AN INVESTIGATION INTO THE ROLE OF X.25 IN A LOCAL AREA DATA NETWORK

by

JOSÉ A. RUELA S. FERNANDES

A Thesis submitted for the Degree of

Doctor of Philosophy

University of Sussex

May 1982

T/12

681.3(043) FER j/INV
UNIVERSIDADE DO PORTO
Faculdade de Engenharia
BIBLIOTECA
N.º 13112-of
CDU
Data 7/12/1983

043 D
F 399 i

TABLE OF CONTENTS

ACKNOWLEDGMENTS.....I

STATEMENT.....II

ABSTRACT.....III

CHAPTER 1

EVOLUTION OF COMPUTER NETWORKS.....1.1

1.1 Introduction.....1.1
1.2 Long-haul networks.....1.2
1.2.1 Packet Switching.....1.3
1.2.2 The evolution of Packet Switching Networks.....1.4
1.3 Evolution of layered Architectures and Protocols..1.6
1.3.1 Protocol standardization.....1.7
1.3.2 The OSI reference model.....1.8
1.4 Local Area Networks.....1.11
1.4.1 Objectives.....1.11
1.4.2 Technologies and Protocols.....1.12
1.4.3 The role of X.25.....1.14

CHAPTER 2

LOCAL AREA NETWORK ARCHITECTURES.....2.1

2.1 Introduction.....2.1
2.2 Some alternative topologies.....2.1
2.2.1 Star network.....2.2
2.2.2 Ring network.....2.3
2.2.3 Bus network.....2.6
2.2.3.1 Baseband bus.....2.7
2.2.3.2 Broadband bus.....2.9
2.3 Proposed Architecture.....2.11
2.3.1 Network requirements.....2.11
2.3.2 Network topology.....2.13
2.3.3 Packet Switching Exchange.....2.14

CHAPTER 3

RATIONALE AND SPECIFICATION OF X.25.....3.1

3.1 Introduction.....3.1
3.1.1 Layered design.....3.1
3.1.2 X.25 as a local interface.....3.2
3.2 Data link layer (level 2).....3.3

3.2.1 Frame structure....3.3
3.2.2 Link control....3.5
3.2.3 Information transfer....3.6
3.2.3.1 Acknowledgment mechanism....3.6
3.2.3.2 Error control....3.7
3.2.3.3 Retransmission strategy....3.9
3.2.3.4 Link reset....3.11
3.2.3.5 Flow control....3.12
3.3 Packet layer (level 3)....3.14
3.3.1 Virtual Circuit Service....3.15
3.3.1.1 Call set-up....3.15
3.3.1.2 Data transfer....3.16
3.3.1.3 Call clearance....3.17
3.3.2 Error control....3.17
3.3.2.1 Time-out recovery....3.18
3.3.2.2 Call reset....3.19
3.3.3 Flow control....3.21

CHAPTER 4

PACKET SWITCHING EXCHANGE - HARDWARE DESIGN....4.1

4.1 Design philosophy....4.1
4.2 Link Access Unit....4.2
4.2.1 Serial link interface....4.3
4.2.1.1 Frame transmission....4.4
4.2.1.2 Frame reception....4.9
4.2.2 The Counter/Timer....4.13
4.2.3 Interunit Interface....4.14
4.2.3.1 Packet transmission....4.14
4.2.3.2 Packet reception....4.16
4.3 Exchange Management Unit....4.17
4.4 Interchannel Unit....4.18
4.4.1 Packet transfer requirements....4.18
4.4.2 Hardware design....4.19
4.4.3 Bus arbiter....4.20

CHAPTER 5

PACKET SWITCHING EXCHANGE - PROTOCOLS....5.1

5.1 Link management....5.2
5.1.1 Acknowledgment strategy....5.3
5.1.2 Buffer allocation....5.6
5.2 Flow and congestion control....5.8
5.2.1 Exchange congestion....5.11
5.2.2 Deadlocks....5.13
5.2.3 Flow control strategy....5.14
5.2.3.1 Virtual circuit flow control....5.15
5.2.3.2 Network access flow control....5.16
5.2.3.3 Deadlock avoidance....5.17
5.3 Internal protocol....5.19
5.3.1 Internal logical channel assignment....5.20
5.3.2 Virtual circuit management....5.22

5.3.3 Restart.....5.23
5.3.4 Recovery mechanism.....5.25
5.3.5 Channel state description.....5.26
5.4 Interface between the Link and Packet layers.....5.27
5.4.1 Link level failure.....5.27
5.4.2 Packet level failure.....5.29

CHAPTER 6

PACKET SWITCHING EXCHANGE - SOFTWARE.....6.1

6.1 Concurrency control.....6.1
6.1.1 Scheduling algorithm.....6.2
6.2 Link Access Unit Initialization.....6.4
6.3 Buffer Manager.....6.5
6.3.1 Buffer requirements.....6.6
6.3.2 Buffer pool structure.....6.7
6.3.3 Buffer transfer routines.....6.9
6.4 Timer Manager.....6.10
6.5 Link level processing.....6.11
6.5.1 Link level procedures.....6.12
6.6 Packet level processing.....6.15
6.6.1 Packet level procedures.....6.16
6.7 Link and bus transfers.....6.18
6.8 Implementation details.....6.19

CHAPTER 7

EXCHANGE MONITORING AND PERFORMANCE EVALUATION.....7.1

7.1 The Exchange Management Unit.....7.1
7.1.1 Exchange Management Unit commands and their functions.....7.2
7.1.2 Implementation details.....7.2
7.2 Execution of Monitor commands by a Link Access Unit.....7.6
7.2.1 Example dialogue.....7.10
7.3 Performance evaluation.....7.11
7.3.1 Central exchange characteristics.....7.11
7.3.2 Analytical models.....7.12
7.3.3 Exchange performance figures.....7.13
7.3.3.1 Throughput.....7.14
7.3.3.2 Delay.....7.17
7.3.3.3 Conclusions.....7.21

CHAPTER 8

DISCUSSION AND CONCLUSIONS.....8.1

8.1 Broadband techniques.....8.1
8.2 Multilink Link Access Units.....8.2
8.3 Additional network components.....8.5

8.3.1 Network address scheme.....8.7
8.4 Conclusions.....8.10

Appendices

APPENDIX A : Additional aspects of the Link Access Procedures

APPENDIX B : Exchange program listings - X.25 control software

APPENDIX C : Exchange program listings - EMU control software

APPENDIX D : Laboratory facilities

APPENDIX E : Publications

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervisor, Dr. F. Halsall, for his friendly encouragement, valuable suggestions and constant guidance throughout the whole period of my research programme.

I must also express my acknowledgments to the Fundação Calouste Gulbenkian, Lisbon, Portugal, for the financial support that made possible my stay at Sussex to carry out this work.

Finally I am very grateful to my wife Isabel and my son Pedro for their great patience, encouragement and moral support, without which I would not have been able to conduct this research.

STATEMENT

The work described in this thesis is the author's own, except where otherwise indicated, and has not been submitted in whole or part for another degree at this or any other university.

José A. Ruela S. Fernandes

JOSÉ A. RUELA S. FERNANDES

School of Engineering and Applied Sciences

University of Sussex

Title : AN INVESTIGATION INTO THE ROLE OF X.25 IN A LOCAL AREA DATA NETWORK

Author : JOSÉ A. RUELA S. FERNANDES

ABSTRACT

Large geographically distributed data communication networks have been in existence for a number of years and indeed internationally agreed standards have been established for their use. The standard access protocol to public packet-switched networks is X.25 and a wide range of hardware and software support for X.25 is now available from most large computer manufacturers.

It is with these activities as a backcloth that a proliferation of local area networks have come into being during the past few years. Until very recently, however, the lack of any form of standards for use with these networks has meant that these differ considerably one from another. One of the primary aims of this thesis, therefore, has been to investigate the role the X.25 standard used for wide area networks might have in a local area context.

The topology proposed in this thesis is a hybrid network which is comprised of a high-level Packet Switching Exchange which adheres to the X.25 standard and this then supports the possibly wide range of other network resources. Typically these will include large mainframe and minicomputers, communities of unintelligent terminals, other smaller subnetworks and probably a connection to one or more public data networks.

The hardware and software design of a microprocessor-based modular packet switch is presented which is suitable for use with such a network topology. It is comprised of a variable number of Link Access Units each controlling a number of full-duplex serial links. These are interconnected by means of a high-speed parallel bus to which is also connected an Exchange Management Unit to provide performance monitoring and diagnostic functions.

The thesis is structured into eight chapters. Chapter 1 traces the evolution of computer communications networks to achieve resource sharing and the importance of the use of packet switching techniques. The role of layered architectures and communication protocols leading to the standardization of the X.25 protocol is presented and this is followed in Chapter 2 by a review of some of the alternative local area network architectures currently in use. Their topological organizations and the different media access control structures and protocols are described.

Chapter 3 discusses the X.25 protocol, its layered design and the set of functions provided by each layer. The more controversial aspects of the protocol are interpreted and analysed as well as the different control mechanisms used by each level and the possible implementation alternatives and interlevel relationships.

Chapter 4 summarizes the design aims of the modular Packet Switching Exchange and describes its hardware design. The exchange is comprised of a number of Link Access Units and an Exchange Management Unit which are interconnected by means of a high speed Interchannel Unit.

Chapter 5 describes the use of the X.25 protocol in connection with the exchange topology and its implementation. The flow and congestion control mechanisms utilized are discussed. An end-to-end virtual circuit flow control is combined with a network access flow control, the latter using an Input Buffer Limit scheme based on a stuctured buffer pool. The internal protocol devised for the interunit communications subsystem is presented and this includes the internal channel assignment

scheme and various recovery mechanisms. Finally, the interlevel signalling is discussed and solutions are proposed to handle possible failures within each level.

Chapter 6 discusses the software organization of the exchange. This includes the identification of the main processing tasks, their forms of communication and the implementation of the scheduling algorithm to achieve concurrency. The Buffer and Timer Managers are then described and the processing structure of each protocol level is detailed by means of event/state tables. Finally the laboratory facilities used to develop and test the exchange hardware and software are presented.

Chapter 7 describes the Exchange Management Unit and the set of functions implemented to provide performance monitoring and diagnostic facilities. The performance figures of the exchange are also assessed by means of an analytical model. Useful results are established concerning the exchange stability and its throughput and delay profile.

Chapter 8 discusses further topics that should be considered in order to overcome some of the exchange limitations and to improve its potential usefulness; namely the use of broadband techniques and the provision of additional components (PAD's, gateways, etc). The use of a single Link Access Unit to control several serial links is also proposed thus constituting an inexpensive and self-contained packet switch. Finally, some conclusions are presented, stressing in particular the importance of current efforts of standardization within local area networks and how they should be integrated into a broader perspective to achieve wider network compatibility and system interconnection.

The work described in this thesis has been the subject of

two technical papers and a copy of each of these is included in an appendix at the end of the thesis.

CHAPTER 1

EVOLUTION OF COMPUTER NETWORKS

1.1 Introduction

The evolution of computer networks over the past decade has been dominated by the continuing advances in integrated circuit technology and the rapid decrease in computer hardware costs. These have made possible the application of new communication techniques and the development of new philosophies in the area of network architectures.

The increasing use of computer systems has created a need, both for technical and economical reasons, for a convenient access mechanism to the often widely distributed computing resources within an organization. These include: remote access to computing power and specialized resources, flexibility of using a variety of different resources, increased reliability and availability, possibility of data transfer between systems, thus maximizing the cost-effective use of computing resources and reducing hardware and software development costs.

Multiaccess time-sharing systems were the first step in meeting these requirements but it was the advent of computer communication networks that heralded the beginning of true interactive sharing of distributed resources [1].

In such networks, host computers and terminals are interconnected by means of a separate communications subnetwork

to form a clean separation between the interconnected computers and terminals and the switching nodes which comprise the network, as shown in Fig. 1.1. Optimization and management of the subnetwork resources (transmission bandwidth and nodal resources), shared by the whole community of network users has become a central issue in the evolution of computer networks.

1.2 Long-haul Networks

Early computer networks covered a wide geographic area and normally made use of common carrier leased lines as the internodal transmission media operating typically at data rates up to 50 Kbaud. These are known, therefore, as long-haul networks.

In such networks, the choice of switching techniques used within the communications subnetwork is mainly determined by the need to efficiently share the transmission media. Since the communications lines used were leased from the telephone companies, early networks utilized circuit switching techniques. Conventional circuit switching however, using fixed pre-allocation of transmission bandwidth, is particularly inefficient for use with bursty traffic (high ratio of peak to average demands) as is typical of computer communications [2,3].

Hence, with the increasing availability of computer hardware and the newly dominant communication costs, by the late nineteen sixties it became technologically possible and economically advantageous to use packet switching as the switching technique in the communications subnetwork [4].

