frame store and hence had to operate at the average rate at which data arrived over the network. Delay uncertainties were overcome by ensuring that the Tx and Rx queues always had sufficient stored data packets to compensate for the maximum delay that would ever be encountered when attempting to access the network.

Practical multimedia systems require a range of bandwidth and connection characteristics to be available. The Orwell system was therefore developed to cope with this by providing multiple connections per node. A software based packetiser/depacketiser was developed which interfaced to a PC bus architecture. This software-base approach was limited in its peak network access potential but provided a flexible connection for PC-based multimedia services.

8.2 Conclusion

This Thesis has described the realisation of a 155.52 Mbits/s fibre optic physical layer from the Orwell protocol based on an ATM technology, and has also discussed how Orwell can offer broadband transport capabilities and support multimedia services. Development of the system has demonstrated that an exceptionally high level of point-to-point bandwidth is possible over an ATM network and that the bandwidth allocation mechanism on the Orwell network guarantees this bandwidth regardless of demands from other stations. A simplex real time video connection is supported, with quality essentially equal to that of an
off air TV signal. This connection is at a constant bit rate of 77.8 Mbits/s and is connected via an adaptation layer implemented totally in hardware. The use of a totally hardware-based end to end data path has resulted in the provision of very high user data rates. This was implemented to demonstrate that a very high guaranteed bandwidth service could be supported, although in reality it would be better to use this bandwidth to support a greater variety of traffic sources. The system in operation is experimental but has proved highly reliable during prolonged tests where uninterrupted video transmission over periods of up to ten hours has been achieved.

8.3 Further work.

This work has centred on the development of an ATM network which implements the Orwell protocol and operates at a physical layer speed of 155.5 Mbits/s. The work was sponsored by BTRL and the network is now functional, consisting of three independent nodes connected by optical fibres to form a ring. One feature of the project is the provision of very high speed network access. The network supports a real time connection at 77.8 Mbits/s, giving high quality pictures as well as an audio channel. Lower rate services are also supported through PC based systems. A great deal of additional work is required in order to integrate this work in a 'real world' environment.

Problems such as policing policies are being addressed by others (e.g RACE), but little attention has been paid to the need to generate revenue from the users of future multimedia services. Research currently in progress at Loughborough University sponsored by SERC is investigating the feasibility of a charging mechanism based upon statistical monitoring techniques to reduce the amount of data otherwise required to manageable levels. The objective of the research is to demonstrate that the monitoring of just a few cells within a call will provide sufficient information to enable the charging function to be implemented. A method of charging will be developed that provides the operator with a fair return whilst at the same time giving the user the efficiency benefits that future multimedia services such as variable-bit-rate (VBR) video could produce. The work is carried out by
simulation studies and the verification of a simple network model will be aided by measured results from the high speed network that has been developed.

In the work carried out in this thesis, no network management tasks were supported except that implicit in the Orwell access protocol. A fruitful area of research could involve implementing network management especially in the areas of policing, call set up and signalling.

When a local area network (LAN) is used purely for local computer networking the main performance measures of interest are the overall throughput and the obtainable point-to-point bandwidth, together with the expected message delay time. Recently, however there has been increasing interest in using LANs to carry voice, leading to the multiservice mode of operation. This gives rise to new constraints and problems relating to performance. In particular the LAN must operate efficiently at a much higher transmission rate and due account must be taken of the delay critical nature of the voice service. Research in this area would be beneficial both to the network users and operators.

The Orwell slotted ring developed is more likely to meet the needs of users on private networks. A major feature of the system is that it can be configured from site to site in a simple way, to meet widely different demands in terms of the services used. It also enables new services such as variable-bit-rate video to be introduced which can take advantage of the flexible bandwidth allocation provided by the switch. While a single ring with speed of about 155 Mbits/s is more than adequate for the loads arising in private network applications, it is not sufficient to meet the loads arising in a public network switch.

For performance under fault conditions it is also undesirable to use a single ring to switch the entire load. For maintenance work to be carried out it would be a requirement that a faulty card on a node can be pulled out without losing the whole switch. Research could be carried out to investigate how the Orwell Torus would solve the above mentioned problems. An Orwell Torus consists of a number of parallel rings and a number of nodes interconnected by the rings. Each node is connected to all of the rings to ensure correct operation of the access protocol.
Every ring has the same channel rate and the same latency (end to end delay). It would be capable of providing very high bandwidth capacity (an effective bandwidth of several gigabits/s) and a means for the addition/removal of a node while maintaining existing connections.

The bandwidth requirement for multimedia traffic is continuously increasing. High-definition television standards are emerging. An HDTV frame of two mega-pixels, 24 bits deep, transmitted at a modest 25 times per second yields 1.2 gigabits per second. Owing to the high data rates, and the desire for low-complexity workstations, it is likely that only relatively simple compression algorithm will be employed, perhaps achieving a compression factor of 10. This will result in bursty streams of around 100 megabits per second and these are ideally suited for transmission over a statistical, multiplexing packet switched network. Gigabit networks would appear most felicitous in terms of cost, performance and complexity.

Another aspect of research that could be very useful is to investigate how the present network can interwork with and interconnect to other networks. This could be achieved through the use of gateways and bridges.[30]

It is hoped that this research has provided the basis for further studies in the area of high speed networks in the future.
REFERENCES


65. Leopold H. et. al., "Will B-ISDN services meet the needs of distributed multimedia communications?", 4th IEE Conference on Telecommunications, Armitage Centre, University of Manchester, April 93.


98. Tanenbaum A. S., "Computer Networks", Prentice Hall, Int. Inc


Software controller for IEEE488 interface

This software developed by British Telecom and adapted at Loughborough University. Language used was GWBasic on a PC.

10 ' Copyright Hewlett-Packard 1984, 1985
20 '
30 ' Set up program for MS-DOS HP-IB I/O Library
40 ' For independent of the PC instrument bus system
50 '
60 CLS
70 DEF SEG
80 CLEAR ,&HFE00
90 I=&HFE00

100 ' PCIB.DIR$ represents the directory where the library files
110 ' are located
120 ' PCIB is an environment variable which should be set from MS-DOS
130 ' i.e. C:SET PCIB=C:\LIB
140 '
150 ' If there is insufficient environment space a direct assignment
160 ' can be made here, i.e
170 ' PCIB.DIR$ = "C:\LIB"
180 ' Using the environment variable is the preferred method
200 '
210 PCIB.DIR$ = ENVIRON$("PCIB")
220 I$ = PCIB.DIR$ + "\PCIBILC.BLD"
230 BLOAD I$,&HFE00
240 CALL I(PCIB.DIR$, I%, J%)
250 PCIB.SEG = I%
260 IF J%=0 THEN GOTO 330
270 PRINT "Unable to load."
280 PRINT "(Error #;J%;")"
290 STOP
300 '
310 ' Define entry points for setup routines
320 '
330 DEF SEG = PCIB.SEG
340 O.S = 5
Establish error variables and ON ERROR branching

DEF.ERR = 50
PCIB.ERR$ = STRING$(64,32)
PCIB.NAME$ = STRING$(16,32)
CALL DEF.ERR(PCIB.ERR,PCIB.ERR$,PCIB.NAME$,PCIB.GLBERR)
PCIB.BASERR = 255
ON ERROR GOTO 830
J=-1
I$=PCIB.DIR$+"\HPIB.SYN"
CALL O.S(I$)
IF PCIB.ERR0 THEN ERROR PCIB.BASERR

Determine entry points for HP-IB Library routines

I=0
CALL I.V(I,IOABORT,IOCLEAR,IOCONTROL,IOENTER)
IF PCIB.ERR0 THEN ERROR PCIB.BASERR
CALL I.V(I,IOENTERA,IOENTERS,IOEOI,IOEOL)
IF PCIB.ERR0 THEN ERROR PCIB.BASERR
CALL I.V(I,IOGETTERM,IOLOCKOUT,IOLOCAL,IOMATCH)
IF PCIB.ERR0 THEN ERROR PCIB.BASERR
CALL I.V(I,IOOUTPUT,IOOUTPUTA,IOOUTPUTS,IOPPOLL)
IF PCIB.ERR0 THEN ERROR PCIB.BASERR
CALL I.V(I,IOPPOLLC,IOPPOLLU,IOREMOTE,IORESET)
IF PCIB.ERR0 THEN ERROR PCIB.BASERR
CALL I.V(I,IOSEND,IOSPOLL,IOSTATUS,IOTIMEOUT)
IF PCIB.ERR0 THEN ERROR PCIB.BASERR
CALL I.V(I,IOTRIGGER,IODMA,J)
IF PCIB.ERR0 THEN ERROR PCIB.BASERR
CALL C.S
I$=PCIB.DIR$+"\HPIB.PLD"
CALL L.P(I$)
IF PCIB.ERR0 THEN ERROR PCIB.BASERR
GOTO 960

Error handling routine


830 IF ERR=PCIB.BASERR THEN GOTO 860
840 PRINT "BASIC error #";ERR;" occurred in line ";ERL
850 STOP
860 TMPERR = PCIB.ERR
870 IF TMPERR = 0 THEN TMPERR = PCIB.GLBERR
880 PRINT "PC Instrument error #";TMPERR;" detected at line ";ERL
890 PRINT "Error: ";PCIB.ERR$
900 STOP