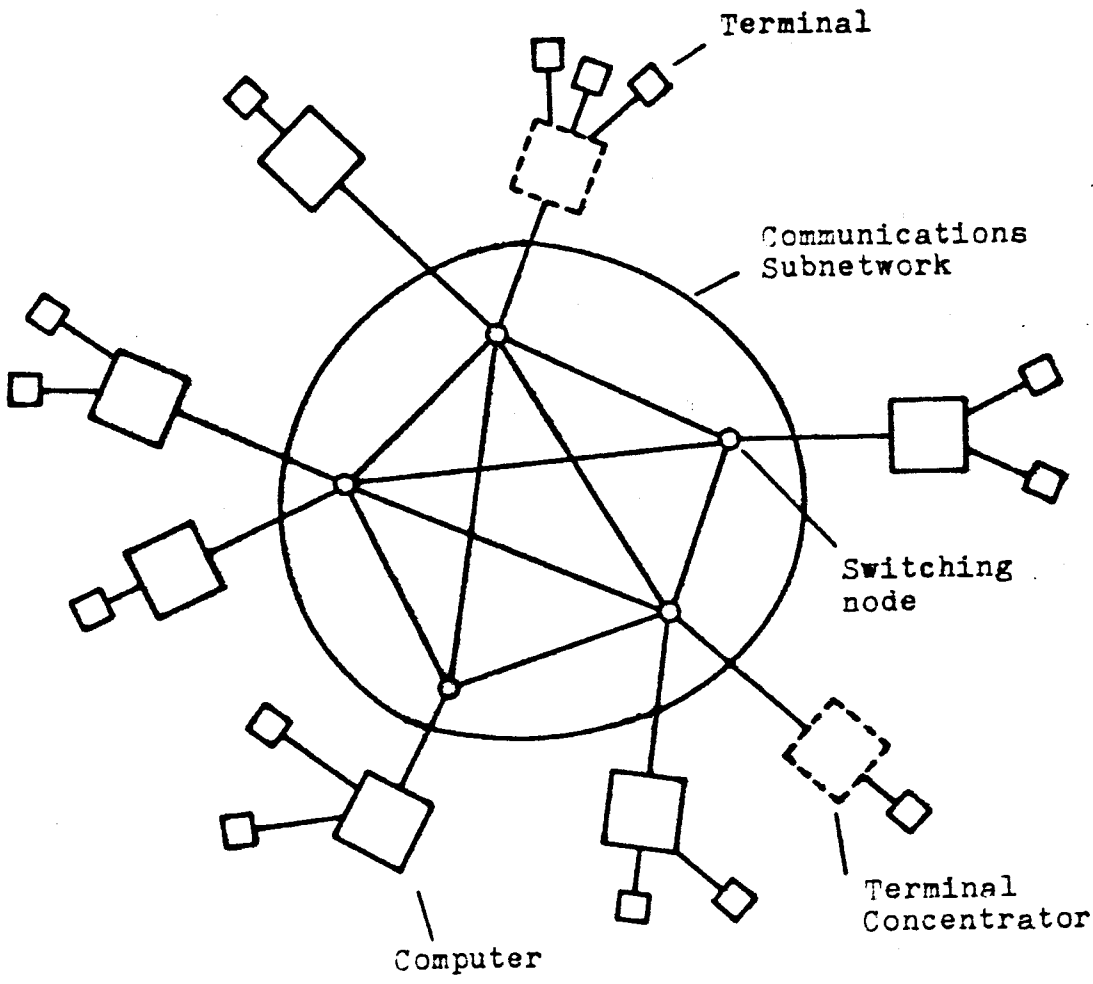


Fig. 1.1 - A computer communications network,

1.2.1 Packet Switching

Packet switching is based on a dynamic allocation of transmission bandwidth and hence of the subnetwork resources. It therefore achieves a more efficient use of the available bandwidth by smoothing the total demand over a large population of users with average low demands.

Messages are fragmented into segments or packets (usually of a maximum size) which travel across the network in a store-and-forward manner, being temporarily buffered and delayed at each switching node as they compete for network resources. For this reason the switching nodes are relatively complex in terms of processing (network control) and storage management.

Packet switching networks, however, offer a number of advantages over conventional circuit switching: for example code and protocol conversion within the switching nodes of the network, efficient handling of bursty traffic, data rate conversion and also the availability of alternative routing strategies. The provision of multiple paths between each source-destination pair in the subnetwork improves the network reliability (in case of link or nodal failures) and allows an adaptation to fluctuating traffic conditions. Most long-haul networks, therefore, utilize unconstrained mesh topologies.

Because of the varied requirements of packet switching networks two types of services are normally provided: a datagram service and a virtual circuit service.

A datagram service provides a transaction processing type of user facility since a datagram is a self-contained packet carrying both source and destination addresses and being

transmitted independently and with no correlation to other packets. The datagram service makes extensive use of alternative routing and normally only the "best delivery" of packets is guaranteed. That is, packets are delivered to the destination eventually out of sequence with possible losses or duplicates. Thus resequencing and recovery actions must be carried out by the host computers themselves.

A virtual circuit service provides a connection type of service since, as the name implies, a virtual circuit is a temporary logical connection set up for the duration of a call between a source-destination pair. A virtual circuit service preserves the correct sequencing of packets and provides detection and recovery mechanisms against losses and duplicates.

1.2.2 The evolution of Packet Switching Networks

Since the early days of the ARPA network [5], a number of networks with different objectives and characteristics have been developed or are currently in use. These include:

(a) Research networks: CYCLADES [6], EIN [7];

(b) Public networks: TRANSPAC [8], DATAPAC [9], PSS [10];

(c) Commercial networks: TYMENET [11];

(d) Computer manufacturer networks: SNA [12], DNA [13].

Although each of these networks has differences in such features as topology, protocols, services and control strategies a wide body of knowledge now exists in packet switching and solid

principles have been established for their operation [14]. Nevertheless, there are still a number of outstanding common problems which have lead to a multiplicity of proposals and solutions in areas that are still subjects of research.

Some of these aspects - network architectures, routing, flow and congestion control, high level protocols, performance evaluation, network interconnection and others - will either be referred to or discussed when they are relevant to the understanding of the network implementation described in this thesis.

Even though considerable progress has been achieved in these fields during the past few years, the lack of a common approach during the early stages of development has introduced a number of severe limitations. These have resulted because:

(a) networks were mainly designed following a bottom-up approach, from the communications subnetwork to the higher level functions and protocols, without a clear and complete identification of the services required by the end users;

(b) design and implementation of higher level protocols were usually delayed, thus preventing networks to be extensively used as the vehicle for interprocess communication between host computers rather than a simple wire-substitute for terminal access;

(c) network interfaces suitable for a wide range of communications requirements were either not available or lacked universal acceptance.

This situation is now changing, however, as a result of the

definition and acceptance of standard protocols, the evolution of local area networks and the growing demand for network interconnection.

1.3 Evolution of layered Architectures and Protocols

Early communication systems were designed to meet specific requirements and so communication interfaces were tailored to the particular applications either through the emulation of existing peripherals or by the introduction of a front-end processor. The advent of distributed, heterogeneous networks of computers with a vast range of different application requirements, however, led to a need to efficiently share communication resources and called for the design of common communication mechanisms and interfaces. The basic design philosophy from which different network architectures evolved was that of modular and hierarchical layered structures [15] based on a few general principles:

(a) each layer uses and adds value to the services provided by lower layers, thus creating new services to the higher layers;

(b) layers in the hierarchy are independent, in the sense that the organization of each layer is transparent to the other layers;

(c) adjacent layers communicate through well defined interfaces.

When these principles are applied to the distributed

communication environment, the communication protocols [16] also appear hierarchically structured in layers, with the higher layers overlaying the lower layers. This naturally leads to:

(a) structured designs supported by a very clean identification of functions and services provided by each layer;

(b) more flexible implementations since layers may be redesigned or replaced by equivalent layers in order to adapt to technological innovations;

(c) the ability to readily add more powerful and user oriented services on top of existing layers (services).

1.3.1 Protocol standardization

A useful network design must be the result of two components: firstly it must be based on the users requirements (top-down approach) and secondly it must be implemented from current technologies (bottom-up approach). Protocol standardization plays an important role in the success of this combined approach. Especially important in this context is the increasing technical support (network interfaces) that are possible, thus reducing implementation and maintenance costs and creating the conditions for wider network compatibility.

Layered architectures and protocols constitute an essential framework for standardization, but the independent evolution of different network architectures, serving a variety of requirements and offering different services, did not work in

favour of this objective. Progress in protocol standardization (or at least wide acceptance of some protocols) was initially directed to standardizing the basic (low level) communication requirements (V.24, HDLC, SDLC ,etc).

The first serious attempt of protocol standardization as a layered hierarchy was achieved by the acceptance of X.25 as the standard protocol to access public packet switching networks[17]. This essentially provides link and network management functions and services but falls short of providing an end-to-end transport service.

X.25 favours the concept of a virtual circuit service as opposed to the datagram service proposed and used in many research networks. The long running debate between virtual circuits and datagrams [18] has prevented wider acceptance of X.25 and has lead to the non existence of any standard protocol at the transport level and above.

These problems became more and more important as network interconnection and wider system interaction required a suitable base of compatibility.

1.3.2 The OSI reference model

Fortunately, a more realistic attitude towards layered architectures and protocol standardization has recently evolved. What was clearly missing was the existence of a common model which different implementations could be referred to without compromising their originality and particular requirements.

The OSI (Open Systems Interconnection) reference model [19],

proposes a conceptual description and the rules of interaction between Open Systems; that is, all systems adhering to the model will be able to communicate (open) and cooperate with each other to perform common, distributed tasks. The model is not concerned with the internal organization of each interconnected system, as only the external behaviour of Open Systems must conform to the OSI architecture (principle of restricted visibility). Reference interfaces constitute the visibility points where the systems adhere to the reference architecture.

The model does not enforce any particular protocol(s); however, by proposing the set of services associated with each level and by defining the concepts used to specify the functions performed by each layer, the model is an important factor towards protocol standardization.

The model proposes a seven layer architecture as represented in Fig. 1.2, corresponding to a well defined hierarchy of functions and services; particularly important as far as the user-oriented layers are concerned.

The three lower communication-oriented layers mainly deal with link and network management functions (error and flow control, routing ,etc). On top of these a transport layer is introduced to provide a reliable, network independent (end-to-end) message transport service, thus isolating communication from user oriented tasks and also masking network differences from the higher protocol layers.

The higher, user-oriented layers must deal with such issues as the establishment and maintenance of logical connections and dialogue control between processes (session layer), the representation and manipulation of stuctured data (presentation

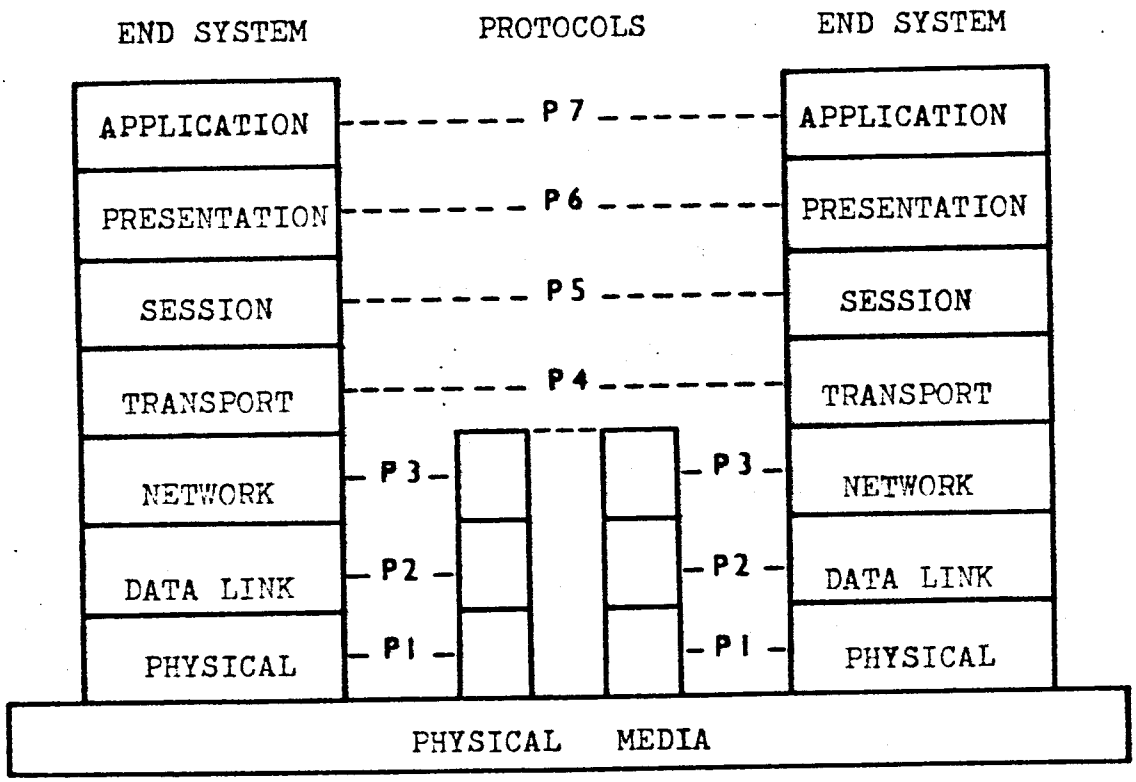


Fig. 1.2 - The Open Systems Interconnection reference model

layer) and the information processing enabling the cooperation of distributed processes (application layer).

Standards at the physical layer (X.21, V.24), at the data link layer (HDLC) and at the network layer (packet level of X.25) are in general agreement with the model. Enhancements of X.25 and in particular the inclusion of datagram facilities [20] are likely to settle existing arguments and allow the standardization of a transport protocol, out of existing proposals [21].

At higher layers, past efforts have been mainly directed to the implementation of terminal and file transfer protocols, identified in the model at the presentation layer. While research networks have usually favoured the Virtual Terminal approach, following the acceptance of X.25 standardization of parametric terminal protocols (X.3, X.28, X.29) [22] has been the objective of public packet switching networks.

Here, non packet-mode terminals access the network via a PAD (Packet Assembler/Disassembler): X.3 defines the PAD, X.28 defines the access to the PAD and X.29 defines the protocol between the PAD and the packet-mode terminal. X.29 is not, however, an end-to-end protocol; in fact it is asymmetrical and it is not applicable to the communication between packet-mode terminals and it does not therefore fit into the OSI model.

The Virtual Terminal approach is more general and allows the handling of terminals in a more uniform way; the evolution of intelligent terminals is likely to be oriented to the support of Virtual Terminal protocols.

1.4 Local Area Networks

It is with these continuing developments as a backcloth that local area networks [23] have emerged. As with long-haul networks in the late nineteen sixties, local area networks are the result of ever continuing developments of computer hardware technology and in particular with the advent of the microprocessor and the widespread availability and use of microprocessor based systems.

Probably the two major reasons that have lead to the development of local area networks were: firstly, it became a common situation in many geographically limited environments (offices, laboratories, groups of buildings, etc), of the existence of a variety of computer systems, typically ranging from large mainframes and minicomputers to smaller microcomputers, intelligent devices and general purpose terminals, the majority of which require to communicate with each other; secondly, the comparatively low communication costs and higher data rates which are possible over the shorter distances involved meant alternative communication strategies could be explored.

1.4.1 Objectives

The prevailing situation in a local area network where bandwidth is not the critical issue but rather the efficient and cost-effective use of local resources, is of course different from the situation within a long-haul network. Local area networks serving a geographically restricted population of users

with a variety of heterogeneous systems and well defined requirements provide the means for:

(a) access to all local resources by any interconnected system (a simple terminal, for example);
(b) easy and fast access to shared data and information;
(c) sharing of specialized resources;
(d) distribution of processing and communication control, thus relieving some systems from the execution of highly demanding centralized functions;
(e) higher flexibility (expansion, reconfiguration, etc).

Although local area networks have evolved to meet current requirements, as with all technological innovations they are now beginning to be used for a range of new applications. However, office automation including information processing and retrieval, teleconferencing, electronic mail, etc, will probably be the domain of most impact in the near future.

1.4.2 Technologies and Protocols

Local area networks are characterized by the wide choice of transmission media that are possible, for example, twisted-pair, coaxial cable, optical fibres, etc. Typical data rates currently range up to about 10 Mbit/s although the use of optical fibres, still limited for economical and technical reasons (splicing, multidropping), are likely to increase this range by at least one order of magnitude (150 Mbit/s).

Until recently baseband modulation techniques have mainly been used. Broadband techniques using CATV cable technology, however, are now emerging in new network designs and this will lead to the availability of very high bandwidths (300 MHz) which make possible the multiplexing (by frequency division) of a large number of independent channels over a single cable.

In local area networks, complex routing strategies and techniques may be avoided by the possibility of using simple, constrained topologies without otherwise compromising the network performance. Moreover, the availability of high bandwidths may possibly lead to simplified communication protocols. Nevertheless, current experience has shown that typical problems of long-haul networks such as flow and congestion control also appear in local area networks with characteristics which are often strongly dependent on the topology being used.

The role of structured architectures and protocol standardization in local area networks is often dismissed by rather limited perspectives that tend to favour their specific technological aspects, mainly the access (and control) to the transmission media. In this sense, control structures and lower level communication-oriented protocols are treated in the particular context of local area networks, while higher level user-oriented protocols are often considered as general problems at a later stage.

We think that a less restrictive approach is advantageous - local area networks, especially in research environments, create the conditions for layered architectural designs based on user requirements and for the development and assessment of higher level protocols. Moreover, it is now generally accepted that in

the near future local area networks will be oriented to the support of "super intelligent" terminals, which will allow a single user to have access to powerful local processing and storage facilities and additional network access to more general resources and shared information [24]. Hence if lessons are to be learned from earlier experiences, higher level protocols and standard host interfaces will be required if the availability and potential advantages of this kind of device are to be exploited.

1.4.3 The role of X.25

So far in the context of local area network architectures, X.25 has mainly been considered as a means of interconnection with remote public networks and indeed, X.25 and local area networks are, in many circumstances viewed as quite separate entities.

X.25, however, is simply a network access interface mainly suited for point-to-point communications (this is especially true for the data link layer), in the same way as other access methods (contention, token, etc) are currently being defined for local area networks [25].

It is felt, therefore, that the growing popularity of X.25, the expanding hardware support, the availability of X.25 software packages from a number of manufacturers, the offer of virtual circuit and datagram services and the expected standardization of transport protocols tightly coupled with X.25, are strong reasons in favour of a wider perspective.

The main aim of this thesis, therefore, has been to

investigate the relevance and possible role of X.25 in a local area network environment.

CHAPTER 2

LOCAL AREA NETWORK ARCHITECTURES

2.1 Introduction

In the previous chapter we considered local area networks in the global context of networking and how they have evolved to meet a set of well defined requirements.

To achieve these requirements and making maximum use of the relatively smaller distances involved a range of entirely new solutions have been considered for local area networks, impossible or at least not efficient in a long-haul network. Simplicity becomes a central aspect and this has a strong impact on the topologies and control structures which have been proposed. Some alternative topologies and their control procedures are now described.

2.2 Some alternative topologies

Three basic topological organizations have been considered as candidates for local area networks - the star, the ring and the bus. Essentially, they may be distinguished either by the way the interconnected systems are interfaced to the transmission media or, more importantly, by how the access to the transmission media is controlled.

Current proposals for local area network standards, such as the IEEE 802 local area network reference model [25], particularly suited for ring and bus networks, support a layered architecture as shown in Fig. 2.1. A correspondence between the three lower layers of this model and the data link and physical layers of the OSI reference model may be readily established; the media access control layer and its protocols are particularly relevant in such architectures as will be discussed.

A complete assessment of the merits of the different topologies cannot be separated from the technologies they use and the profound impact of technological evolution and innovation. A brief overview therefore of each type of system will now be presented.

2.2.1 Star network

The star network is a completely centralized network with all the network functions concentrated in a central node, as shown in Fig. 2.2.

Access to the communication media is effectively a point-to-point communication controlled either by the central node (polling) or shared between the central node and each attached system over the corresponding link. The control mechanisms are thus similar to those used for connecting a host computer to a switching node in a long-haul store-and-forward network and hence most of the known experience and results in those networks and in particular the use of standard access protocols and interfaces (X.25, etc) are directly applicable to the star network.

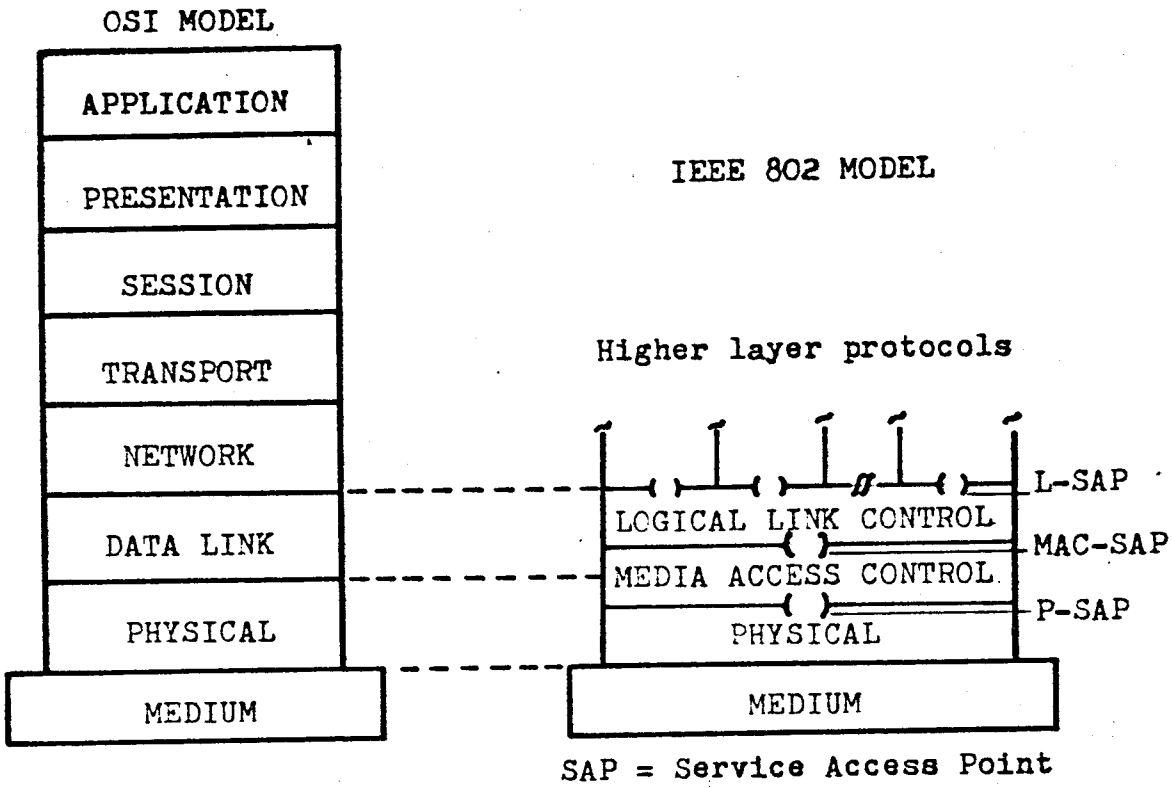


Fig. 2.1 - The IEEE 802 reference model

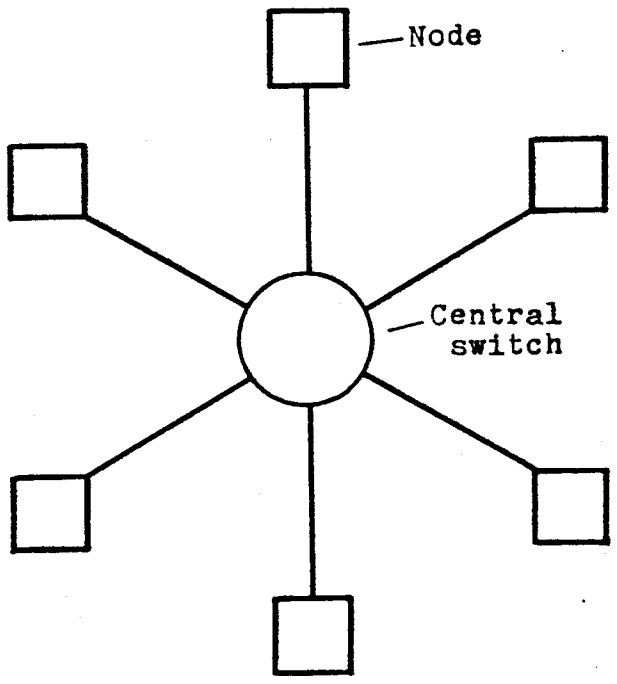


Fig. 2.2 - The star network

The simplest example is when a single central node is used as a packet switch serving a population of host computers and terminals. The central node then performs the necessary routing and management functions. This will be considered more fully later in this chapter.

2.2.2 Ring network

The ring network presents a very simple structure with the nodes connected by point-to-point links, forming a closed loop, as represented in Fig. 2.3.

Packets are routed around the ring, usually along unidirectional links, from source to destination and eventually back to the source. Packets are not normally stored but instead are simply delayed at intermediate nodes as they enter and leave the node.

Only a limited number of packets (depending on the total delay of the ring) may be in transit at any time. This requires the implementation of mechanisms to regulate the access to the media. Routing decisions are minimal and are associated with the removal of a packet from the ring. Until this happens (and this may be done by the destination or eventually by the source after a complete loop) packets are simply passed from one node over to the next. To achieve this function active repeaters are physically inserted into the ring.

Several control structures have been used or proposed in connection with ring networks [23,26], but the most commonly adopted are control token, message slot and register insertion.

Each will now be briefly described.

Control token

A well defined pattern in the data stream - the control token - is sequentially passed around the ring. The node in possession of the token may delay its passage by inserting one of its own packets prior to passing the token on to the next node. More than one packet may be in transit although with a single circulating token only one packet may be inserted into the ring at a time.

A ring supervisor is necessary to create the control token(s) and also for recovery purposes. Variable length messages are possible and usually a packet travels around the ring until it is removed by the source.

Message slot

A series of fixed length message slots (or "empty packets"), as many as the number of packets the ring can accomodate in transit, is continuously passed around the ring each carrying a full/empty mark. Any passing empty slot may be filled with a packet, then being marked as full; after removal of the packet the slot is again marked as empty. Initialization of the message slots is also required as a centralized function.