910 ' COMMON declarations are needed if your program is going to chain
to other programs. When chaining, be sure to call DEF.ERR as
well upon entering the chained-to program
920 COMMON PCIB.DIR$,PCIB.SEG
930 COMMON LD.FILE,GET.MEM,PANELS,DEF.ERR
940 COMMON PCIB.BASERR,PCIB.ERR,PCIB.ERR$,PCIB.NAME$,PCIB.GLBERR
950 COMMON IOABORT,IOCLEAR,IOCONTROL,IOENTER,IOENTERA,
IOENTERS,IOEOL,IOGETTERM,IOLOCKOUT,IOLOCAL,IOMATCH,IOOU
TPUT,IOOUTPUTA,IOOUTPUTS,IOPOLL,IOPOOLL,IOPOOLLU,IOREMOTE,IOR
ESET,IOSEND,IOSPOLL,IOSTATUS,IOTIMEOUT,IOTRIGGER,IODMA
960 COMMON FALSE, TRUE, NOERR, EUNKNOWN, ESEL, ERANGE, ETIME,
ECTRL, EPASS, ENUM, EADDR
970 ' End Program Set-up
980 ' User program can begin anywhere past this point
990 SCREEN 9
1000 SS = 2: ss=1 S-Slots; ss=0 Trials/Resets; ss=2 Not Defined
1010 KEY OFF
1020 DIM BIT$(8,7)
1030 RESTORE 8690
1040 FOR R=0 TO 6
1050 FOR B=7 TO 0 STEP -1
1060 READ BIT$(R,B)
1070 NEXT B:NEXT R
1080 ISC=7
1090 'INPUT"Sub-address register device number (even Nos. 0 to 28) = ",SUBDEV
'PRINT"Sub-address register device number (even Nos. 0 to 28) = 2"
SUBDEV=2
SUBDEV=ISC*100+SUBDEV;DATDEV=SUBDEV+1
VALUE=2;CHANNEL=3;CONF=0;SRQT=1;MATCH$=CHR$(10);FLAG=0
'PRINT"Choose your test configuration TORUS or RING"
'INPUT "Press your selection code (exit=0, torus=1, ring=2) ";T4
'PRINT "Press your selection code (exit=0, torus=1, ring=2) ring!"
T4=2
IF T4=0 THEN GOTO 1870
'INPUT "When test board is configured press ";RET
'PRINT "When test board is configured press 
'INPUT "Do you want to print the trace text (yes=1)";T1
'PRINT "Do you want to print the trace text - NO!"
T1=0
IF T4=0 THEN
T4=2
T2=1
IF T2=2 THEN GOTO 1700
T3=0
T4=0
T2=1
T3=1
T4=0
'INPUT "What TIMEOUT do you want";TIMEOUT
TIMEOUT=1
CALL IOTIMEOUT(ISC,TIMEOUT)
IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
'PRINT "TIMEOUT = ";TIMEOUT;" seconds"
CALL IODMA(ISC,VALUE,CHANNEL)
IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
CALL IOMA(TCH(ISC,MATCH$,FLAG)
IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
CALL IOEOL(TCH(ISC,MATCH$,FLAG)
IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
INPUT "What is the first IC serial number (=or digits)";SN
PRINT "What is the first IC serial number (=or digits)"
SN=1
SN$="SN"+MID$(STR$(SN),2)
DIM RATE%(2,4,4,74)
DIM DO(64),DI(64),CAM(128)
GOTO 1700
STOP
CLOSE #1,#2
SN=SN+1
SN$="SN"+MID$(STR$(SN),2)
F1$="PHIC\"+SN$+".TST"
F2$="PHIC\"+SN$+".ERR"
IF T3=1 THEN OPEN "O",#1,F1$ ELSE OPEN "O",#1,"SCRN:";
IF T3=1 THEN OPEN "O",#2,F2$ ELSE OPEN "O",#2,F2$
1740 CALL IOABORT(ISC)
1750 IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
1760 CALL IOCLEAR(ISC)
1770 IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
1780 CALL IOPPOLLC(SUBDEV,CONF)
1790 IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
1800 ' 
1810 LET RTA=16 : RTB=16 : AUX.A=0 : AUX.B=0 : R=0
1820 'PRINT #1,"IC Serial No. = ";SN
1830 IF T4=1 THEN GOTO 1850
1840 GOTO 1860
1850 INPUT "Switch SW4 on both boards to position 2 (OFF) press ";RET
1860 GOTO 4670 'IF T2=1 THEN GOSUB 9000 ELSE GOTO 5000
1870 STOP
1880 'Sub-routine to check and report serial poll requests
1890 '
1900 CALL IOSTATUS(ISC,SRQT,SRQ)
1910 IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
1920 IF SRQ=0 GOTO 2030
1930 CALL IOSPOLL(SUBDEV,SPR)
1940 IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
1950 IF SPR &H80 GOTO 2030
1960 A=&H40:MAXLEN=1
1970 PRINT #1,"Serial poll requested by the Orwell test board"
1980 GOSUB 2290
1990 PRINT #1,"Node A Reg 0 Lo = ";HEX$(DI(I));" Hex."
2000 'A=&H80
2010 'GOSUB 2200
2020 'PRINT #1,"Node B Reg 0 Lo = ";HEX$(DI(I));" Hex."
2030 RETURN
2040 ' 
2050 STOP
2060 'Sub-routine to set the test board sub-address
2070 ' 
2080 A$=CHR$(A)
2090 LONG=1
2100 CALL IOOUTPUTS(SUBDEV,A$,LONG)
2110 IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
2120 RETURN
2130 ' 
2140 STOP
2150 'Sub-routine to write data to the test board
2160 'at the current Sub-address = A.
2170 GOSUB 2060
2180 D$="":IF A=1 THEN AUX.A=DO(1)
2190 IF A=3 THEN AUX.B=DO(1)
2200 FOR I=1 TO LENGTH
2210 D$=D$ + CHR$(DO(I))
2220 NEXT I
2230 CALL I0OUTPUTS(DATDEV,D$,LENGTH)
2240 IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
2250 IF T1=1 THEN PRINT #1,"Have WRITTEN data to sub-address ";HEX$(A);" Hex."
2260 RETURN
2270 '  
2280 STOP
2290 'Sub-routine to read data from test board
2300 'at the current Sub-address A
2310 GOSUB 2060
2320 ACTLEN=0
2330 D$=SPACE$(MAXLEN)
2340 CALL IOENTERS(DATDEV,D$,MAXLEN,ACTLEN)
2350 IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
2360 FOR I=1 TO ACTLEN
2370 X$=MID$(D$,I,1)
2380 DI(I)=ASC(X$)
2390 NEXT I
2400 IF T1=1 THEN PRINT #1,"Have READ data from sub-address ";HEX$(A);" Hex."
2410 RETURN
2420 '  
2430 STOP
2440 'Sub-routine to load the TRAFFIC fifo
2450 'Restore needed to set the DATA pointer, Set the high byte sub-address
2460 K=0
2470 LENGTH=55
2480 FOR J=1 TO LENGTH
2490 READ DO(J)
2500 NEXT J
2510 GOSUB 2150
2520 IF K=1 GOTO 2550
2530 K=1:A=A-32 'Set sub-address for low byte of TRAFFIC fifo
2540 GOTO 2480
2550 IF T1=1 THEN PRINT #1,"Have loaded TRAFFIC fifo @ address ";HEX$(A);" Hex."
2560 A=&H4A : GOSUB 2060
2570 RETURN
2580 '  
2590 STOP
2600 'Sub-routine to read the CAMERA fifo
2610 'Set the high byte sub-address
2620 '  
2630 K=0:MAXLEN=64:LENGTH=1:TMP=A:A=&H1:AUX.A=AUX.A AND 
&HBF:DO(1)=AUX.A
2640 GOSUB 2150 'Disable CAMERA
2650 A=TMP

- 138 -
2660 GOSUB 2290
2670 FOR J=1 TO MAXLEN
2680 IF K=0 THEN CAM(J*2-1)=DI(J) ELSE CAM(J*2)=DI(J)
2690 NEXT J
2700 IF K=1 GOTO 2730
2710 K=1 : A=A-32 'Set sub-address for low byte of CAMERA fifo
2720 GOTO 2660
2730 'print
2740 PRINT
2750 FOR 1=0 TO 4
2760 IF CAM(I*22 + 1) &H96 THEN GOTO 2820
2770 IF (CAM(I*22+2) AND &HF0) = &HC0 THEN PRINT TAB(0);"S-slot";
2780 IF (CAM(I*22+2) AND &HF0) = &H0 THEN PRINT TAB(0);"Trial0";
2790 IF (CAM(I*22+2) AND &HF0) = &H10 THEN PRINT TAB(0);"Trial1";
2800 IF (CAM(I*22+2) AND &HF0) = &H40 THEN PRINT TAB(0);"Reset";
2810 IF (CAM(I*22+2) AND &HF0) = &H80 THEN PRINT TAB(0);"Data";
2815 IF (CAM(I*22+2) AND &HF0) = &H90 THEN PRINT TAB(0);"Data(B)";
2820 PRINT TAB(9);HEX$(CAM(I*22 + 1));
2830 PRINT TAB(14);HEX$(INT(CAM(I*22 + 2)/16));
2840 PRINT TAB(18);HEX$((CAM(I*22+4) AND &HF0)/16 + ((CAM(I*22+3)*16))
2850 PRINT TAB(26);HEX$((CAM(I*22+4) AND &HF));
2860 PRINT TAB(30);HEX$(CAM(I*22+6) + CAM(I*22+5)*256);
2870 PRINT TAB(36);
2880 FOR J = 7 TO 22
2890 IF CAM(I*22+J)&H10 THEN PRINT "0";HEX$(CAM(I*22+J));
2900 IF CAM(I*22+J)&HF THEN PRINT HEX$(CAM(I*22+J));
2910 NEXT J
2920 PRINT TAB(71);"Slot";J+1
2930 NEXT I
2940 RETURN
2950 ' 
2960 STOP
2970 'Sub-routine to load both MAP memories on both boards
2980 GOTO 3260
2990 'and check the contents by reading it back.
3000 'First disable the MAEN bits
3010 A=&H61 : LENGTH=1 : MAXLEN=1 : GOSUB 2290
3020 DO(1)=DI(1) AND &HEF : GOSUB 2150
3030 A=&HA1 : LENGTH=1 : MAXLEN=1 : GOSUB 2290
3040 DO(1)=DI(1) AND &HEF : GOSUB 2150
3050 A=&H50: 'Set MAP2 base address for board A
3060 LENGTH=1:MAXLEN=1:E=0
3070 RESTORE 3330
3080 FOR J=1 TO 8
3090 READ DO(1)
3100 GOSUB 2150
3110 GOSUB 2290

- 139 -
3120 IF DO(1)DI(1) THEN E=1
3130 A=A+1
3140 NEXT J
3150 IF A&H70 THEN GOTO 3170 ELSE A=&H70 'Set MAP1 base address for board A
3160 GOTO 3080
3170 IF A&H90 THEN GOTO 3200 ELSE A=&H90 'Set MAP2 base address for board B
3180 RESTORE 3330
3190 GOTO 3080
3200 IF A&HB0 THEN GOTO 3220 ELSE A=&HBO 'Set MAP1 base address for board B
3210 GOTO 3080
3220 IF E=1 THEN GOTO 3250
3230 IF T1=1 THEN PRINT #1,"MAP1 and MAP2 on both boards have been loaded"
3240 GOTO 3260
3250 PRINT #2,"Read/Write ERROR while loading MAP" : GOTO 3360
3260 'Finally enable the MAEN bits
3270 A=&H61 : LENGTH=1 : MAXLEN=1 : GOSUB 2290
3280 DO(1)=DI(1) OR &H10 : GOSUB 2150
3290 'A=&HA1 : LENGTH=1 : MAXLEN=1 : GOSUB 2200
3300 'DO(1)=DI(1) OR &H10 : GOSUB 2100
3310 GOTO 3360
3320 'Map2--DA=02AO + NA(3:0) Map1--DA=FC00
3330 DATA &HOO,&HOO,&HOO,&HOO,&HOO,&H04,&HOO,&HOO
3340 DATA &H8O
3350 RETURN
3360 RETURN
3370 'Sub-routine to read a single received slot when a PPL response=TRUE
3380 'Set the high byte sub-address
3390 TIM=TIMER+TIMEOUT+1
3400 WHILE TIM>TIMEOUT
3410 CALL IOPPOLL(ISC,PPR)
3420 IF PCRn.ERR NOERR THEN ERROR PCRn.BASERR
3430 L--0
3440 IF PPR AND 1 = 0 GOTO 3550
3450 K=0:MAXLEN=11
3460 GOSUB 2290
3470 FOR J=1 TO MAXLEN
3480 IF K=0 THEN CAM(J*2-1)=DI(J) ELSE CAM(J*2)=DI(J)
3490 NEXT J
3500 IF K=1 THEN GOTO 3540
3510 K=1:A=0 'Set sub-address for low byte of CAMERA fifo
3520 GOTO 3470
3530 TIM=TIMEOUT-1
3540 IF L=0 THEN PRINT #2,"Node reception TIMEOUT ERROR"
3570 IF L=0 GOTO 3620
3580 FOR J=1 TO 22
3590 PRINT #1, USING "\n"; HEX$(CAM(J));
3600 NEXT J
3610 PRINT #1, "Received Slot"
3620 RETURN
3630 'STOP
3640 'Sub-routine to read the node status and state registers
3650 'and to print the results to FILE #1
3660 'MAXLEN=1:AD=&H60:A=AD:X=0
3670 STREG=0
3680 PRINT: PRINT: PRINT
3690 PRINT "bit status"
3696 PRINT "Register 7 6 5 4 3 2 1 0"
3697 PRINT
3700 FOR J=1 TO 10
3710 FOR K=1 TO 2
3720 IF A &H7F GOTO 3850
3730 X=A AND &H2F: IF X=34 GOTO 3800 ELSE GOSUB 2290
3740 IF X=2 THEN GOSUB 3860
3750 PRINT #1, "Reg. \n"; HEX$(A); "(H) contains \n";
3760 PRINT #1, USING "\n"; HEX$(DI(1));
3770 PRINT #1, "(H) \n";
3780 GOSUB 7930: ' print reg bit names
3790 STREG=STREG+1: IF STREG=9 THEN STREG=0
3800 A=A-32
3810 NEXT K
3820 A=A+32
3830 IF J4 THEN A=AD+59+J ELSE A=AD+J
3840 NEXT J
3850 RETURN
3860 RT=16/(2^(DI(1) AND &HE0)/32))
3870 IF J THEN RTA=RT ELSE RTB=RT
3880 RETURN
3890 STOP
3900 'Sub-routine to read the various rates
3910 'and print the results to FILE #1
3920 'Set the reset timer scaling parameters RTA and RTB
3930 PRINT #1, "RATE METER MEASUREMENTS"
3940 LENGTH=1:MAXLEN=1:A=&H6E
3950 FOR J=1 TO 2
3960 FOR K=1 TO 2
3970 GOSUB 2290
3980 IF K=1 THEN RR=DI(1)*256 ELSE RR=RR+DI(1)
3990 A=A-32
4000 NEXT K
4010 IF J=1 THEN RATE%(1,4,R,NXTD)=RR*RTA ELSE
RATE%(2,4,R,NXTD)=RR*RTB
4020 IF J=1 THEN A=&HAE
4030 NEXT J
4040 PRINT #1,"RR.A=";RATE%(1,4,R,NXTD),"RR.B=";RATE%(2,4,R,NXTD)
4050 A=&H1 : TB=0 : LP=0 : TMP.A=AUX.A : TMP.B=AUX.B
4060 FOR J=1 TO 2
4070 FOR K=0 TO 3
4080 IF J=1 THEN DO(1)=(AUX.A AND &HFO)+K ELSE DO(1)=(AUX.B AND &HFO)+K
4090 TIM=O : GOSUB 2150
4100 WHILE TIMd
4110 TIM=TIM+1
4120 WEND
4130 GOSUB 2290
4140 IF LP=1 GOTO 4200
4150 IF DI(1)) THEN TB=4 ELSE GOTO 4200
4160 IF DI(1)2 THEN TB=8
4170 IF DI(1)
4180 IF J=1 THEN DO(1)=(AUX.A AND &HFO)+TB+K ELSE DO(1)=(AUX.B AND &HFO)+TB+K
4190 TBF=1 : LP=LP+1 : IF LP THEN GOTO 4090
4200 IF TB=0 THEN TBF=10
4210 IF TB=4 THEN TBF=5
4220 IF TB=8 THEN TBF=2
4230 PRINT #1,"Rate/2 milli-sec for SEL ";K," on node ";J," = ";DI(l)*TBF
4240 TB=O : LP=O : RATE%(J,K,R,NXTD)=DI(I)*TBF
4250 NEXT K
4260 IF J=1 THEN A=&H3
4270 NEXT J
4280 A=&H1:DO(1)=TMP.A
4290 GOSUB 2150 'Return AUXCTL’s to the previous values
4300 A=&H3:DO(1)=TMP.B
4310 GOSUB 2150
4320 RETURN
4330 STOP
4340 'Sub-routine to use the CAMERA triggered by GET
4350 A=&H1:LENGTH=1:AUX.A=AUX.A AND &HBF:DO(1)=AUX.A:GOSUB 2150
4360 DO(1)=AUX.A+&H40:GOSUB 2150
4370 CALL IOTRIGGER(SUBDEV)
4380 IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
4390 TIM=TIMER+1
4400 WHILE TIMd
4410 WEND
4420 A=&H20
4430 GOSUB 2600
4440 RETURN
4450 ' Sub-routine to perform a re-synchronisation
4460 STOP
4470 AD=A:LENGTH=I:MAXLEN=1
4480 GOSUB 2290 ' Fetch old control value
4490 TMP=DI(1):DO(I)=(TMP AND &HFC) OR &H1
4500 GOSUB 2150 ' Set SYNCH state
4510 TIM=TIMER+1
4530 WHILE TIM<TIMER
4540 TIM=TIMER+1
4550 GOSUB 4340 ' Check the T-slots using the CAMERA
4560 A=&H1:LENGTH=1:AUX.A=AUX.A OR &H40:DO(I)=AUX.A
4570 GOSUB 2150 ' Set the CAMEN bit
4580 A=AD:DO(I)=TMP
4590 GOSUB 2150 ' Return to COMMS and trigger the CAMERA
4600 TIM=TIMER+1
4610 WHILE TIM<TIMER
4620 WEND
4630 A=&H20
4640 GOSUB 2600 ' Check start sequence
4650 RETURN
4660 '
4670 ' Main ORWELL PH test programme
4680 ' 
4690 ' 
4700 ' 
4710 ' 
4720 ' 
4730 ' 
4740 ' Set AUXCTL registers to zero
4750 A=&H9 : LENGTH=1 : MAXLEN=1
4760 DO(1)=AUX.A
4770 GOSUB 2150
4780 ' STOP
4790 A=&H3
4800 GOSUB 2150
4810 ' 
4820 ' Write control registers of both Nodes and check by reading back
4830 A=&H41:DO(1)=&H36 ' Node A as Monitor set EAR & ELECTION state
4840 GOSUB 2150
4850 GOSUB 2290
4860 IF DI(1):DO(1) THEN PRINT #2,"Node A Reg 1 Lo R/W ERROR"
4870 A=&H61:DO(1)=&H90 ' RCR & MAEN
4880 GOSUB 2150
4890 GOSUB 2290
4900 IF DI(1):DO(1) THEN PRINT #2,"Node A Reg 1 Hi R/W ERROR"
4910 ' A=&H81:DO(1)=&H84 ' Node B Monitor capable node set EAR & COMMS state
4920 'GOSUB 2100
4930 'GOSUB 2200
4940 'IF DI(1)DO(1) THEN PRINT #2,"Node B Reg 1 Lo R/W ERROR"
4950 'A=&HA1:IF T4=1 THEN DO(1)=&H91 ELSE DO(1)=&H90 'RCR MAEN & SPD
4960 'GOSUB 2100
4970 'GOSUB 2200
4980 'IF DI(1)DO(1) THEN PRINT #2,"Node B Reg 1 Hi R/W ERROR"
4990 A=&H42:D0(1)=&H0 'Minimum reset times
5000 GOSUB 2150
5010 GOSUB 2290
5020 IF DI(1)DO(1) THEN PRINT #2,"Node A Reg 2 Lo R/W ERROR"
5030 'A=&H82
5040 'GOSUB 2100
5050 'GOSUB 2200
5060 'IF DI(1)DO(1) THEN PRINT #2,"Node B Reg 2 Lo R/W ERROR"
5070 A=&HE4:DO(1)=&H0
5080 GOSUB 2150
5090 A=&HC4:NXTD=&H1:DO(1)=NXTD 'Next-d=00 01
5100 GOSUB 2150
5110 A=&H44
5120 GOSUB 2290
5130 IF DI(1)DO(1) THEN PRINT #2,"Node A Reg 4 Lo R/W ERROR"
5140 'A=&H84
5150 'GOSUB 2200
5160 'IF DI(1)DO(1) THEN PRINT #2,"Node B Reg 4 Lo R/W ERROR"
5170 A=&H46:DO(1)=&HAA 'Set Node address (low) = aa aa for node A
5180 GOSUB 2150
5190 GOSUB 2290
5200 IF DI(1)DO(1) THEN PRINT #2,"Node A Reg 6 Lo R/W ERROR"
5210 A=&H66:DO(1)=&HAA 'Set Node address (high)= aa aa for node A
5220 GOSUB 2150
5230 GOSUB 2290
5240 IF DI(1)DO(1) THEN PRINT #2,"Node A Reg 6 Hi R/W ERROR"
5250 'A=&H86:DO(1)=&HBB 'Set Node address (low) = bb bb for node B
5260 'IF T4=1 THEN DO(1)=&HAA 'Make node B address = node A address TORUS node
5270 'GOSUB 2100
5280 'GOSUB 2200
5290 'IF DI(1)DO(1) THEN PRINT #2,"Node B Reg 6 Lo R/W ERROR"
5300 'A=&HA6:DO(1)=&HBB 'Set Node address (high)= bb bb for node B
5310 'IF T4=1 THEN DO(1)=&HAA 'Make node B address = node A address TORUS node
5320 'GOSUB 2100
5330 'GOSUB 2200
5340 'IF DI(1)DO(1) THEN PRINT #2,"Node B Reg 6 Hi R/W ERROR"
5350 'A=&HE6:DO(1)=&H0
5360 'GOSUB 2100
5370 A=&HE8:DO(1)=&H0  'Set slot count Hi-byte = 0
5380 GOSUB 2150
5390 A=&HC8:DO(1)=&HFF  'Set slot count Lo-byte = FF
5400 GOSUB 2150
5410  "IF T4=1 THEN GOTO 5650 ELSE INPUT "Switch SW4 on board A to position 1 (ON) then press ";RET"
5420  "Write the MAPs"
5430 GOSUB 2970
5440  "Write the TRAFFIC fifo"
5450 CALL IOCLEAR(SUBDEV)
5460 IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
5470 IF T4=1 THEN RESTORE 8430 ELSE RESTORE 8170
5480 A=&H20
5490 GOSUB 2440
5500 A=&H22
5510 GOSUB 2440
5520 IF T2=1 THEN GOSUB 6730
5530  "Use CAMERA to check S-slot generation"
5540 GOSUB 4340
5550 IF T2=1 THEN GOSUB 6730
5560 IF CAM(2)+CAM(3)+CAM(4)=220 GOTO 5580
5570 PRINT #2,"Incorrect S-slot in CAMERA"
5580  
5590  "Use CAMERA to check the START sequence"
5600 A=&H1:LENGTH=1:DO(1)=0:GOSUB 2150  'Disable CAMEN
5610 DO(1)=&H40:GOSUB 2150  'Set CAMEN again
5620 A=&H41:IF T4=1 THEN DO(1)=&H37 ELSE DO(1)=&H34  'Node A set EAR & COMMS
5630 GOSUB 2150
5640 INPUT "PRESS ";RET
5650 TIM=TIMER+1
5660 WHILE TIM<TIMER
5670 WEND
5680 A=&H20
5690 GOSUB 2600
5700 IF T2=1 THEN GOSUB 6730
5710 GOSUB 3650  'Read status and state
5720 A=&H48:MAXLEN=1
5730 GOSUB 2290  'Read slot count from Monitor node A
5740 PRINT "AT 5935"
5750 SLTCTN=DI(1):DO(1)=SLTCTN:LENGTH=1:A=&H88
5760 PRINT #1,"SLOT COUNT = ";SLTCTN:GOSUB 2150  'Write SLTCTN to node B
5770 IF T2=1 THEN GOSUB 6730
5780 R=0:GOSUB 3900  'Read rates
5790 IF T2=1 THEN GOSUB 6730
5800 IF T4=1 THEN GOTO 5960
5810  'Send the five slot types from Node B to Node A in single slot mode
5820 A=&H1:LENGTH=1:DO(1)=&H20 'Node A set to receive
5830 GOSUB 2150
5840 'A=&H3:DO(1)=&H90 'Node B set to transmit slots one at a time
5850 'GOSUB 2100
5860 FOR M=1 TO 5
5870 A=&H20
5880 GOSUB 3380 'Read single slot
5890 A=&H21:MAXLEN=1
5900 GOSUB 2290 'Read RTPA
5910 PRINT #1,"RTPA = ";HEX$(DI(1));" Hex"
5920 CALL IOTRIGGER(SUBDEV)
5930 IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
5940 NEXT M
5950 IF T2=1 THEN GOSUB 6730
5960 A=&H81:IF T4=1 THEN GOSUB 4470 'Do re-synchronisation of node B
5970 'Send high performance traffic
5980 GOSUB 3650 'Read status and state
5990 A=&H1:LENGTH=1:AUX.A=&H10 'Node A set for fast transmission
6000 DO(1)=AUX.A:GOSUB 2150
6010 'A=&H3:AUX.B=AUX.A 'Node B set for fast transmission
6020 'GOSUB 2100
6030 'GOSUB 3800 'Read rates
6040 'GOSUB 3600 'Read status and state
6050 IF T2=1 THEN GOSUB 6730
6060 '
6070 'Use CAMERA to check the RESET and TRIAL SLOTS
6080 GOSUB 4340
6090 IF T2=1 THEN GOSUB 6730
6100 IF T4=1 THEN NXTD=42 ELSE NXTD=21
6110 'Set the Next d_value on both Nodes
6120 DO(1)=NXTD:LENGTH=1:A=&HC4
6130 GOSUB 2150
6140 IF T2=1 THEN GOSUB 6730
6150 GOSUB 3900 'Read rates
6160 'Set the Next d_value on both nodes
6170 IF T4=1 THEN NXTD=46 ELSE NXTD=23
6180 DO(1)=NXTD:LENGTH=1:A=&HC4
6190 GOSUB 2150
6200 GOSUB 3900 'Read rates auto_resets should be present
6210 '
6220 'Disable auto-resets
6230 A=&H41:LENGTH=1:MAXLEN=1
6240 FOR J=1 TO 2
6250 GOSUB 2290
6260 DO(1)=DI(1) AND &HFB
6270 GOSUB 2150
6280 A=&H81
6290 NEXT J
6300 'Set the mask phase of node B to 1
6310 A=&H8D:LENGTH=1:DO(1)=&H40
6320 GOSUB 2150
6330 A=&H41 : GOSUB 4470 'Do re-synchronisation node A
6340 A=&H81 : IF T4=1 THEN GOSUB 4470 'Do re-synchronisation node B
6350 GOSUB 3900 'Read rates to check that transmission has re-established
6360 PRINT #1, "Check the overload rates for various Next-d and r values"
6370 FOR R=2 TO 4
6380 FOR L=18 TO 74 STEP 8
6390 A=&HED:DO(1)=R:LENGTH=1:MAXLEN =1
6400 GOSUB 2150
6410 A=&HC4:NXTD=L/T4:DO(1)=NXTD
6420 GOSUB 2150
6430 PRINT #1, "r-value = ", R, "Next-d = ", NXTD
6440 GOSUB 3900
6450 NEXT L
6460 NEXT R
6470 IF T4=1 THEN GOTO 6710
6480 PRINT #1, "Election test for various REF and address combinations"
6490 DO(1)=&H32:A=&HC1 'Set both nodes into ELECTION with REF
6500 FOR L=1 TO 3
6510 LENGTH=1
6520 GOSUB 2150
6530 GOSUB 3650 'Read status and state
6540 GOSUB 4340 'Check S-slot generation
6550 A=&H1:DO(1)=&H40:LENGTH=1 'Set CAMEN
6560 GOSUB 2150
6570 A=&HC1:DO(1)=&H30 'Return both nodes to COMMS state
6580 GOSUB 2150
6590 TIM=TIMER+1
6600 WHILE TIM<TIMER
6610 WEND
6620 A=&H20:GOSUB 2600 'Read START sequence in CAMERA
6630 GOSUB 3650 'Read status and state
6640 IF L=3 THEN A=&H88 ELSE A=&H48:MAXLEN=1
6650 GOSUB 2290 'Read the slot count
6660 IF L=1 THEN DO(1)=&HA ELSE DO(1)=&H22
6670 IF L=2 THEN A=&H81
6680 'Set both nodes into ELECTION without REF or AEP, EPADc=1
6690 'ELSE Set only node B into ELECTION with REF,AEP=0,EPADc=0
6700 NEXT L
6710 INPUT "Do you want to test another IC (yes=1)"; T5
6720 IF T5=1 THEN PRINT "Switch OFF test board PSU, change IC, switch PSU ON again, then type CONT ":GOTO 1660 ELSE GOTO 1870
6722 'One second delay routine
6723 T = TIMER
6724 WHILE TIMER  T + .6
6725 WEND
6726 RETURN
6730 GOSUB 6735:GOTO 7060
6735 PRINT TAB(24);"--- Select Function Key ---":PRINT
6740 PRINT:PRINT:PRINT TAB(25);"- F1 - Initiate S-Slots"
6750 PRINT:PRINT TAB(25);"- F3 - Initiate Trials and Resets"
6760 PRINT:PRINT TAB(25);"- F5 - Snapshot Camera"
6770 PRINT:PRINT:PRINT TAB(25);"- F7 - Examine Registers"
6780 PRINT:PRINT:PRINT:PRINT TAB(25);"- F10 - EXIT System"
6820 LINE (140,54)-(165,70),,B
6830 LINE (140,83)-(165,99),,B
6850 LINE (140,111)-(165,127),,B
6870 LINE (140,140)-(165,156),,B
6890 LINE (140,194)-(165,210),,B
6892 IF SS = 1 THEN LINE (140,54)-(165,70),,BF
6894 IF SS = 0 THEN LINE (140,83)-(165,99),,BF
6900 'Menu listing
6910 KEY 1,"S-slots"
6920 KEY 2,""
6930 KEY 3,"Trials"
6940 KEY 4,""
6950 KEY 5,"Camera"
6960 KEY 6,""
6970 KEY 7,"Diagnost"
6980 KEY 8,""
6990 KEY 9,""
7000 KEY 10,"EXIT"
7010 KEY (1) ON
7020 KEY (3) ON
7030 KEY (5) ON
7040 KEY (7) ON
7050 KEY (10) ON
7055 RETURN
7060 LOOP=1 : T5=0
7070 ON KEY (1) GOSUB 7151
7080 ON KEY (3) GOSUB 7160
7090 ON KEY (5) GOSUB 7170
7100 ON KEY (7) GOSUB 7141
7110 ON KEY (10) GOSUB 7990
7120 WHILE LOOP=1
7130 WEND
7140 RETURN
7141 KEY (1) OFF:KEY (3) OFF:KEY (5) OFF:KEY (7) OFF:KEY (10) OFF
7142 LINE (140,140)-(165,156),,BF:BEEP:GOSUB 6722:CLS
7143 PRINT:PRINT TAB(24);"Hit any key to return to menu"
7144 LINE (140,12)-(165,28),,B:GOSUB 3650
7145 WHILE Z$ = ""
7146 Z$ = INKEY$:WEND
7147 BEEP:LINE (140,12)-(165,28),,BF:GOSUB 6722:Z$ = INKEY$:WHILE Z$ ""
7148 Z$ = INKEY$:WEND
7149 KEY (1) ON:KEY (3) ON:KEY (5) ON:KEY (7) ON: KEY (10) ON
7150 CLS:GOSUB 6735:RETURN
7151 BEEP:A$ = "41":DOS$ = "36":GOSUB 7330
7152 LINE (140,54)-(165,70),BF
7154 LINE (141,84)-(164,98),0,BF
7155 SS = 1
7156 RETURN
7160 BEEP:A$ = "41":DOS$ = "34":GOSUB 7330
7162 LINE (140,83)-(165,99),BF
7164 LINE (141,55)-(164,69),0,BF
7165 SS = 0
7166 RETURN
7170 LINE (140,111)-(165,127),BF:BEEP:GOSUB 6722:CLS
7180 KEY (1) OFF:KEY (3) OFF:KEY (5) OFF:KEY (7) OFF:KEY (10) OFF
7182 PRINT:PRINT TAB(24);"Hit any key to return to menu"
7186 LINE (140,12)-(165,28)"B
7195 PRINT:PRINT:PRINT "Type JIK Cl O/addr C2 S/addr Data field
    Sequence"
7196 VIEW PRINT 7 TO 24
7200 Z$ = ""
7205 WHILE Z$ = ""
7233 LINE (140,12)-(165,28),BF:BEEP:GOSUB 6722
7234 KEY (1) ON:KEY (3) ON:KEY (5) ON:KEY (7) ON: KEY (10) ON
7235 VIEW PRINT 1 TO 24:CLS
7236 GOSUB 6735:Z$ = INKEY$
7237 WHILE Z$ ""
7238 Z$ = INKEY$:WEND
7239 RETURN
7240 INPUT "Input the register address to be read";A$
7250 'PUT "Input the number of bytes to be read";MAXLEN
7260 GOSUB 7420
7270 MAXLEN=1
7280 GOSUB 2290
7290 FOR I=1 TO MAXLEN
7300 PRINT #1,"D$(";I;""") = ";HEX$(DI(I));" Hex."
7310 NEXT I
7320 RETURN
7330 'Routine to program selected register
7340 'INPUT "Input the register address to be written";A$
7350 'PUT "Input the number of bytes to be written";LENGTH
7360 GOSUB 7420
7370 LENGTH=1
7380 FOR I=1 TO LENGTH
7390 'PRINT #1,"Type the value of byte No. ";I;"" ;INPUT DO$
7400 IF LEFT$(DO$,2)"&H" THEN DO(I)=VAL("&H"+DO$) ELSE DO(I)=VAL(DO$)
7410 NEXT I:GOSUB 2150:RETURN
7420 IF RIGHT$(A$,2)="LA" THEN A=&H40:GOTO 7480
7430 IF RIGHT$(A$,2)="HA" THEN A=&H60:GOTO 7480
7440 IF RIGHT$(A$,2)="LB" THEN A=&H80:GOTO 7480
7450 IF RIGHT$(A$,2)="HB" THEN A=&HA0:GOTO 7480
7460 IF LEFT$(A$,2)"&H" THEN A=VAL("&H"+A$) ELSE A=VAL(A$)
7470 RETURN
7480 A=A+VAL("&H"+LEFT$(A$,1))
7490 RETURN
7500 PRINT #1,"Reading the status and state registers on both nodes"
7510 GOSUB 3650
7520 RETURN
7530 '7540 LOOP=0
7550 RETURN
7560 PRINT #1,"Reading the various rates on both nodes"
7570 GOSUB 3900
7580 RETURN
7590 CALL IOCLEAR(SUBDEV)
7600 IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
7610 PRINT #1,"Have performed a device clear DCL"
7620 RETURN
7630 '7640 CALL IOTRIGGER(SUBDEV)
7650 IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
7660 'PRINT #1,"Have performed a group execute trigger GET"
7670 RETURN
7680 '7690 PRINT #1,"Loading both Traffic fifo's."
7700 CALL IOCLEAR(SUBDEV)
7710 IF PCIB.ERR NOERR THEN ERROR PCIB.BASERR
7720 IF T4=1 THEN RESTORE 8430 ELSE RESTORE 8170
7730 A=&H20:GOSUB 2440
7740 A=&H22:GOSUB 2440
7750 RETURN
7760 '7770 'PRINT #1,"Reading the camera fifo"
7780 A=&H20
7790 GOSUB 2600
7800 RETURN
7810 '7820 PRINT #1,"Reading a single received data slot"
7830 A=&H20
7840 GOSUB 3380
7850 RETURN
7860 ' setup camera