The Cambridge ring [27] is probably the most popular example of this access method. Packets (usually called "minipackets") are

of short fixed length of 38 bits. A minipacket contains 16 bits of user data and the remaining bits are used for addressing (8 bits each for source and destination) and general purpose control. In particular two are reserved as responses to be used by the destination. The current Cambridge packet format and bit usage are shown in Fig. 2.4.

To avoid a single node hogging the ring, packets are not removed at the destination but instead are returned to the source with the response bits changed. This constitutes a form of acknowledgement and the information carried in the response bits reports either acceptance , rejection, busy condition or eventually packet ignored by the destination. It also provides a convenient mechanism for sharing the usage of the ring between the connected nodes.

Protocols devised for the Cambridge ring, are based on a basic block protocol. As each minipacket contains only two bytes of data, a basic block (in fact corresponding to what is usually considered a packet) is made up of a sequence of minipackets: a header minipacket, a routing minipacket, up to 1024 user data minipackets and finally a higher-level checksum minipacket.

Register insertion

This is a completely distributed control mechanism. Each node uses a shift-register into which it loads a local packet (when the media is idle) and then delays any incoming packets by effectively inserting the shift-register into the ring. The shift-register is then progressively removed from the ring when

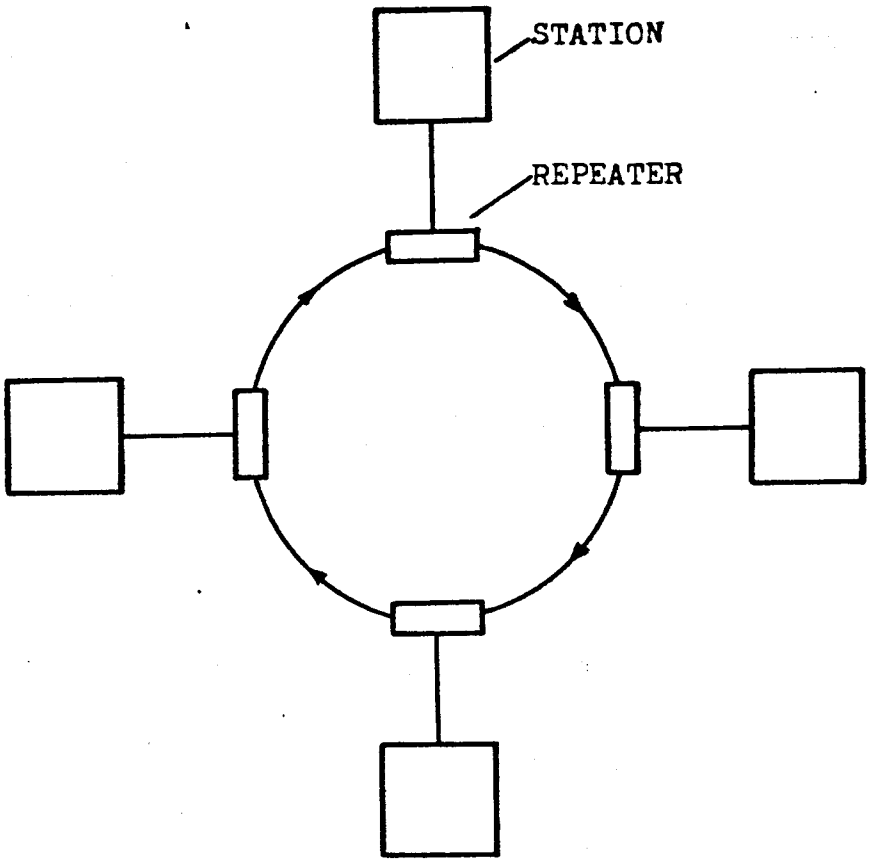


Fig. 2.3 - The ring network

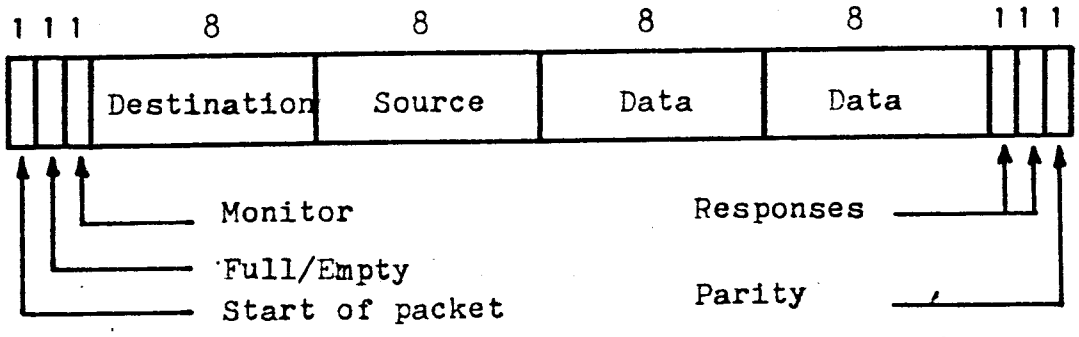


Fig. 2.4 - The Cambridge ring packet format

the link returns to the idle state or when the inserted packet returns to the source node. The total delay of the ring is therefore variable and depends on the buffer delay inserted at each node.

This method has been used in the Distributed Loop Computer Network [28]. It uses a bit oriented protocol in many aspects similar to SDLC/HDLC. Packets (variable length) are removed at the destination and a short acknowledgement packet is generated if requested.

2.2.3 Bus network

The bus network is a multiaccess link, with all nodes sharing a single common highway or bus as shown in Fig. 2.5.

Parallel buses have been used for many years as the communications facility within a single computer. In local area networks, however, because of the physical separation of connected nodes a serial bus is normally used. The most common transmission medium is coaxial cable to which the attached systems are simply tapped by means of a passive connector.

Each packet is in fact broadcast and although intended for some destination(s) it is made virtually available to any "listening" node in the network. The packet is only subject to propagation delays and no mechanism is required to enforce its removal from the media; it is simply copied by each node that recognizes the packet as addressed to itself.

Clearly only one packet can be successfully transmitted at a time thus requiring special contention mechanisms to share the

access to the bus. Several mechanisms have either been proposed or implemented [29] and usually fall into one of three categories: fixed assignment (pre-allocation of bandwidth), random access and polling.

2.2.3.1 Baseband bus

The essential characteristic of all random access contention techniques is that several stations may attempt to access the media simultaneously in which case a "collision" is said to occur and, if transmission continues, this will result in the corruption of the affected packets.

Several schemes have been implemented concerning the control of transmission, the detection of collisions and the scheduling of retransmissions. Some alternatives are now considered.

Aloha

This scheme was first implemented in the Aloha packet radio network [30]. In its pure version each station may transmit at any time, even if another transmission is in progress. Collisions, and more generally, packet losses, are detected by the absence of an acknowledgement after a time-out period. Retransmission delays are randomized to avoid repeated collisions. The maximum attainable throughput has been shown to be 18.6 %.

A slotted version, requiring each station to synchronize its

transmission to the beginning of a time slot (which in turn enforces complete transmission overlap and achieves a higher maximum efficiency - the double of pure Aloha) has also been implemented.

Carrier Sense Multiple Access (CSMA)

Significant improvement in performance is achieved if a station listens to the media before transmitting thus considerably reducing (but not eliminating) the probability of a collision. If the media is sensed to be idle, transmission may take place, otherwise transmission is delayed and rescheduled according to a pre-defined algorithm. These include non persistent, p-persistent and corresponding slotted versions.

Collisions are detected as in the previous scheme, and retransmission, after a time-out, is scheduled according to the particular method adopted.

Carrier Sense Multiple Access with Collision Detection (CSMA-CD)

This scheme constitutes a refinement of the basic CSMA since a station listens to the media both before and while transmitting. This is the mechanism first implemented in Ethernet [31] and is by far the most frequently used in current or proposed local area baseband bus networks.

As before, a station is allowed to transmit if the media is sensed idle. The transmitting station, however, listens to the

media during the critical collision interval at the start of the transmission and , if its transmission is garbled assumes a collision has occured. The collision is then enforced to ensure the collision is detected by the other affected station(s) and the transmission is then aborted. To reduce the probability of new collisions, retransmissions are scheduled at random intervals and in Ethernet a binary exponential back off algorithm is used.

The protocols at the data link layer are very simple. A datagram type facility is usually implemented; corrupted packets, other than collisions, are detected by CRC codes and discarded, and recovery is defaulted to a higher protocol level. A typical Ethernet configuration and packet format are shown in Fig. 2.6.

High throughputs may be achieved (above 90 %) but are critically dependent on packet size, number of stations, round trip delay and data rates. The current Ethernet specification [32] fits into the proposed IEEE 802 standard.

2.2.3.2 Broadband bus

Broadband techniques utilizing the total bandwidth in the range of 30 to 350 MHz available with a CATV coaxial cable offers a transmission facility with considerable flexibility. Two modes of operation are normally used:

(a) the whole bandwidth is partitioned into a forward (lower) band for transmission and a return (higher) band for reception. The cable must therefore be terminated by a "head-end" the main function of which is to act as a

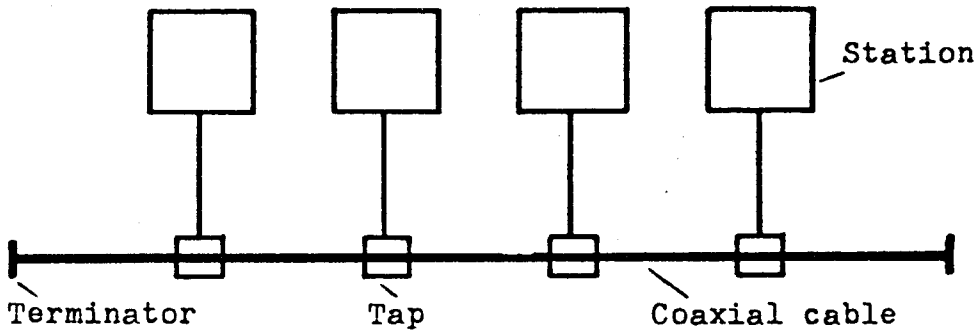
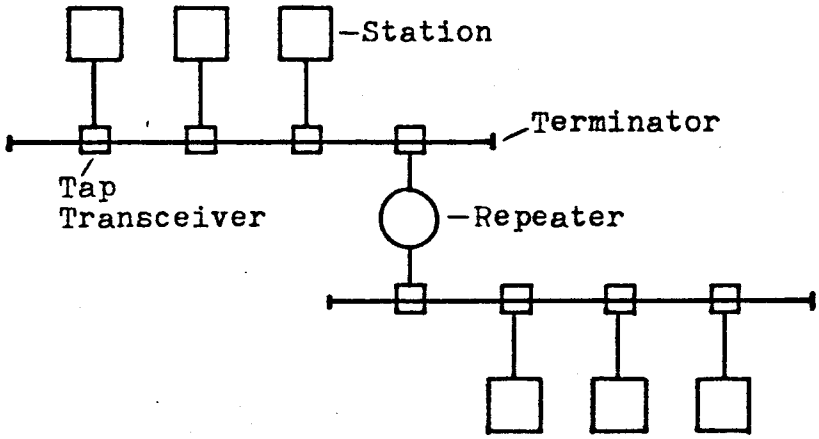


Fig. 2.5 - The bus network



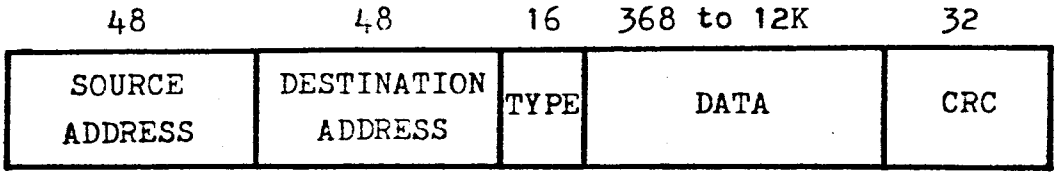


Fig. 2;6 - Typical Ethernet configuration and packet format

frequency translator from the lower to the higher band. This technique is compatible with the use of single or double cable;

(b) the whole bandwidth is used both for forward and return bands. This requires the use of two cables: one for transmission, the other for reception. A simple crossover point separates both cables without the need for frequency translation.

In either case the available bandwidth (total or split) is partitioned by frequency division multiplexing to allow for several different communication channels. A wide variety of configurations is then possible and indeed may coexist within the same cable by reserving particular subbands for each purpose. For example one proposed system - the WangNet [33]- utilizes a multiplicity of different channels including:

(a) one or more Ethernet like CSMA-CD channels;

(b) a number of dedicated frequency channels (emulating leased lines) to support point-to-point or multipoint communications using fixed frequency modems;

(c) a number of switched frequency channels, dynamically allocated by a polling central data switch, supporting temporary full-duplex connections and using frequency agile modems.

More discussion about broadband bus systems will be reserved until later in this thesis.