- 150 -
7870 A=&H1:DO(1)=AUX.A AND &HBF:LENGTH=1
7880 GOSUB 2150
7890 A=&H1:DO(1)=AUX.A + &H40:LENGTH=1
7900 GOSUB 2150
7910 'PRINT "Camera initialised. Press F6 to trigger."
7920 RETURN
7930 ' print out bit names for status registers
7940 FOR B=7 TO 0 STEP -1
7950 IF DI(1) AND (2^B) THEN PRINT #1,TAB(29 + 6*(7-B));BITS(STREG,B);
7960 NEXT B
7970 PRINT #1,""
7980 RETURN
7990 'INPUT "Do you want to test another IC (yes=1)";T5
8000 LINE (140,194)-(165,210)"BF:BEEP:GOSUB 6722
8010 CLS
8020 'IF T51 THEN GOTO 9790
8030 'PRINT "Switch OFF test board PSU, change IC, switch PSU ON again, then type CONT "
8040 'GOTO 1300
8050 'PRINT "Have ended the programme run"
8060 KEY 1,"LIST"
8070 KEY 2,"RUN"
8080 KEY 3,"LOAD"
8090 KEY 4,"SAVE"
8100 KEY 5,"CONT"
8110 KEY 6,"LPT1:"
8120 KEY 7,"TRON"
8130 KEY 8,"TROFF"
8140 KEY 9,"KEY"
8150 KEY 10,"SCREEN 0,0,0"
8160 END
8170 'Traffic slots for a 2 node ring
8180 'Node A high byte data
8190 DATA &H96,&Hbb,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8200 DATA &H96,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8210 DATA &H96,&Haa,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8220 DATA &H96,&Haa,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07, &H08
8230 DATA &H96,&Hang,&H00,&H01,&H02,&H03, &H04,&H05,&H06, &H07, &H08
8240 'Node A low byte data addressed to B
8250 DATA &H8b,&Hb4,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
:'NAQD
8260 DATA &H8F,&H04,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
:'MAP1
8270 DATA &H8b,&H24,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
:'MAP2
8280 DATA &H9a,&Hac,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
:'BROADCAST
8290 DATA &H9a,&Ha8,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H58
  :P2ERR
8300 'Traffic slots for a 2 node ring
8310 'Node B high byte data addressed to A
8320 DATA &H96,&H00,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8330 DATA &H96,&H00,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8340 DATA &H96,&H00,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8350 DATA &H96,&H00,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8360 DATA &H96,&H00,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8370 'Node B low byte data addressed to A
8380 DATA &H80,&H1C,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
  :'NAQD
8390 DATA &H96,&H00,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
  :'MAP1
8400 DATA &H80,&H14,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
  :'MAP2
8410 DATA &H90,&H24,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
  :'BROADCAST
8420 DATA &H90,&H20,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
  :P2ERR
8430 'Traffic slots for a 1 node torus
8440 'Node A high byte data
8450 DATA &H96,&H00,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8460 DATA &H96,&H00,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8470 DATA &H96,&H2A,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8480 DATA &H96,&H00,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8490 DATA &H96,&H00,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8500 'Node A low byte data addressed to B
8510 DATA &H80,&H1C,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
  :'NAQD
8520 DATA &H80,&H14,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
  :'MAP1
8530 DATA &H80,&H14,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
  :'MAP2
8540 DATA &H90,&H24,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
  :'BROADCAST
8550 DATA &H90,&H20,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
  :P2ERR
8560 'Traffic slots for a 1 node torus
8570 'Node B high byte data addressed to A
8580 DATA &H96,&H00,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8590 DATA &H96,&H00,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8600 DATA &H96,&H2A,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8610 DATA &H96,&H00,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8620 DATA &H96,&H00,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
8630 'Node B low byte data addressed to A
8640 DATA &H80,&H1C,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08
  :'NAQD
8650 DATA &H8F,&H04,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08 :'MAP1
8660 DATA &H80,&H04,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08 :'MAP2
8670 DATA &H9A,&HAC,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08 :'BROADCAST
8680 DATA &H99,&H98,&H00,&H01,&H02,&H03,&H04,&H05,&H06,&H07,&H08 :'P2ERR
8690 'data for status reg bits
8700 DATA " MOF "," MPE "," SERR "," P2ERR "," P1ERR "," 2M "," WIN "," MON 
8710 DATA " 0 "," 0 "," 0 "," RLU "," ARST "," QUIET "," SOV "," SPE 
8720 DATA " RCR "," ER "," PR "," MAEN "," SPD3 "," SPD2 "," SPD1 "," SPD0 
8730 DATA " MDIS "," MRO "," REF "," AEP "," EPAD2 "," EAR "," M2 "," M1 
8740 DATA " RT2 "," RT1 "," RT0 "," ART4 "," ART3 "," ART2 "," ART1 "," ART0 
8750 DATA " TC_S "," TC_T "," TCT "," TCD "," S4 "," S3 "," S2 "," S1 
8760 DATA " LOS2 "," LOS1 "," PFS "," PFT "," EPAD "," FIFO "," TCR "," ORF 
8770 END
APPENDIX II

Software packetiser interface routines

Software routines written in C language necessary for the operation of PC based network applications.

SEARCH

#include <stdio.h>
#include <dos.h>

/* This routine searches memory locations for the presence of any Orwell boards. A network interface board has CODE(Hex) as its identifier. A camera board has FOOD(Hex) as its identifier. All board locations are selectable by means of a couple of PCB jumpers. Locations 300H, 308H, 310H, and 318H are selectable. Running this routine allows inspection of memory from 300H through to 320H and should locate the appropriate board identifier(s). */

void main()
{

    int addr, value, d;
    printf("


"");

    for (addr = 768; addr = 800; addr = addr + 2)
    {
        printf("\nMemory address: %X \tValue: %X \n", addr, inport(addr));
        if (inport(addr) == 0xC0DE) printf(" (Orwell Interface)\n");
    }

}
if (inport(addr) == 0xF00D) printf(" (Camera Board)");
}

/* The functions ort and outport transfer data through the I/O space within the PC's memory. This space is memory-mapped starting at location 300H (768 decimal). The function ort above simply reads a data word from each location in the I/O space. If a board is present it will generate a valid identifier code. */

printf("\n\nEnd of run
\n\nHit any key and ");
scanf("%X",&d);
}

RESET

#include <stdio.h>
#include <dos.h>

main()
{
    int control;
    control = 0x0302;

    outport(control, 0x0000);
    outport(control, ((0x0050) | inport(control)));
    outport(control, ((0xFFAF) & inport(control)));
    outport(control, ((0x00A8) | inport(control)));
}
STATUS

#include <stdio.h>

main()
{
    int status, value;
    status = 0x0304;

    value = import(status);
    printf("\n");
    printf(" ADDRESS / VALUE : %4X %6X \n", status, value);
    printf("\n");

    printf("Tx Full = ");
    if (value & 128) printf("No\n"); else printf("Yes\n");

    printf("Tx Half = ");
    if (value & 64) printf("No\n"); else printf("Yes\n");

    printf("Tx Empty = ");
    if (value & 32) printf("No\n"); else printf("Yes\n");

    printf("\n");

    printf("Rx Full = ");
    if (value & 4) printf("No\n"); else printf("Yes\n");

    printf("Rx Half = ");
    if (value & 2) printf("No\n"); else printf("Yes\n");

    printf("Rx Empty = ");
    if (value & 1) printf("No\n"); else printf("Yes\n");
    }
printf("\n");
printf("Dval 0 = ");

if (value & 8) printf("True\n"); else printf("False\n");

printf("\n");
printf("Spare = ");
if (value & 16) printf("1\n"); else printf("0\n");

}

DVAL

#include <stdio.h>
#include <dos.h>

main()
{
    int dval_address, dvalue;
    dval_address = 0x0300;
    dvalue = 0x000A;
    outport(dval_address, dvalue);
}

SHOTS

#include <stdio.h>
#include <dos.h>

main()
# Appendix II

```c
int i, fifo, status, control, count;
int source_address, destination_address;

fifo = 0x306;
status = 0x304;
control = 0x302;

source_address = 0x1234;
destination_address = 0xC0C1;

/* This routine format data into slot and transfer 16 slots into the transmit fifos. */

/* Reset Tx FIFOs */

outport(control, (0x0010 | inport(control)));
outport(control, (0xFFEF & inport(control)));

/* Enable FIFOs */

outport(control, (0x0020 | inport(control)));

for (count = 1; count = 16; count++)
{
    /* Write the HEADER */

    outport(fifo,(0x9680 + (((destination_address) & 0xF000) / (16*16*16)));
    outport(fifo,(0x0002 + (((destination_address) & 0xFFF) * 16)));
    outport(fifo,source_address);

    // 158
```
/* Write the DATA */

outport(fifo,count);
outport(fifo,0x1111);
outport(fifo,0x2222);
outport(fifo,0x3333);
outport(fifo,0x4444);
outport(fifo,0x5555);
outport(fifo,0x6666);
outport(fifo,0x7777);

} /* Write the padding words */

for (i = 0; i = 244; i++)

outport(fifo,0xDDDD);

/* Assert PHIC TRQ */

outport(control, (0x0001 | inport(control)));

/* Wait for empty fifo */

do { } while ((inport(status) & 64) == 0);

/* Release PHIC TRQ */

outport(control, (0xFFFF & inport(control)));
}
#include <stdio.h>
#include <dos.h>

main()
{
    int cell_count, count, status, fifo;

    status = 0x0304;
    fifo = 0x306;
    cell_count = 1;

    printf("\n");
    if ((inport(status) & 0x0001) == 0)
        printf("No data in Rx fifo\n");
    else {
        do {
            printf("cell %2d", cell_count);
            for (count = 0; count = 8; count++)
                printf("%6X", inport(fifo));
            printf("\n");
            cell_count++;

        } while ((inport(status) & 1) == 1);
    printf("\nTotal cell(s) received = %2d", cell_count - 1);
    printf("\nRx fifo now empty\n");
    }
}
TRAFFIC

#include <stdio.h>
#include <dos.h>

main()
{

int count, j, i, z, cell_count;
int status, fifo, control, source_address, destination_address;
long txccell, y;

status = 0x0304;
fifo = 0x0306;
control = 0x0302;

source_address = 0xC0C1;
destination_address = 0xC0C1;

printf("\n1000 Blocks of 20 slots being transmitted...");
printf("\n");

for (y = 0; y = 9; y++)
{
    for (z = 0; z = 99; z++)
    {
        /* Reset FIFOs */
        outport(control, (0x0010 l inport(control)));
        outport(control, (0xFFEF & inport(control)));

        /* Enable FIFOs */
    }
}
outport(control, (0x0020 | inport(control)));
for (j = 0; j = 19; j++)
{

    /* Write the HEADER */

    outport(fifo, 0x9680 + ((destination_address & 0xFOOO)/(16*16*16)));
    outport(fifo, 0x0002 + ((destination_address & 0x0FFF) * 16));
    outport(fifo, source_address);

    /* Write the DATA */

    outport(fifo, 256*0);
    outport(fifo, 256*1);
    outport(fifo, 256*2);
    outport(fifo, 256*3);
    outport(fifo, 256*4);
    outport(fifo, 256*5);
    outport(fifo, 256*6);
    outport(fifo, 256*7);
}

    /* Write the Padding words */

    for (i = 0; i = 244; i++)
    outport(fifo, 0xDDDD);

    /* Assert PHIC TRQ */

    outport(control, (0x0001 | inport(control)));

    /* Wait for Tx Half Full Flag */
do { } while ((inport(status) & 64) == 0);
/* Release PHIC TRQ */
outport(control, (0xFFFE & inport(control)));
}
}
printf("\nTx finish");
printf("\n");
}

LISTEN

#include <stdio.h>
#include <dos.h>

/* This routine will "listen" to any data received from the Orwell ring.
Any data that arrives will be displayed on the PC screen in the same
slot structure.
*/

main()
{
    int scrap, count, status, fifo;
    long cell_count;

    status      = 0x0304;
    fifo        = 0x306;
    cell_count  = 1;

    /* The variables status and fifo point to I/O locations in memory.
    status is a control word generated by the Orwell network card
    and indicates the status of the network interface. fifo is the
    memory location in which data arriving from the network is stored.*/
Both these variables assume that the interface card is set to memory location 300H.

*/

printf("\n\n");

printf("Emptying Rx fifo\n");
do { scrap = inport(fifo); } while ((inport(status) & 1) == 1);

/* The above line reads data from the receive fifo (i.e., receive data buffer) until there is no data left (i.e., it empties it). The line (inport(status) & 1) == 1 is a flag indicating if there is any more data in the receive fifo.
A better way to do this would be to use the control byte to reset the receive fifo (this could not be done originally).
*/

printf("Listening...\n\n");

do {
    do {} while ((inport(status) & 1) == 0); /* Wait for data */
    printf("Cell: %4d", cell_count);
    printf("\n");
    for (count = 0; count = 8; count++) /* Read nine data */
    {
        /* words from I/F */
    do {} while ((inport(status) & 1) == 0); /* board and */
    printf("%6X", inport(fifo)); /* display on PC */
    } /* screen in slot */
    printf("\n"); /* format */
    cell_count++;

    } while (2 == 2); /* Repeat forever */
CAMERA

#include <stdio.h>
#include <dos.h>

/* This routine searches memory locations for the presence of an
Orwell camera board, then initiates the camera function. A camera
board has F00D(Hex) as its identifier. All board locations are
selectable by means of a couple of PCB jumpers. Locations 300H,
308H, 310H, and 318H are selectable so the code searches for the
appropriate location.

Documentation exists at the end of this file detailing status and
control bytes that affect the operation of the camera board.

*/

void main()
{
    int addr, value, key, base;
    int board_present, network_present, yes, no;
    int control, status, fifo;
    int control_byte, status_byte;
    int count, slot;
    int frame_error, slot_error, start_of_slot;

    int nibble[44];

    yes = 1;
    no = 0;
    board_present = 0;
    network_present = 0;
    base = 0;
    frame_error = 0;
slot_error = 0;
start_of_slot = 0;
printf("\n\n\n\n\n");
printf("Camera Board Control Software\n");
printf("-----------------------------\n");
printf("Searching for camera board ... ");

for (addr = 768; addr = 800; addr = addr+2)
{
    if (inport(addr) == 0xFOOD) board_present = yes;
    if (inport(addr) == 0xFOOD) base = addr;
}
if (board_present == yes)
{
    printf("Board located at address %X Hex.",base);
    control = base + 2;
    status = base + 4;
    fifo = base + 6;
    printf("\n\nSearching for network board ... ");
    status_byte = inport(status);
    if ((status_byte & 0x0004) == 0) network_present = yes;
    if (network_present == yes)
    {
        printf("Network board located.\n\n");
        printf("Initiating code ...\n\n");
        do
        {
            printf("\n\n");
            outport(control,0x0000); /* Initial control byte value */
            outport(control,0x0001); /* Reset camera FIFO */
        }
outport(control, 0x0000);

outport(control, 0x0008); /* Enable buffer driver */
outport(control, 0x000A); /* Enable FIFO read/write */
outport(control, 0x000E); /* Capture ring data */

do {} while ((inport(status) & 0x0001) 0);
/* Wait until FIFO is full */

outport(control, 0x000A); /* Terminate capture process */

/* Strip off leading part-slot up to start of new slot */
start_of_slot = no;
frame_error = no;
do {
    nibble[1] = ((inport(fifo) & 0x001F));
    if (nibble[1] == 0x0019) start_of_slot = yes;
    if ((inport(status) & 0x0002) == 0) frame_error = yes;
while ((start_of_slot == no) & (frame_error == no));

if (frame_error == yes) printf("Framing errors detected\n\n\n");
nibble[1] = (nibble[1] & 0x000F);