2.3 Proposed Architecture

Rather than discussing and emphasizing the advantages and disadvantages of different network architectures based on general assumptions, it seems more important to assess the attributes of each architecture in respect to meeting the requirements of a particular network. Such factors as the number and type of devices to be connected, their physical distribution and separation, initial configuration and future expansion, etc, are clearly important.

The continuing evolution of networking technology (broadband techniques, optical fibres, satellite communications, etc) and the substantial support currently being given to different types of local network architectures suggest that the trend will probably be to provide convenient forms of network interconnection. In addition it is likely that a requirement for most local networks will be for a connection to one or more public data networks and in this respect the tendency will be to adhere to standard protocols and interfaces, such as X.25.

2.3.1 Network requirements

When considering the implementation of a local area network it is essential to first identify both the specific requirements of the network and also the potential advantages that can be achieved by subscribers using such a network. Although these will vary from one network to another, the requirements of a typical local area network may be summarized as in Fig. 2.7.

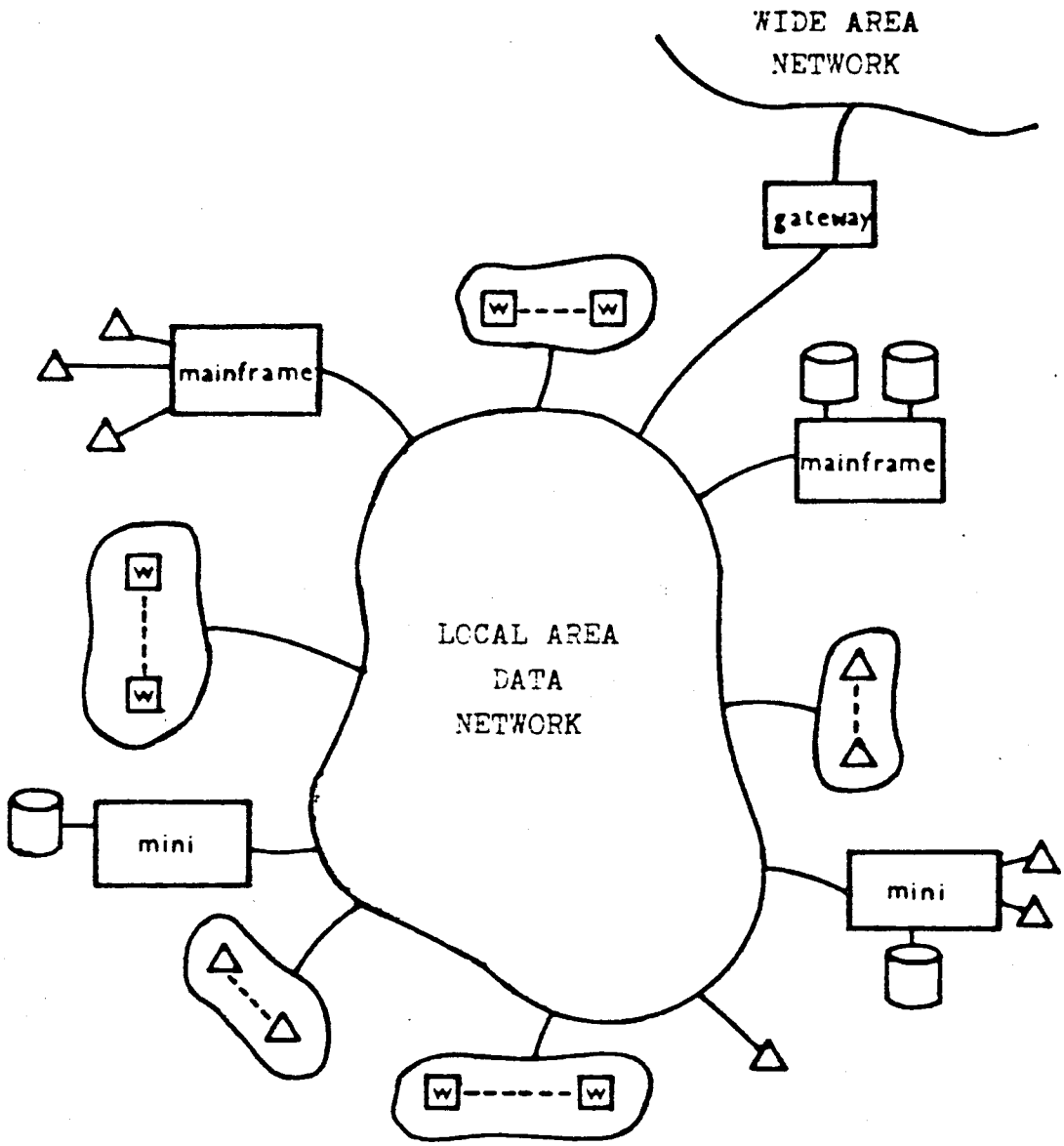


Fig. 2.7 - Typical network requirements

As a result of the rapid expansion of the computing resources found within an organization, instead of an establishment handling all its data processing functions in a single large computer, it is now likely that a number of separate machines will be used. A primary requirement for many networks, therefore, is to provide a communications facility for a number of large mainframes and/or minicomputers either to allow communications between them or to allow other network devices access to them in order to share a specific resource or files of information.

Secondly, although many of the larger machines will each have its own community of user devices connected to it (terminals, printers, etc), the provision of a switched connection between a distributed community of unintelligent terminals and a number of large machines may still be an important requirement for many networks.

Thirdly, the existence of a large number of small single user systems (word processors, intelligent work-stations, etc) is likely to require a communications facility between such systems and other network resources either within a localized community of similar systems or on a network-wide basis.

Finally, there is possibly a requirement for a communications facility between some local devices and one or more external networks, for example for a mainframe to submit or receive transactions from a remote centre.

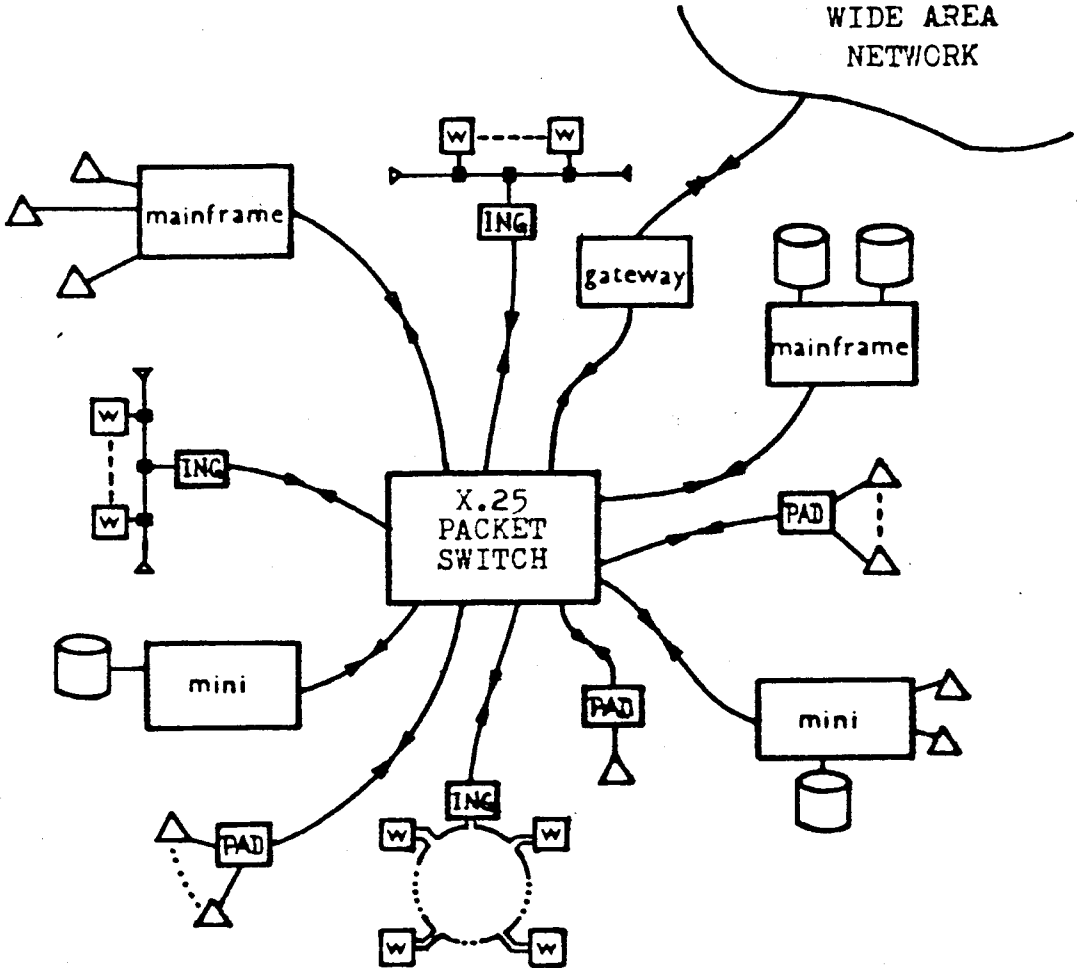
2.3.2 Network topology

From the previous section it can be concluded that the requirements of a local area network are extremely varied and thus it seems unlikely that these will be satisfied by a single type of network. Moreover, although it may be relatively simple to write a new communications package for a small single user computer system and integrate this into its operating system, the same does not apply for a large mainframe or minicomputer, thus creating perhaps the most demanding requirements for the network designer.

The proposed topology, therefore, is a hybrid network as shown in Fig. 2.8. The central component of the network is a an X.25 based modular Packet Switching Exchange. A range of additional components must be included to meet the other network requirements.

Unintelligent character mode devices (VDU's, etc) are integrated into the network by means of Packet Assembly/Dissassembly devices (PAD's) which appear to the exchange as multiple call packet-mode devices.

The networking requirements for the possible large number of local communities of small microprocessor based systems are usually of two main types: the first to access some locally held shared resource (printer, file server, etc) and the second to access a network-wide resource (shared data base, for example). Since the first is likely to dominate, it is proposed that these requirements will be met by the provision of a number of separate local networks (based for example on Ethernet or the Cambridge ring), while a small internetwork gateway, possibly situated



ING = Inter Network Gateway
PAD = Packet Assembler / Disassembler

Fig. 2.8 - Proposed network topology

within the local community of users, will handle any network-wide communications.

Finally, communications with other external networks (public or private) is handled through a separate internetwork gateway and to meet these requirements the exchange includes an internetwork gateway interface.

Further discussion concerning the integration of these components into the network is reserved for Chapter 8.

2.3.3 Packet Switching Exchange

As was outlined in Chapter 1, the internationally agreed standard protocol now accepted for access to public packet switching networks is X.25 and indeed rapidly growing support is evolving, as many computer manufacturers now offer X.25 packages as part of their supported software library. The exchange thus presents a standard interface on each access link based on the X.25 protocol and hence maximum use of existing and proposed manufacturers hardware and software can be made.

The function of the central exchange in the proposed architecture is similar to that of a single node in a wide area public data network. Normally, however, a node in a public network is comprised of a single large minicomputer with extensive storage and software to perform such functions as call charging and network management [34]. Clearly, many of these functions are either not required or may be considerably reduced in a local area network and hence it was decided to investigate the implementation of a Packet Switching Exchange that was more

suitable for use in a local network environment.

It was decided that no call charging software would be included and also, to minimize the number of packet buffers (and hence storage) required, it was decided to reduce the amount of packet processing (X.25 packet level) performed in the exchange.

Also, in order to produce an exchange of low initial cost but capable of incremental growth, it was proposed to make maximum use of microprocessor technology.

An overview of X.25 and the design and implementation details of the exchange and its associated performance measurements are now presented in the following chapters of this thesis.

CHAPTER 3

RATIONALE AND SPECIFICATION OF X.25

3.1 Introduction

The draft recommendation of X.25 [17], was first approved in 1976. It defines " The interface between Data Terminal Equipment (DTE) and Data Circuit-terminating Equipment (DCE) for terminals operating in the packet mode on public data networks ".

Revisions and ammendements [35,36] were later issued in response to incompatibilities and undefined aspects in the original recommendation and also to clarify different interpretations [37,38] and to introduce additional facilities. In this chapter an overview of X.25 will be presented and in addition a discussion of the more controversial aspects of the protocol, especially those concerned with possible deadlocks and ill-defined states will be given.

3.1.1 Layered design

X.25 is structured into three protocol layers or levels:

level 1 (the physical layer) defines the "physical, electrical, functional and procedural characteristics to establish, maintain and disconnect the physical link between the

DTE and the DCE". These characteristics must be in accordance with recommendation X.21 or X.21 bis [39];

level 2 (the link layer) defines the "Link Access Procedures for data interchange across the link between the DTE and the DCE". Level 2 is a subset of the ISO High level Data Link Control protocol HDLC [40] and currently two link access procedures are supported;

level 3 (the packet layer) defines the "packet format and control procedures for the exchange of packets containing control information and user data between the DTE and the DCE".

The three X.25 layers are equivalent to the three lower layers in the OSI reference model and correspond to the physical layer, the data link layer and the network layer respectively.

3.1.2 X.25 as a local interface

The recommendation states that X.25 defines only the interface between a host computer or a packet mode device and the communications subnetwork. That is, it does not enforce any particular topology or mode of operation within the subnetwork and hence any host computer supporting X.25 should be able to communicate using any subnetwork that provides an X.25 service at the access interface.