/* Read remainder of first slot */
for (count = 2; count = 44; count++)
    nibble[count] = (inport(fifo) & 0x000F);

for (slot = 1; slot = 10; slot++)
{
    printf("\x1B[37;40;m");
    if (nibble[3] == 0) printf("\x1B[37mTrial0 ");
    if (nibble[3] == 1) printf("\x1B[37mTrial1 ");
    if (nibble[3] == 4) printf("\x1B[32mReset ");

    printf("\x1B[37;40;0m");
if (nibble[3] == 8) printf("\x1B[35mData ");
if (nibble[3] == 9) printf("\x1B[35mData - B ");
if (nibble[3] == 12) printf("\x1B[36mS-slot ");

slot_error = no;
if (nibble[3] == 2) slot_error = yes;
if (nibble[3] == 3) slot_error = yes;
if (nibble[3] == 5) slot_error = yes;
if (nibble[3] == 6) slot_error = yes;
if (nibble[3] == 7) slot_error = yes;
if (nibble[3] == 10) slot_error = yes;
if (nibble[3] == 11) slot_error = yes;
if (nibble[3] == 13) slot_error = yes;
if (nibble[3] == 14) slot_error = yes;
if (nibble[3] == 15) slot_error = yes;

if (slot_error == yes) printf("\x1B[31mError ");

printf("%X", nibble[1]); /* Start of slot del (9) */
printf("%X", nibble[2]); /* Start of slot del (6) */

printf(" %X", nibble[3]); /* Control field C1 */

printf(" %X", nibble[4]); /* Destination address */
printf("%X", nibble[5]); /* Destination address */
printf("%X", nibble[6]); /* Destination address */
printf("%X", nibble[7]); /* Destination address */

printf(" %X", nibble[8]); /* Control field C2 */

printf(" %X", nibble[9]); /* Source address */
printf("%X", nibble[10]); /* Source address */
printf("%X", nibble[11]); /* Source address */
printf("%X ",nibble[12]);  /* Source address */

for (count = 13; count = 44; count = count + 2)
{
    printf("%X", nibble[count]);
    printf("%X ", nibble[count + 1]);  /* Data bytes */
    / * Read next slot */
    for (count = 1; count = 44; count++)
        nibble[count] = (inport(fifo) & 0x000F);
        printf("\n");
}
delay(1300);

} while (2 == 2);

else printf("No network board.\n\nTerminating ...\n");
}

else printf(" Board not located.\n\nTerminating ...\n"); /*scanf("%X",&key);*/
}
/*

Status Byte pin configuration
-----------------------------------------------

<table>
<thead>
<tr>
<th>lsb 0</th>
<th>value = 0</th>
<th>FIFO is full</th>
<th>value = 1</th>
<th>FIFO is not full</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FIFO empty</td>
<td></td>
<td>FIFO is not empty</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>network is present</td>
<td>network is not present</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>N/C</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>/</td>
<td>/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>/</td>
<td>/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>/</td>
<td>/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>msb 7</td>
<td>N/C</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Control Byte pin configuration
-----------------------------------------------

<table>
<thead>
<tr>
<th>lsb 0</th>
<th>value = 0</th>
<th>normal FIFO operation</th>
<th>value = 1</th>
<th>Reset FIFO</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FIFO disabled</td>
<td>FIFO enabled</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Hold data</td>
<td>Capture data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Disable FIFO driver</td>
<td>Enable FIFO driver</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>N/C</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>/</td>
<td>/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>/</td>
<td>/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>msb 7</td>
<td>N/C</td>
<td>N/C</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*/
Colour codes:

Foreground colours (add 10 to get background colour):

<table>
<thead>
<tr>
<th>Code</th>
<th>Colour</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>Black</td>
</tr>
<tr>
<td>31</td>
<td>Red</td>
</tr>
<tr>
<td>32</td>
<td>Green</td>
</tr>
<tr>
<td>33</td>
<td>Yellow</td>
</tr>
<tr>
<td>34</td>
<td>Blue</td>
</tr>
<tr>
<td>35</td>
<td>Magenta</td>
</tr>
<tr>
<td>36</td>
<td>Cyan</td>
</tr>
<tr>
<td>37</td>
<td>White</td>
</tr>
</tbody>
</table>

Attributes:

<table>
<thead>
<tr>
<th>Code</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>All attributes off</td>
</tr>
<tr>
<td>1</td>
<td>Bold On</td>
</tr>
<tr>
<td>4</td>
<td>Underscore (monochrome display only)</td>
</tr>
<tr>
<td>5</td>
<td>Blink on</td>
</tr>
<tr>
<td>7</td>
<td>Reverse video on</td>
</tr>
<tr>
<td>8</td>
<td>Concealed on</td>
</tr>
</tbody>
</table>

*/
APPENDIX III

Technical details of the PHIC device

4B/5B ENCODE/DECODE TABLE

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Tx Data</th>
<th>ENC Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0 0000</td>
<td>01110</td>
</tr>
<tr>
<td>1</td>
<td>0 0001</td>
<td>01001</td>
</tr>
<tr>
<td>2</td>
<td>0 0010</td>
<td>10101</td>
</tr>
<tr>
<td>3</td>
<td>0 0011</td>
<td>10011</td>
</tr>
<tr>
<td>4</td>
<td>0 0100</td>
<td>01010</td>
</tr>
<tr>
<td>5</td>
<td>0 0101</td>
<td>01011</td>
</tr>
<tr>
<td>6</td>
<td>0 0110</td>
<td>01101</td>
</tr>
<tr>
<td>7</td>
<td>0 0111</td>
<td>10111</td>
</tr>
<tr>
<td>8</td>
<td>0 1000</td>
<td>10010</td>
</tr>
<tr>
<td>9</td>
<td>0 1001</td>
<td>11001</td>
</tr>
<tr>
<td>A</td>
<td>0 1010</td>
<td>11010</td>
</tr>
<tr>
<td>B</td>
<td>0 1011</td>
<td>11011</td>
</tr>
<tr>
<td>C</td>
<td>0 1100</td>
<td>01111</td>
</tr>
<tr>
<td>D</td>
<td>0 1101</td>
<td>11101</td>
</tr>
<tr>
<td>E</td>
<td>0 1110</td>
<td>11110</td>
</tr>
<tr>
<td>F</td>
<td>0 1111</td>
<td>10110</td>
</tr>
<tr>
<td>I</td>
<td>1 0000</td>
<td>11111</td>
</tr>
<tr>
<td>J</td>
<td>1 1001</td>
<td>11000</td>
</tr>
<tr>
<td>K</td>
<td>1 0110</td>
<td>10001</td>
</tr>
<tr>
<td>Q</td>
<td>1 1111</td>
<td>00000</td>
</tr>
<tr>
<td>S</td>
<td>1 1100</td>
<td>10100</td>
</tr>
<tr>
<td>T</td>
<td>1 0011</td>
<td>00101</td>
</tr>
</tbody>
</table>
## MAP Programming Table

### MAP1

<table>
<thead>
<tr>
<th>Octal</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>10 01</td>
<td>11 01</td>
<td>12 01</td>
<td>13 01</td>
<td>14 01</td>
<td>15 01</td>
<td>16 01</td>
<td>17 01</td>
</tr>
<tr>
<td>1</td>
<td>10 02</td>
<td>11 02</td>
<td>12 02</td>
<td>13 02</td>
<td>14 02</td>
<td>15 02</td>
<td>16 02</td>
<td>17 02</td>
</tr>
<tr>
<td>2</td>
<td>10 04</td>
<td>11 04</td>
<td>12 04</td>
<td>13 04</td>
<td>14 04</td>
<td>15 04</td>
<td>16 04</td>
<td>17 04</td>
</tr>
<tr>
<td>3</td>
<td>10 08</td>
<td>11 08</td>
<td>12 08</td>
<td>13 08</td>
<td>14 08</td>
<td>15 08</td>
<td>16 08</td>
<td>17 08</td>
</tr>
<tr>
<td>4</td>
<td>10 10</td>
<td>11 10</td>
<td>12 10</td>
<td>13 10</td>
<td>14 10</td>
<td>15 10</td>
<td>16 10</td>
<td>17 10</td>
</tr>
<tr>
<td>5</td>
<td>10 20</td>
<td>11 20</td>
<td>12 20</td>
<td>13 20</td>
<td>14 20</td>
<td>15 20</td>
<td>16 20</td>
<td>17 20</td>
</tr>
<tr>
<td>6</td>
<td>10 40</td>
<td>11 40</td>
<td>12 40</td>
<td>13 40</td>
<td>14 40</td>
<td>15 40</td>
<td>16 40</td>
<td>17 40</td>
</tr>
<tr>
<td>7</td>
<td>10 80</td>
<td>11 80</td>
<td>12 80</td>
<td>13 80</td>
<td>14 80</td>
<td>15 80</td>
<td>16 80</td>
<td>17 80</td>
</tr>
</tbody>
</table>

### MAP2

<table>
<thead>
<tr>
<th>Octal</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>10 01</td>
<td>11 01</td>
<td>12 01</td>
<td>13 01</td>
<td>14 01</td>
<td>15 01</td>
<td>16 01</td>
<td>17 01</td>
</tr>
<tr>
<td>1</td>
<td>10 02</td>
<td>11 02</td>
<td>12 02</td>
<td>13 02</td>
<td>14 02</td>
<td>15 02</td>
<td>16 02</td>
<td>17 02</td>
</tr>
<tr>
<td>2</td>
<td>10 04</td>
<td>11 04</td>
<td>12 04</td>
<td>13 04</td>
<td>14 04</td>
<td>15 04</td>
<td>16 04</td>
<td>17 04</td>
</tr>
<tr>
<td>3</td>
<td>10 08</td>
<td>11 08</td>
<td>12 08</td>
<td>13 08</td>
<td>14 08</td>
<td>15 08</td>
<td>16 08</td>
<td>17 08</td>
</tr>
<tr>
<td>4</td>
<td>10 10</td>
<td>11 10</td>
<td>12 10</td>
<td>13 10</td>
<td>14 10</td>
<td>15 10</td>
<td>16 10</td>
<td>17 10</td>
</tr>
<tr>
<td>5</td>
<td>10 20</td>
<td>11 20</td>
<td>12 20</td>
<td>13 20</td>
<td>14 20</td>
<td>15 20</td>
<td>16 20</td>
<td>17 20</td>
</tr>
<tr>
<td>6</td>
<td>10 40</td>
<td>11 40</td>
<td>12 40</td>
<td>13 40</td>
<td>14 40</td>
<td>15 40</td>
<td>16 40</td>
<td>17 40</td>
</tr>
<tr>
<td>7</td>
<td>10 80</td>
<td>11 80</td>
<td>12 80</td>
<td>13 80</td>
<td>14 80</td>
<td>15 80</td>
<td>16 80</td>
<td>17 80</td>
</tr>
</tbody>
</table>

The first column represents addresses and the second column represents data.
3.4.8 POST GATES (all 3 sheets)

Some of the outputs are not taken directly from registers but are produced by extra combinatorial logic gates. These signals are: EL (sheet 1 U16) which controls the NA circuits during the ELECTION state to store the destination address of each input slot for comparison with the next slot to ensure that the election has genuinely been lost; and some of the signals contained in bus NSROU(7:0). This bus is the set signals for the status register 0.

- NSROU(7) NMOF- Monitor fault has been detected U264
- NSROU(6) NMPE- Second pass error detected U263
- NSROU(5) NSERR- Symbol error detected U262
- NSROU(4) NP2ERR- Parity bit 2 error detected U252
- NSROU(3) NP1ERR- Parity bit 1 error detected U261
- NSROU(2) ND2M- Reference clock signal detected *
- NSROU(1) NWIN- (see above in section 7)
- NSROU(0) NMON- This node is the active monitor U10

* Note this signal is derived from the PFD2M circuit and is simply passed straight on to REGO as part of the bus NSROU.

3.5 MAP

The routing Table comprises two 64 bit entry bit maps, Map1 & Map2 corresponding to bits 15 - 10 and 9 - 4 of the DA.

Map1 is organized as 8 Rows and 8 Columns Read/Write

Bits 15 - 13 of the DA provide the Row address
Bits 12 - 10 " " " " Column address
Map2 is organized as 4 Rows and 16 columns for ring accesses, and as 8 Rows and 8 columns for CPU accesses. Bits 9 - 7 of the DA provide the Row address, Bits 6 - 4 \( \text{Column address} \)

The Read organization of Map2 allows its Row address to be contained within a nibble boundary, minimizing the Row access time for a Ring access, whilst the write organization enables byte wide access from the Control Bus.