It can be concluded, therefore, that it is only possible to attribute local significance to the information and control procedures exchanged across each DTE/DCE interface. This is

particularly obvious at the physical and link levels where point-to-point links are independently controlled. At the packet level, however, although the X.25 procedures only notionally apply at the access interfaces, the management of a virtual circuit also has end-to-end effects. This has led to misinterpretations and misuse of X.25 and will be discussed in more detail later.

3.2 Data link layer (level 2)

The data link layer provides a means of reliably exchanging blocks of data (frames) between a DTE and a DCE using an established and operational physical link.

3.2.1 Frame structure

The format of a frame is shown in Fig. 3.1 (a). As can be seen, each frame is bounded by two flags where a flag is a special character or bit sequence (Ø1111110). Transparency to the flag pattern within a frame is achieved by the technique known as bit stuffing or zero bit insertion. Whenever five contiguous ones are detected during transmission, a zero is inserted to ensure a flag is not transmitted. Similarly, during reception, the zero is removed.

Since in X.25 only point-to-point links are supported, the address field is not used as a real address identifier and instead is used simply to distinguish between two different frame types (commands and responses).

The control field identifies the frame type as shown in Fig. 3.1 (b) and provides additional parameters for specific control purposes:

(a) Information frames are used for information transfer and are characterized by a variable Information field, transparently handled on behalf of the packet layer;

(b) Supervisory frames are used for specific control purposes that will be discussed in connection with the acknowledgment, error and flow control mechanisms;

(c) Unnumbered frames are used for synchronization purposes (set mode, reset, terminate mode).

The P/F bit has a rather limited function in X.25 and is exclusively used during time-out recovery.

Information frames are sequentially numbered (modulo M) by means of a Send Sequence Number N(S). Information and supervisory frames also carry a Receive Sequence Number N(R) which indicates the next expected Information frame in the reverse direction.

The Frame Check Sequence (FCS) field (two bytes) provides a cyclic redundancy checksum based on the generator polynomial X**16+X**12+X**5+1. Its error detection properties are discussed, for example, in [41,42].

8	8	8	N	16	8
FLAG	ADDRESS	CONTROL	INFORMATION	FCS	FLAG

(a)

			1	2	3	4	5	6	7	P
TYPE	COMMAND	RESPONSE	CONTROL FIELD							
Information transfer	I (information)		0	N(S)			P		N(R)	
Supervisory	RR (receive ready)	RR (receive ready)	1	0	0	0	P/F		N(R)	
	RNR (receive not ready)	RNR (receive not ready)	1	0	1	0	P/F		N(R)	
	REJ (reject)	REJ (reject)	1	0	0	1	P/F		N(R)	
Unnumbered	SARM (set asynchronous response mode)	DM (disconnect mode)	1	1	1	1	P/F	0	0	0
	SABM (set asynchronous balanced mode)		1	1	1	1	P	1	0	0
	DISC (disconnect)		1	1	0	0	P	0	1	0
		UA (unnumbered acknowledgment)	1	1	0	0	F	1	1	0
		CMDR (command reject)	1	1	1	0	F	0	0	1

(b)

Fig. 3.1 - Frame structure and types

3.2.2 Link control

Before any Information frame may be transferred across the DTE/DCE interface it is first necessary to set-up the logical link, that is, it is necessary to create the protocol context that will keep both stations synchronized.

Two link access procedures are currently defined: LAP and LAPB corresponding to the Asynchronous Response Mode (ARM) and to the Asynchronous Balanced Mode (ABM), respectively.

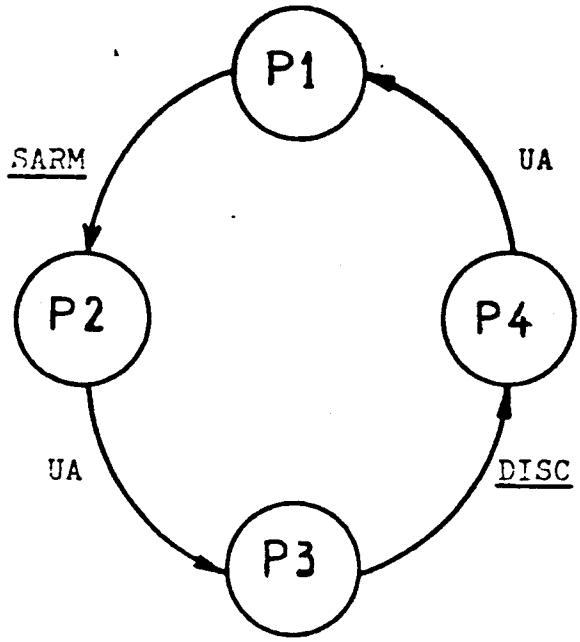
In LAP, the DTE and the DCE each implement a Primary and a Secondary station. Each Primary is responsible for the control of the remote Secondary (activation, termination, error control and recovery). For this purpose each Primary issues commands and expects responses from the remote Secondary.

In LAPB both the DTE and the DCE constitute a combined station. Control of the link is shared by both combined stations each of which transmits and receives commands and responses.

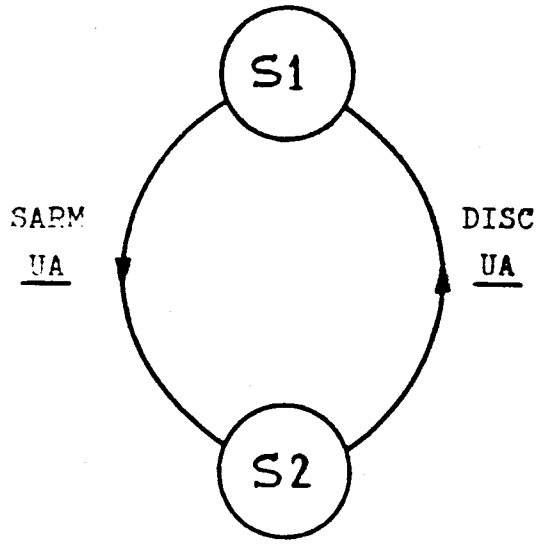
The differences between LAP and LAPB are mainly related to the set-up and disconnection of a link.

In LAP each Primary station may initiate the set-up procedure, thus initializing the link in the corresponding Primary-Secondary direction. Initialization of the opposite direction must follow; this requires a close-coupling between Primary and Secondary functions. A link is set-up by the Primary sending a SARM and the Secondary acknowledging its receipt by returning a UA. Once a link has been set-up, information transfer may take place until a logical disconnection (DISC) is initiated by either station (Primary). A state transition diagram is shown in Fig. 3.2. There are in fact some potential problems with

PRIMARY



SECONDARY



P1 = Primary disconnected
P2 = Primary connection
P3 = Primary connected
P4 = Primary disconnection

S1 = Secondary disconnected
S2 = Secondary connected

Note: frame transmissions are underlined

Fig. 3.2 - Link state transition diagram

this set-up procedure but these are discussed in Appendix A and a more complete state description will be presented in Chapter 5.

In LAPB either combined station may initiate the set-up procedure (SABM), thus initializing the link in both directions and the same applies to the link disconnection (DISC).

3.2.3 Information transfer

Error and flow control mechanisms are used during the information transfer phase firstly to ensure that Information frames are reliably exchanged across the serial link without losses or duplicates and secondly to control the rate at which Information frames are exchanged. This is accomplished using the following schemes.

3.2.3.1 Acknowledgment mechanism

Any acknowledgment mechanism must be based on the assignment of a unique identifier to each Information frame so that transmitter and receiver may unambiguously refer to the same frame [18]. Clearly, if it was possible to use an infinite set of identifiers, no contention would arise and any losses or duplicates would be easily detected and recovered. As this is not possible, however, it is necessary to cyclicly reuse the same finite set of identifiers.

In HDLC, Information frames are sequentially numbered, usually modulo M=8. For this purpose the Primary keeps a Send

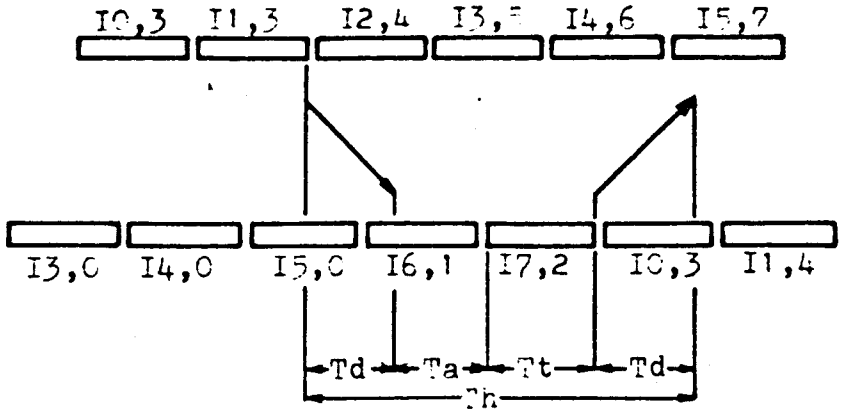
State Variable V(S), which refers to the next Information frame to be transmitted (N(S)) and is incremented after each transmission. Thus, in the absence of errors, Information frames are expected by the Secondary strictly in sequence. In addition, the Secondary keeps a Receive State Variable V(R) which refers to the next in-sequence Information frame to be received. Then, if the N(S) in the received frame equals V(R), the frame is accepted and V(R) is incremented. A receive sequence number N(R) is then updated to the current V(R) and returned to the transmitter in a RR supervisory frame or "piggybacked" in an Information frame. This N(R) is then checked against the N(S)'s in the currently unacknowledged Information frames and any correctly acknowledged frame (with N(S) up to N(R)-1) is then released. Examples of this mechanism are shown in Fig. 3.3.

The maximum number of unacknowledged Information frames that may be outstanding is naturally limited (K<=M-1) but this number is further determined by the error and flow control strategies. A window is considered as the ordered set [W0,W1[of Information frames authorized to cross the interface where W=(W1-W0) mod M is the window size.

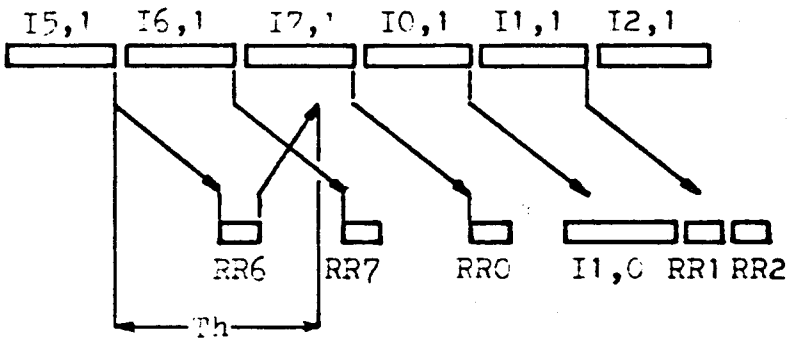
3.2.3.2 Error control

Error control mechanisms are required to detect and recover from error situations, which usually fall into one of four categories:

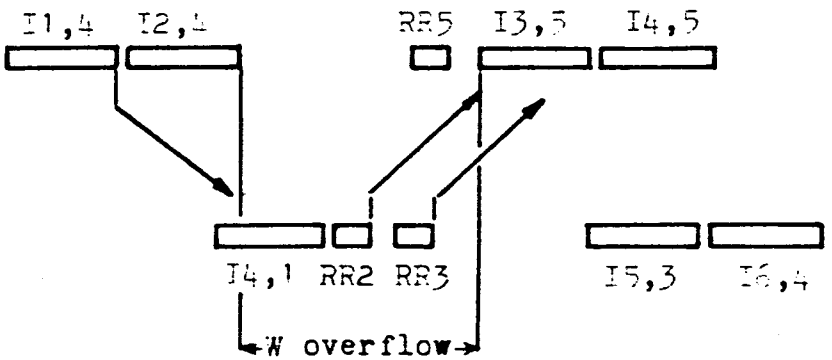
(a) corruption of frames in the transmission media (noise,



(a) piggyback



(b) no piggyback



(c) window overflow

Td = processing delay
Ta = acknowledgment delay
Tt = transmission time
Th = holding time

Im,n = Information N(S),N(R)
RRn = Receiver Ready N(R)

Fig. 3.3 - Examples of the acknowledgment mechanism

crosstalk);

(b) loss or corruption of frames due to undetected errors in the serial I/O interfaces as a result of overruns, underruns, aborts, etc;

(c) loss of frames due to the lack of buffer resources or the application of flow control;

(d) errors induced by recovery actions.

All the attempts to recover from errors are based on their detection, their type and on some state information, that is the context in which they occur.

The integrity of frames in the transmission media is protected by a CRC code. Frames in error are simply ignored and they are handled by the recovery mechanism as a frame loss.

A continuous Automatic Request (ARQ) mechanism is supported by the link level which means that several unacknowledged Information frames, up to the window, are allowed outstanding. Out of sequence frames detected by the receiver are handled as follows:

The sender goes back to the first missing frame as determined by a REJ supervisory frame from the receiver and retransmits from this point as in the normal way. The receiver then ignores all Information frames until it receives an Information frame with N(S)=V(R). This guarantees that no duplicates arise and only one REJ frame is allowed outstanding before recovery is complete.

There are situations, however, when this retransmission mechanism cannot be triggered:

(a) the missing Information frame(s) is (are) the last in a sequence of Information frames and so any losses are not detected by the receiver;

(b) the REJ is lost;

(c) the first retransmitted frame is again lost.

An additional retransmission mechanism based on the absence of an acknowledgment for some outstanding Information frame(s) following a specified time-out interval is then utilized. In this case it is the sender that triggers the retransmission and, in doing so, duplicates may be generated. This can happen if either an acknowledgment is lost or delayed greater than the time-out interval.

The global recovery strategy depends, therefore, on the way duplicates are distinguished from other out-of-sequence frames and more generally on the possible interactions between these two recovery mechanisms.

3.2.3.3 Retransmission strategy

One way of distinguishing duplicates from other out-of-sequence frames is by reducing the window ($W <= M/2$), to account for different instantaneous views of the window by the sender and the receiver [43].

However, X.25 does not support this mechanism; instead, out-of-sequence frames are globally considered and, additionally, the time-out recovery relies on the use of checkpoint retransmission using the P/F bit. This raises an important question: how many

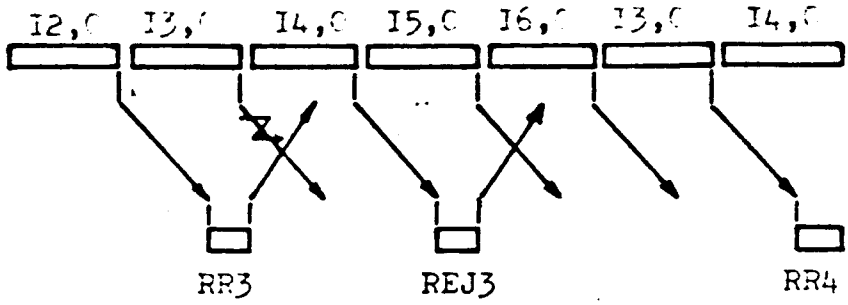
Information frames should be retransmitted during time-out recovery? We recall that the REJ recovery allows unrestricted (up to the window) retransmission of unacknowledged frames.

No controversy arises concerning the first retransmitted Information frame; it must be the first unacknowledged and must carry the Poll bit set,thus creating a useful reference for the Primary and the Secondary.

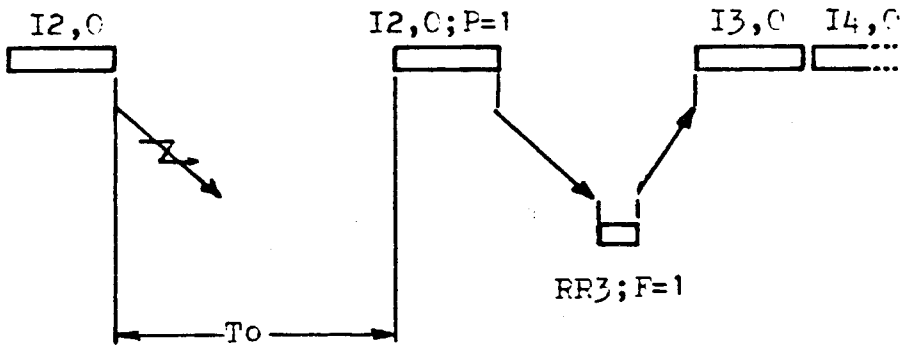
This requires, however, that no interference with this mechanism is induced. This could be the case if the Primary retransmitted other Information frames (necessarily with P=Ø) immediately following. The first Information frame is protected by the Poll bit but, if the remaining Information frames are duplicates, they simply appear to the receiver as out of sequence and hence the normal recovery mechanism will result in a REJ frame being sent. When received by the Primary, the REJ frame may thus interfere with the current retransmission (in the same way as two outstanding REJ frames) and hence induce unnecessary retransmissions with possible looping effects.

One solution to avoid this problem is to retransmit only one Information frame during time-out recovery; this ensures, through the P/F exchange, that normal transmission is unambiguously defaulted to the correct Information frame. Thus, with this scheme, use of a RR or REJ frame (with F=1) is virtually equivalent, since only the N(R) is relevant at this stage.

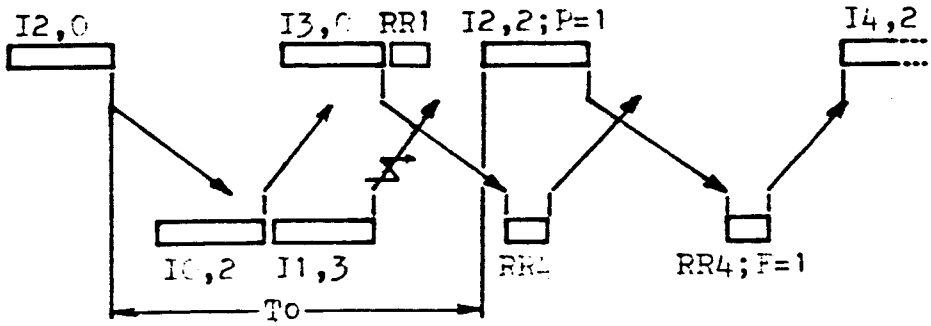
Examples of the REJ and time-out recovery mechanisms are shown in Fig. 3.4.



(a) REJ recovery



(b) Time-out recovery (no duplicate)



(c) Time-out recovery (duplicate)

= frame lost To = time-out

Fig. 3.4 - Examples of the recovery mechanisms

3.2.3.4 Link reset

In both recovery procedures so far considered no complete loss of synchronization occurs and normal operation resumes after recovery.

A complete loss of synchronization may occur, however, due to other errors or cumulative effects of abnormal delays. Under these circumstances recovery becomes impossible simply by retransmission and reinitialization is then necessary. This is achieved by the transmission of a reset frame (SARM in LAP or SABM in LAPB) and by so doing the corresponding Primary and Secondary stations (or both combined stations) return to a known and agreed state before normal information transfer resumes.

Two procedures may apply:

(a) the reset may be initiated by the Primary (SARM) when receiving a response with unknown code, containing an invalid Information field, carrying an invalid N(R) or with an invalid F bit. A received N(R) is considered invalid if it "acknowledges" an Information frame not outstanding at the time;

(b) the Secondary may request a reset by the Primary (by transmitting CMDR) when receiving a command with unknown code or containing an invalid Information field (invalid length, for example).

During reset, Information frames are suspended in the corresponding direction(s) and no acknowledgment is acted upon until the reset is complete (UA received). Because Information

frames are not released before an acknowledgment is received, Information frames outstanding and unacknowledged at the time will be retransmitted later. It can be concluded that:

(a) due to the creation of a new protocol context, duplicates created as a result of a link reset are not detectable within level 2 of X.25. Implications of this to the higher layers must therefore be considered;

(b) failures in the acknowledgment mechanism not immediately detected and recovered by a reset may lead to invalid release of Information frames, thus creating an irrecoverable loss at the link layer.

3.2.3.5 Flow control

X.25 does not enforce any particular flow control mechanism. Rather it provides the means to implement the flow control protocol, at the network access interface.

Flow control at the DCE cannot be separated from the control strategy implemented within the communications subnetwork (where it is mainly concerned with the control of congestion) and the way it is exercized at different protocol layers cannot be viewed independently.

At this point,however, only the means by which a DCE (or DTE) may control the flow of frames across the interface will be considered.

HDLC uses a combination of two mechanisms:

(a) Stop-and-Go - the Secondary when entering a busy condition requests transmission of Information frames from the Primary to be stopped by sending a RNR frame and transmission of Information frames will resume only after receipt of an explicit authorization. This is achieved by sending a RR or REJ frame, the latter applying if correctly received Information frames were deliberately discarded. Two aspects are relevant in this mechanism. First, the degree of anticipation in issuing a RNR; it is therefore necessary to decide whether to accept or discard any currently outstanding Information frames. Secondly, since an explicit authorization is required to resume transmission, a recovery mechanism is necessary to account for the possible loss of the corresponding RR/REJ frame;

(b) Credit - the Primary is granted permission by the Secondary to transmit a number of Information frames. This mechanism also raises two problems. First, if a credit mechanism is exclusively based on a pre-allocation of buffer resources, inefficient use of resources may result. Secondly, credits must be protected by an acknowledgment mechanism to account for the possible loss or duplication of credits.

In HDLC, to avoid the last problem the credit scheme is combined with the acknowledgment mechanism. Any acknowledgment (except while the busy condition is not cleared) carries an implicit credit (window size) relatively to Information frames

flowing in the reverse direction, thus updating the lower window edge (window rotation).

The credit scheme is used under normal conditions but, when required, a RNR is issued to stop credit. Then, depending on the anticipation factor, outstanding Information frames may be accepted or discarded (some or even all). This does not give rise to particular difficulties, however, as losses are recovered by retransmission but, to avoid possible blocking of the Primary (due to loss or delay of the RR/REJ), the Primary is allowed to transmit or retransmit the Information frame pointed to by the RNR. A potential problem with this mechanism is also discussed in Appendix A.

3.3 Packet layer (level 3)

A packet is the Information field contained in each Information frame transferred across a DTE/DCE interface and transparently handled by the data link layer. Whilst in the data link layer frames are handled in a pure point-to-point way, at the packet layer the true nature of the network communications begins to appear since, in general, a packet also contains an identification of the particular data stream to which the packet belongs.

For this reason a virtual circuit is first established between two communicating DTE's possibly involving the DTE/DCE interfaces and also the subnetwork. The Packet layer of X.25, however, only provides the mechanisms to control the virtual circuits across the DTE/DCE interfaces.

3.3.1 Virtual Circuit Service

Multiple virtual circuits are supported in a single link by the use of a number of logical channels. The establishment of a virtual circuit (or call) requires the assignment of a logical channel at both DTE/DCE interfaces with the communications subnetwork and also the provision of a subnetwork virtual circuit service characterized by ordered delivery of packets without losses or duplicates. This is normally achieved by assigning different logical channels on successive links from source node to destination node during the set-up of the virtual circuit. Packets belonging to the same virtual call are then routed along the same path.

The relevant information exchanged between two end systems is conveyed in Data packets and additional packets are used to accomplish such functions as the management of virtual circuits, error and flow control, etc. The general packet format and a list of the packet types used is shown in Fig. 3.5 and their functions are now discussed.

3.3.1.1 Call set-up

Set-up of a virtual circuit is initiated by a DTE sending a Call Request packet. A logical channel in the DTE/DCE interface is then locally assigned. After the packet has been routed across the subnetwork, the reverse process takes place at the remote DTE/DCE interface: the Incoming Call is sent to the destination DTE over an available logical channel and the virtual circuit is

General format identifier	Logical channel group number
Logical channel number	
Packet type identifier	
Address, control or data fields (variable length depending on the packet type)	

Packet Type		8	7	6	5	4	3	2	1
From DCE to DTE	From DTE to DCE	Packet identifier							
Incoming Call	Call Request	0	0	0	0	1	0	1	1
Call Connected	Call Accepted	0	0	0	0	1	1	1	1
Clear Indication	Clear Request	0	0	0	1	0	0	1	1
DCE Clear Confirmation	DTE Clear Confirmation	0	0	0	1	0	1	1	1
DCE Data	DTE Data	P(R)			M		P(S)		0
DCE Interrupt	DTE Interrupt	0	0	1	0	0	0	1	1
DCE Interrupt Confirmation	DTE Interrupt Confirmation	0	0	1	0	0	1	1	1
DCE RR	DTE RR	P(R)			0	0	0	0	1
DCE RNR	DTE RNR	P(R)			0	0	1	0	1
	DTE REJ	P(R)			0	1	0	0	1
Reset Indication	Reset Request	0	0	0	1	1	0	1	1
DCE Reset Confirmation	DTE Reset Confirmation	0	0	0	1	1	1	1	1
Restart Indication	Restart Request	1	1	1	1	1	0	1	1
DCE Restart Confirmation	DTE Restart Confirmation	1	1	1	1	1	1	1	1
Diagnostic		1	1	1	1	0	0	0	1

Fig. 3.5 - Packet format and types

established by the called DTE returning a Call Accepted/Call Connected packet. This is shown in Fig. 3.6 and, as can be seen, it is purely end-to-end. If a call cannot be established by the subnetwork or the called DTE does not wish to accept it, the call is cleared.

Contention for a logical channel may arise since the same logical channel may be simultaneously assigned to both a Call Request issued by the DTE and an Incoming Call issued by the DCE. To resolve this, X.25 gives priority to the Call Request and, to reduce the probability of a collision occuring, the DTE and the DCE assign logical channel numbers in reverse order.

3.3.1.2 Data transfer

Once a virtual circuit has been established, Data packets may flow in both directions and are subject to error and flow control procedures. Data packets are sequentially numbered, usually modulo 8 and are acknowledged either by piggybacking the acknowledgment in a Data packet or by returning a supervisory (RR/RNR) packet.