**Normal Programming Mode** \( \text{MAEN} = 0 \)

Both maps are programmed by the CPU via the Control Bus - CON(7:0). The table on page 3 shows the correlation between DA(15-4) and the CPU address and data. Programming multiple entries into the same row address requires the data parts of the table to be ORed. Similarly when modifying the contents of the MAP, a Read/Modify Write cycle should be used ORing the old data with the new.

CPU address bit 4 indicates a MAP address, with Map2 addressed by the Lower Data Strobe (LDS:N) and Map1 addressed by the Upper (UDS:N). The lower three bits of the address bus - ADD(2:0) - provide the Row address for CPU byte wide Read/Write access to the selected Map. Address bit 3 is used for test purposes only.

Addresses to the Map from both sources are latched in MALATCH1-3, so that ADD(2:0) map directly to DA(15:13) for Map1 and to DA(9:7) for Map2. All zero's in DA(12:10) and DA(6:4) address the least significant column in the selected Row. The CPU writes a 1 to the MAEN bit when programming is completed, which enables ring accesses to commence. MAP1 and MAP2, the two Map outputs, are held low whilst MAEN = 0.

**Normal Running Mode** \( \text{MAEN} = 1 \)

In normal mode, a start of slot delimiter, JK, entering the node from the ring, generates S1START. This is delayed in the Map control logic block (MAPCONT), which generates three timing strobes HIPERT(1:3) which latch the ring Map addresses - DA(15:4) - into MALATCH(1:3). The first timing strobe, HIPERT1, asserts RACC which disables any CPU access by forcing ROWSEL and COLSEL low, for the duration of the ring access.
A Map CPU Read access is initiated when one of the Data Strobes (NUDS, NLDS) is asserted low, with ADD(4) high. This condition is fed into MAPCONT, where it is passed through a two bit shift register MAPCONT/U10, U15, allowing two symbol clock periods to complete the access. NREADY is then asserted low, which latches the data in one of the byte wide holding latches - MAP/U19 or U18 - and allows the PH to assert DTACK:N, confirming valid data to the CPU. A ring access occurring now, will not disturb the data in the holding latches, so NREADY remains low.

If, however, H1PERT1 occurs before the CPU access is complete, the two bit shift register is cleared by RACC and NREADY remains high, forcing DTACK:N to remain high. This obliges the CPU to insert wait states until completion of the ring access. The ring access time is defined by the islot counter, commencing when the counter reaches state 2 (102 time), and finishing at state 15 (115 time). If at, or after, 115 time, a CPU access is requested, RACC returns inactive low, releasing the clear from the shift register, and enabling ROWSEL to latch the new CPU addresses.

Test Mode  MAEN = 0  NTSTMAP = 0

Every memory element in both Map1 and Map2 can be written to and read from the Control Bus, simplifying the testing of the memory arrays, and Row address latches. NTSTMAP, the Map test bit 3 from REG9, enables testing of the column address latches and multiplexers. ADD(3) is enabled during test mode to demultiplex and latch ADD(2:0) into Row and Column addresses. ADD(2:0) are fed to the Row address latches when ADD(3) = 0, and to the Column address latches when ADD(3) = 1. The outputs from the two column multiplexers can be observed on bit 0 of the Control Bus, when NTSTMAP is set.
5.4 APPENDIX D  SLOT HEADER STRUCTURE

Slot Delimiter - 8 bits

Slot Header - 40 bits

- data

J : K : C1 : DA : C2 : Cid : Data

- sync


- monitor

: 4 bits : 16 bits : 4 bits : 16 bits : 128 bits

- broadcast

M in the C1 field is set by the monitor. If the bit is set when the monitor reads it, the slot is overwritten.

DA is the destination address of the slot.

The two parity bits P1 AND P2 have the following scope:

P1 covers the fields C1, DA and P1
P2 covers the fields P2, S/A, H bits and the Cid field.

The remaining control field bits are used to signify the slot type:

<table>
<thead>
<tr>
<th>C1</th>
<th>C2</th>
<th>SLOT TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>00</td>
<td>Trial slot type 0</td>
</tr>
<tr>
<td>0010</td>
<td>01</td>
<td>Trial slot type 1</td>
</tr>
<tr>
<td>0100</td>
<td>00</td>
<td>Reset slot</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DATA SLOTS</td>
</tr>
<tr>
<td>1000</td>
<td>00</td>
<td>Synch slot - low priority</td>
</tr>
<tr>
<td>1000</td>
<td>01</td>
<td>Synch slot - high priority</td>
</tr>
<tr>
<td>1100</td>
<td>00</td>
<td>Asynch slot</td>
</tr>
<tr>
<td>1100</td>
<td>01</td>
<td>Asynch header slot</td>
</tr>
<tr>
<td>1110</td>
<td>00</td>
<td>Signalling slot</td>
</tr>
<tr>
<td>1110</td>
<td>01</td>
<td>Signalling header slot</td>
</tr>
</tbody>
</table>

* H bit is used as part of the type field (in conjunction with 2 bits of Cid) to give up to 8 types of synch slot.
### 3. PH CHIP OPERATION

#### 3.1 BIT, NIBBLE, BYTE AND WORD ORDER ON ORWELL RINGS

**3.1.1 RING SYMBOL CODES** - Information enters and leaves the PH-chip in five bit coded symbols. Bit_5 is a non-data flag which is set to ZERO when bits 4 to 1 represent binary data. When the non-data flag is set to ONE, bits 4 to 1 represent control and slot framing codes as shown in the table below.

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>HEX</th>
<th>NOT DATA</th>
<th>BINARY NIBBLE</th>
<th>COMMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q</td>
<td>F</td>
<td>1</td>
<td>1111</td>
<td>Quiet (no input activity TTL pull-up)</td>
</tr>
<tr>
<td>I</td>
<td>0</td>
<td>1</td>
<td>0000</td>
<td>Idle</td>
</tr>
<tr>
<td>J</td>
<td>9</td>
<td>1</td>
<td>1001</td>
<td>1st symbol of start delimiter *</td>
</tr>
<tr>
<td>k</td>
<td>6</td>
<td>1</td>
<td>0110</td>
<td>2nd symbol of start delimiter *</td>
</tr>
<tr>
<td>T</td>
<td>3</td>
<td>1</td>
<td>0011</td>
<td>Synchronizing slot identifier</td>
</tr>
<tr>
<td>S</td>
<td>C</td>
<td>1</td>
<td>1100</td>
<td>Election slot identifier</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0000</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0001</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0010</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0011</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>0</td>
<td>0100</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>0</td>
<td>0101</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>0</td>
<td>0110</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>0</td>
<td>0111</td>
<td>Data codes in binary</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>0</td>
<td>1000</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>0</td>
<td>1001</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>A</td>
<td>0</td>
<td>1010</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>0</td>
<td>1011</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>C</td>
<td>0</td>
<td>1100</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>D</td>
<td>0</td>
<td>1101</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>E</td>
<td>0</td>
<td>1110</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>0</td>
<td>1111</td>
<td></td>
</tr>
</tbody>
</table>

5 (4321) Data bit numbers

\[
\text{msb of binary data nibble}
\]

[non-data flag is labeled as wire 5 within PH-chip]
### 3.6 REGTOP

#### 3.6.1 REGISTER LIST

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>READ/ WRITE</th>
<th>BIT No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATUS REGISTER O Lo</td>
<td>R 0 0 0 0 RLU ARST QUIET SOF SPE</td>
<td>Interrupt conditions</td>
</tr>
<tr>
<td>STATUS REGISTER O Hi</td>
<td>R MOF MPE SERR P2ERR P1ERR 2M WIN MON</td>
<td></td>
</tr>
<tr>
<td>STATUS/CONT REG. 1 Lo</td>
<td>R/W MDIS MEN REF AEP EPADc EAR M2 M1</td>
<td></td>
</tr>
<tr>
<td>STATUS/CONT REG. 1 Hi</td>
<td>R/W RCR ER PR MAEN SPD3 SPD2 SPD1 SPD0</td>
<td></td>
</tr>
<tr>
<td>STATUS/CONT REG. 2 Lo</td>
<td>R/W RT2 RT1 RTO ART4 ART3 ART2 ART1 ART0</td>
<td></td>
</tr>
<tr>
<td>STATUS/CONT REG. 2 Hi</td>
<td>R TXWT1 TXWTO AROV AUTO-RESET TIMER (4:0)</td>
<td></td>
</tr>
<tr>
<td>STATE MC/Q REG. 3 Lo</td>
<td>R/T LOS2 LOS1. PFS PFT EPAD FIFO TCN ORF</td>
<td></td>
</tr>
<tr>
<td>STATE MC/Q REG. 3 Hi</td>
<td>R/T TC_S TC_T TCT TCD S4 S3 S2 S1</td>
<td></td>
</tr>
<tr>
<td>Next 'd' Register</td>
<td>R/W 16 bit</td>
<td></td>
</tr>
<tr>
<td>d - Value</td>
<td>R 16 bit</td>
<td></td>
</tr>
<tr>
<td>Node Address Hi</td>
<td>R/W 8 bit</td>
<td></td>
</tr>
<tr>
<td>Node Address Lo</td>
<td>R/W 8 bit</td>
<td></td>
</tr>
<tr>
<td>Election Address Hi</td>
<td>R 8 bit</td>
<td></td>
</tr>
<tr>
<td>Election Address Lo</td>
<td>R 8 bit</td>
<td></td>
</tr>
<tr>
<td>Slot Count Register</td>
<td>R/W 16 bit</td>
<td></td>
</tr>
<tr>
<td>Slot Phasing Counter Lo</td>
<td>R 8 bit SPL(7) TRIG(6) SPC(5:0)</td>
<td></td>
</tr>
<tr>
<td>Test Reg Hi R/W DLTCN CTEST TSTMSK TSTCT TSTMAP MCWE RETEST TDD1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input Slot Synchronizer Lo</td>
<td>R 8 bit</td>
<td></td>
</tr>
<tr>
<td>Input Slot Synchronizer Hi</td>
<td>R 8 bit</td>
<td></td>
</tr>
<tr>
<td>Fifo Input Address Counter Lo</td>
<td>R 8 bit</td>
<td></td>
</tr>
<tr>
<td>Fifo Input Address Counter Hi</td>
<td>R 8 bit</td>
<td></td>
</tr>
<tr>
<td>Masked Reset Ctr/Phase Cont Ctr Lo</td>
<td>R 8 bit MPC(7:6)MASK(5) Cab(4:0)</td>
<td></td>
</tr>
<tr>
<td>Masked Reset Cont Register Lo</td>
<td>W 3 bit MPR(7:6) MASK(5)</td>
<td></td>
</tr>
<tr>
<td>STATE MC Out Reg Hi</td>
<td>R TT RC LDSLOTCT LDMSK SAVDA FOUTA MSET S1MC</td>
<td></td>
</tr>
<tr>
<td>'r' Register Hi</td>
<td>W 3 bit r(2:0)</td>
<td></td>
</tr>
<tr>
<td>Reset Rate</td>
<td>R 16 bit</td>
<td></td>
</tr>
<tr>
<td>Reset Timer (test only)</td>
<td>R 16 bit</td>
<td></td>
</tr>
<tr>
<td>(16:23) Map1 &amp; Map2</td>
<td>2X6 bit Route</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S2 S1</th>
<th>MDIS MEN</th>
<th>S4 S3 (Start mode only)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0</td>
<td>Comms 0 0 Monitor by election</td>
<td>0 0 IF6 Procedure</td>
</tr>
<tr>
<td>0 1</td>
<td>Start 0 1 Invoked Monitor</td>
<td>0 1 Await Slot</td>
</tr>
<tr>
<td>1 0</td>
<td>Sync 1 0 Monitor Function Disabled</td>
<td>1 0 Set Fifo Length</td>
</tr>
<tr>
<td>1 1</td>
<td>Election 1 1</td>
<td>1 1 Size Ring</td>
</tr>
</tbody>
</table>
3.6.1.1 REGISTER LIST NOTATION

ARST - Auto Reset has taken place
AROV - Auto Reset Timer timed out
ART4:0 - Auto Reset Timer selection (00000 = 62.5uS --> 11111 = 2mS)
AEP - Auto Extra Padding
EPAD - Extra Padding flag - set by node in post election AEP
EPADc - Extra Padding control flag - the EPAD flag copied to sub nodes
ER - Erroneous slot reception enabled
FIFO - Fifo initialized flag
FOUTEN - Fifo output addr. ctr. enable
LMSK - load mask pulse
LDSLOTCT - load slot count into reg10
LOS1 - lose first election test ie DA<NA
LOS2 - lose election test in second successive 'S' slot
MAEN - Map Enable
MDIS - Monitor Disable
MEN - Monitor Enable
MOF - Monitor failure detected
MON - Monitor mode status/control
MPC - Mask Phase Counter
MPR - Monitor Pass Error (M=1 detected by monitor)
MSET - Monitor bit setting signal
ORF - Outstanding Reset Flag
P1ERR - Parity error detected P1
P2ERR - Parity error detected P2
PR - Priority Receive only control
PFS - 'S' slot persistence flag
PFT - 'T' slot persistence flag
QUIET - Ring break zero codes received
RC - Receive Control
RCR - Register Clear on Read
REF - Reference type (1 if PSTN clock source)
RLU - Reset lock-up detected
RTO-2 - Reset timer interval selection  (000 = 125uS --> 100 = 2mS)
SAVDA - Save the DA of this slot for election tests
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Slot Phase Difference Control (Symbols)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SERR</td>
<td>Symbol error detected</td>
<td>(4, 8, 12, 16, 20, 24, 28, 32, 36, 40)</td>
</tr>
<tr>
<td>SOV</td>
<td>Slot overlap detected</td>
<td></td>
</tr>
<tr>
<td>SPDO-3</td>
<td>Slot phase difference control</td>
<td></td>
</tr>
<tr>
<td>SPE</td>
<td>Slot phase error detected</td>
<td></td>
</tr>
<tr>
<td>SPL</td>
<td>Slot Phase Lock</td>
<td></td>
</tr>
<tr>
<td>S1MC</td>
<td>Data mux control signal</td>
<td></td>
</tr>
<tr>
<td>TCD</td>
<td>Transmit control DATA SLOT</td>
<td></td>
</tr>
<tr>
<td>TCR</td>
<td>Transmit control RESET SLOT</td>
<td></td>
</tr>
<tr>
<td>TCT</td>
<td>Transmit control TRIAL SLOT</td>
<td></td>
</tr>
<tr>
<td>TC_S</td>
<td>Transmit control [s] SLOT</td>
<td></td>
</tr>
<tr>
<td>TC_T</td>
<td>Transmit control [t] SLOT</td>
<td></td>
</tr>
<tr>
<td>TT</td>
<td>Trial type</td>
<td></td>
</tr>
<tr>
<td>TXWT1-0</td>
<td>Transmit Wait Timer (2 bits)</td>
<td></td>
</tr>
<tr>
<td>WIN</td>
<td>Win monitor election flag</td>
<td></td>
</tr>
<tr>
<td>2M</td>
<td>Reference clock signal detected</td>
<td></td>
</tr>
</tbody>
</table>

- Appendix III -
<table>
<thead>
<tr>
<th>PIN</th>
<th>VDD</th>
<th>RO(1)</th>
<th>RO(2)</th>
<th>SCK</th>
<th>RTP(1)</th>
<th>RTP(3)</th>
<th>VSS</th>
<th>RTP(4)</th>
<th>RTP(6)</th>
<th>SELCK</th>
<th>PFo</th>
<th>MDN</th>
<th>VDD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2,7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3,1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3,3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3,0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3,2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3,5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1,6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1,7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1,8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1,9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5,1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5,2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5,3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5,4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5,5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5,6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5,7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5,8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6,5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6,6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6,7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6,9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7,0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7,1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7,2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7,3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7,4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7,5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7,6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7,7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7,8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8,0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8,1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8,2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8,3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8,4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8,5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**ORWELL PH**

**VIEW ON PIN SIDE OF PACKAGE**
### 5. APPENDICES

#### 5.1 APPENDIX A ORWELL PH CHIP PINOUT – 100 PIN PGA

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>No. of Pins</th>
<th>Pin No.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Interface</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RX/TX Bus I/O</td>
<td>17</td>
<td>96:99, 1:8, 10, 11, 15, 16, 18</td>
</tr>
<tr>
<td>POWER</td>
<td>3</td>
<td>12:14</td>
</tr>
<tr>
<td>Transmit FIFO Clock</td>
<td>1</td>
<td>19</td>
</tr>
<tr>
<td>Receive FIFO Clock</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>Output Enable</td>
<td>1</td>
<td>21</td>
</tr>
<tr>
<td>SLOT</td>
<td>1</td>
<td>22</td>
</tr>
<tr>
<td>Transmit Request</td>
<td>1</td>
<td>23</td>
</tr>
<tr>
<td>POWER</td>
<td>1</td>
<td>24</td>
</tr>
<tr>
<td>POWER</td>
<td>1</td>
<td>25</td>
</tr>
<tr>
<td><strong>Ring Interface (TX)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ringout Data</td>
<td>5</td>
<td>26:30</td>
</tr>
<tr>
<td>Transmit Clock</td>
<td>1</td>
<td>31</td>
</tr>
<tr>
<td><strong>Received Slot Controls</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Received Slot types</td>
<td>7</td>
<td>32, 33, 35, 36, 40, 41, 43</td>
</tr>
<tr>
<td>POWER</td>
<td>3</td>
<td>37:39</td>
</tr>
<tr>
<td>Received Slot Type Strobe</td>
<td>1</td>
<td>44</td>
</tr>
<tr>
<td>Transmit Clock Select</td>
<td>1</td>
<td>45</td>
</tr>
<tr>
<td>National Network Clock</td>
<td>1</td>
<td>46</td>
</tr>
<tr>
<td>Phase Frequency Detector</td>
<td>1</td>
<td>47</td>
</tr>
<tr>
<td><strong>Torus Interface</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Monitor</td>
<td>1</td>
<td>48</td>
</tr>
<tr>
<td>D Counter Clock</td>
<td>1</td>
<td>49</td>
</tr>
<tr>
<td>POWER</td>
<td>1</td>
<td>50</td>
</tr>
<tr>
<td>Reset Lines</td>
<td>4</td>
<td>51:54</td>
</tr>
<tr>
<td>Slot Phase Reference</td>
<td>1</td>
<td>55</td>
</tr>
<tr>
<td><strong>CPU Interface</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Initialise Bar</td>
<td>1</td>
<td>56</td>
</tr>
<tr>
<td>Read/Write</td>
<td>1</td>
<td>57</td>
</tr>
<tr>
<td>Chip Select Bar</td>
<td>1</td>
<td>58</td>
</tr>
<tr>
<td>PIN NAME</td>
<td>No. of Pins</td>
<td>Pin No.</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------</td>
<td>---------</td>
</tr>
<tr>
<td>Address Strobe Bar</td>
<td>AS_N</td>
<td>60</td>
</tr>
<tr>
<td>Interrupt Bar</td>
<td>INTRPT_N</td>
<td>61</td>
</tr>
<tr>
<td>POWER</td>
<td>VSS</td>
<td>62:64</td>
</tr>
<tr>
<td>Control Bus</td>
<td>D(0:7)</td>
<td>65,66,68:73</td>
</tr>
<tr>
<td>CPU Address Bus</td>
<td>A(4:0)</td>
<td>74,76:79</td>
</tr>
<tr>
<td>POWER</td>
<td>VDD</td>
<td>75</td>
</tr>
<tr>
<td>Upper Data Strobe Bar</td>
<td>UDS_N</td>
<td>80</td>
</tr>
<tr>
<td>Lower Data Strobe Bar</td>
<td>LDS_N</td>
<td>81</td>
</tr>
<tr>
<td>Data Transfer Ack Bar</td>
<td>DTACK_N</td>
<td>82</td>
</tr>
<tr>
<td>Testout</td>
<td>TO</td>
<td>83</td>
</tr>
<tr>
<td>Tri-state Testin</td>
<td>NZTI</td>
<td>84</td>
</tr>
<tr>
<td>Ntestin</td>
<td>NTESTI</td>
<td>85</td>
</tr>
</tbody>
</table>

Ring Interface (RX)

<table>
<thead>
<tr>
<th>PIN NAME</th>
<th>No. of Pins</th>
<th>Pin No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive Clock</td>
<td>RSCK</td>
<td>86</td>
</tr>
<tr>
<td>POWER</td>
<td>VSS</td>
<td>87:89</td>
</tr>
<tr>
<td>Ringin Data</td>
<td>RI(5:1)</td>
<td>90,91,93:95</td>
</tr>
<tr>
<td>POWER</td>
<td>VDD</td>
<td>100</td>
</tr>
</tbody>
</table>
3.2.5.2 ELECTION LOCK UP

A ring lock up condition can occur if more than one node enters ELECTION with MON set. This can occur if MEN is set or the MON:N line is allowed to float to ground in two nodes.

In this case both will enter SYNCH when the CCP sets \( M_1 = M_2 \) and neither will repeat the others T slots but will continually transmit their own. The whole ring will then remain in SYNCH until only one node has MON set.

3.3 ORWELL START SEQUENCE

Abbreviations: OSC - Out slot counter
FIFIN - FIFO input address counter
FIFOUT - FIFO output address counter
FIFO=0 - FIFO unconnected, i.e. \( S1\text{REFIN}=S1\text{REFOUT}=1 \)
FIFO=1 - FIFO connected into data path
S1FINEN - enable signal to FIFIN
S1FOUTEN - enable signal to FIFOUT

The padding scheme begins by initializing the FIFO with an IF6 procedure. This enables the FIFO to operate in dynamic equilibrium, with the FILL rate equalling the UNLOAD rate, whilst maintaining a sufficient residual data storage of 6 nibbles to allow for jitter. An input start delimiter will quickly fall to the bottom of the FIFO and when detected, the UNLOAD clock is stopped, causing the FIFO to fill up with the remainder of the slot.
The node meanwhile, has been transmitting a continuous stream of $S$ or $T$ slots, depending on the point of entry into START. If now the UNLOAD clock is enabled, coincident with the start of a new slot transmission (ie. when OSC=0), the FIFO will contain the required padding, necessary to pad out the ring to an integer number of whole slots.

To ensure equal slot counts in TORUS, the concept of the EPAD (Extra Padding) is used. If the start of a new slot transmission occurs less than seven clock cycles after the UNLOAD clock is stopped, EPADc is set, and the UNLOAD clock is not enabled until the next OSC=0. A torus subnode with its EPADc bit set will not look for a start of slot transmission until 14 clock cycles after the UNLOAD clock is stopped. This is sufficient margin to ensure the first occurrence of OSC=0 is ignored, guaranteeing equal slot counts in each arm.

It is convenient for a monitor capable node to initialize its FIFO on entering ELECTION, with a residual data storage of nearly half the memory (IF22 procedure) to allow for frequency differences between transmit clocks. On losing the election, the FIFO is shorted out completely, causing a disturbance in slot length but only as far as the next down stream node still bidding for monitor. A consequence of this, is that a monitor capable node in ELECTION, can use its raw SCK (rather than an on chip multiplexed clock) to clock the data registers, Inslot Counter, and UNLOAD clock to the FIFO.

3.4 A STATE MACHINE TO IMPLEMENT THE ORWELL PROTOCOL

3.4.1 INTRODUCTION

The function of this machine is to control the initialization of either a single Orwell Ring or a multiple ring Orwell Torus; then to control the communications between nodes. The machine has four primary states.

1. COMMS The state in which normal communications are carried out; and the network is monitored for malfunction.

2. ELECTION The state in which nodes with monitor capability compete in an election for the duty of ring monitor. All other nodes lock onto the incoming signals irrespective of their source, and passively repeat them.
APPENDIX IV

Circuit diagrams

Multiple queue interface:
   AT interface card
   4-way interface card
   Tx request arbiter

Receiver queue
Transmit queue

Packetiser
Depacketiser

Physical layer Rx
Physical layer Tx

PHIC slave board
PHIC Monitor board
Multiple Queue Interface
Tia Rodgers and Charles Nche
Loughborough University
May 1992
Page 1 of 2
Node Sub-Address and Rx Decoding

Dead Important Timing

Don't forget!!
Generate RST, DMA, and DEC, DMA
from T1, en and SBT OR's with DEN
and don't talk to strangers

- 189 -
Rx Physical Layer
Loughborough University
November 1991
C.F. Nche

- Appendix IV -

[Diagram of circuit with labels and components]