Send and receive sequence numbers in Data and supervisory packets are not necessarily preserved along the virtual circuit. This is the case, for example, if different packet lengths are used at the DTE/DCE interfaces and within the subnetwork, thus requiring fragmentation and/or reassembly by the subnetwork or as a result of the application of flow control.

3.3.1.3 Call clearance

Normally a call is cleared by either DTE once the data transfer is complete. Clear Request/Clear Indication packets are used for this purpose and they traverse the complete network in order that the whole virtual circuit may be released. Clear Confirmation may either be performed locally (hop-by-hop) and the released logical channels made immediately available or it may be performed end-to-end, as shown in Fig. 3.7.

In addition, a call may be cleared by the subnetwork for a variety of reasons; for example, congestion, invalid call, called DTE busy, logical channels not available, protocol error, etc. If a clear is initiated by the subnetwork, however, Clear Indication packets are returned in both directions to the connected DTE's indicating the reason for clearing.

From the previous description, a state transition diagram for the call set-up and clearing phases applying to a logical channel across the DTE/DCE interface may be derived, and this is represented in Fig. 3.8.

3.3.2 Error control

Most errors fall into the category of protocol errors since they result in a loss of protocol context between the DTE and the DCE; for instance, the receipt of a Call Request while the channel is in the data transfer state.

The normal action in such conditions is the clearance of the

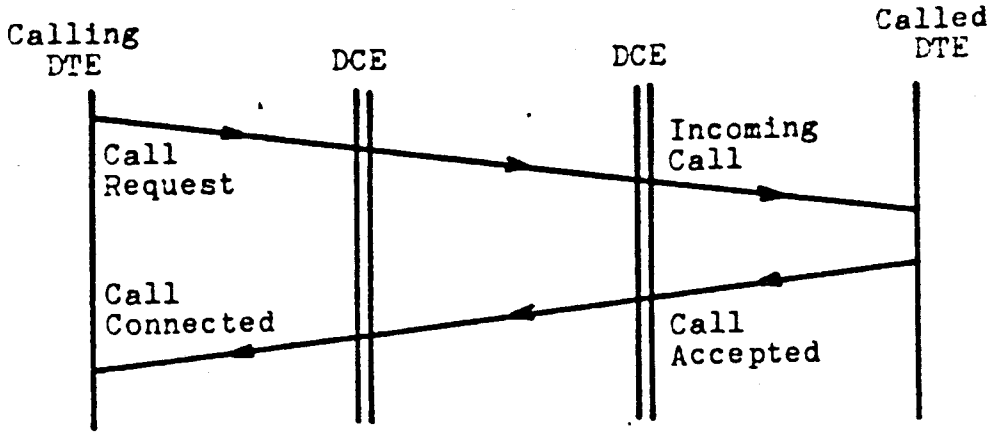


Fig. 3.6 - Call set-up

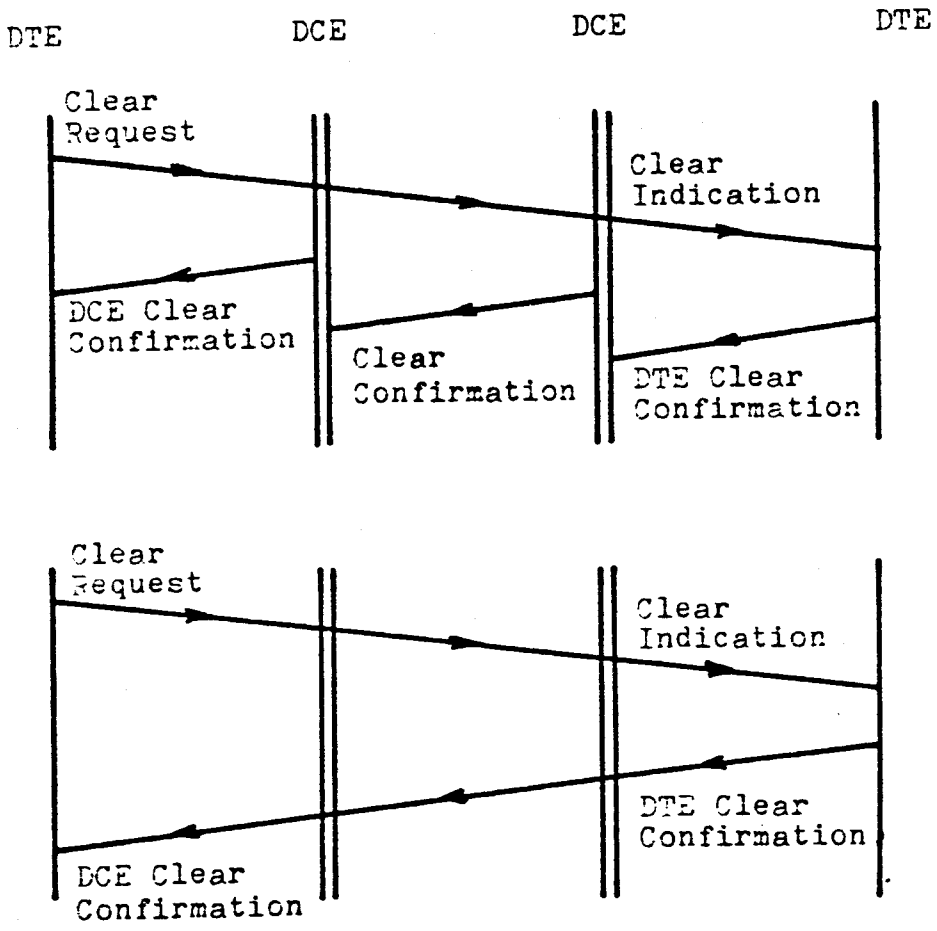
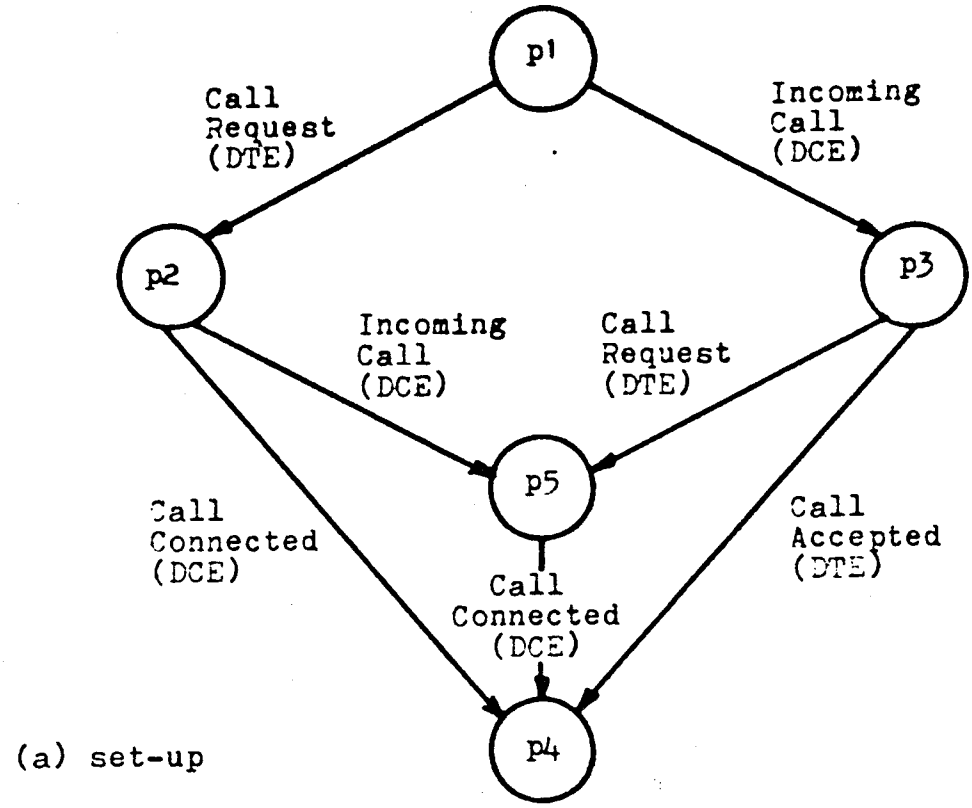
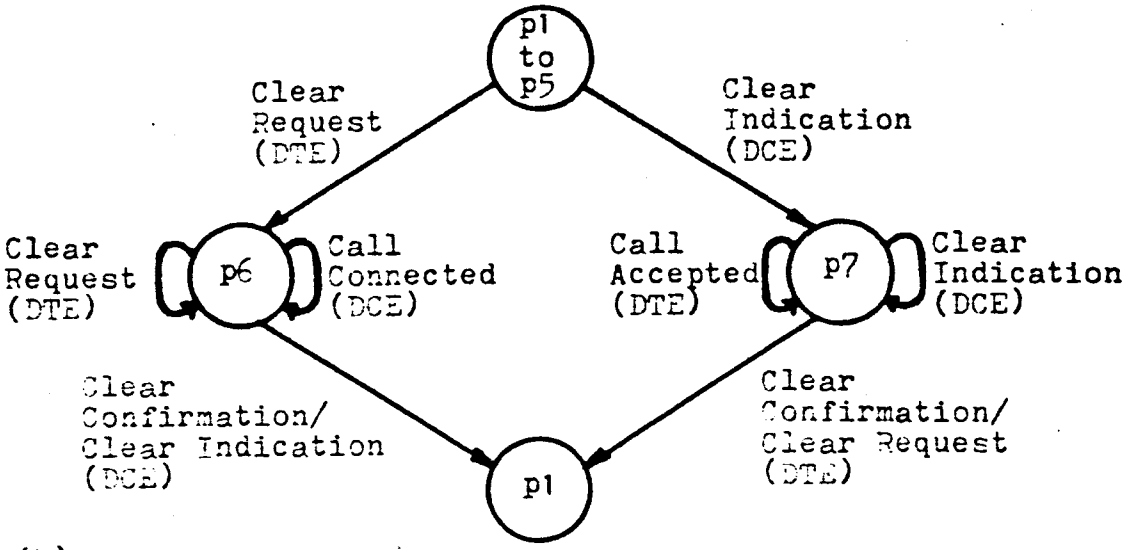


Fig. 3.7 - Call clearing



(a) set-up



(b) clearing

p1 = ready
p2 = DTE waiting
p3 = DCE waiting
p4 = data transfer
p5 = call collision
p6 = DTE clear request
p7 = DCE clear indication

Fig. 3.8 - X.25 logical channel state transition

call in order to restore the protocol context and to avoid the propagation of errors across the network.

In addition, X.25 provides a global clearance of all calls (virtual circuits) by means of Restart packets across the DTE/DCE interface. In the absence of broadcast facilities, a DCE has therefore to generate Clear Indication packets, across the subnetwork, for each call currently in progress at the DTE/DCE interface.

3.3.2.1 Time-out recovery

The need to implement a time-out mechanism arises from the fact that, although improbable due to the protection of level 2, expected confirmation packets may be missing or at least delayed beyond a reasonable time interval.

Normally, time-outs are recovered by retransmission. The use of retransmission at the packet level, however, at least for some type of packets, can give rise to additional problems:

(a) retransmission of a Call Request/Incoming Call may induce a protocol error, thus resulting in the call being cleared;

(b) retransmission of Indication packets (Restart, Clear, Reset) may result in long looping conditions [44,45,46].

The first problem may be solved simply by clearing the call and eventually retrying.

As far as the second problem is concerned, it is now accepted that Indication packets should not be retransmitted and

instead Diagnostic packets should be optionally sent by the DCE [47]. These are simply used by the DTE for logging purposes.

X.25 does not directly address the time-out problems associated with the absence or delay of an acknowledgment for Data packets. As will be discussed this may be induced by a particular flow control strategy and has the effect of blocking the flow of Data packets.

It is accepted, however, that no retransmission of Data packets is supported by a DTE, under any circumstances [38]. This has a significant effect on the reset and flow control strategies as follows.

3.3.2.2 Call reset

Reset at the packet layer (Reset Request/Reset Indication) is used as an error recovery mechanism applying only during the data transfer phase and has similar objectives to those achieved by a link level reset.

However, Data or supervisory packets associated with the affected virtual circuit are removed and, because no retransmission is supported, a call reset can therefore give rise to packet losses.

X.25 specifies the use of a Reset for two main reasons. Firstly, when the receive sequence number, P(R), is received in error, thus indicating a Data packet is not outstanding and secondly, when the send sequence number, P(S) is out of sequence (due to losses or duplicates) or violates the window. In addition, a call may be reset (following a time-out) to reopen

the window when a transmitter is blocked due to a lack of credit.

As has been described, duplicates at the packet layer may be created by a mechanism external to the packet layer (link reset). It is impossible, therefore, even by window limitation to discriminate duplicates from other errors; signalling of a link reset does not solve the problem as the receiver cannot ascertain the boundaries of the eventual duplicate(s). In general it can be stated that a call reset may eliminate duplicates but induce losses; hence, errors will necessarily propagate above level 3, thus requiring the implementation of a recovery mechanism at the transport layer, on an end-to-end basis and possibly at the message level [48,49].

Several alternative strategies, certainly based on assumptions about the type and frequency of errors seem possible:

(a) no use of reset, except probably if the transmitter is blocked, thus defaulting all error recovery to the transport layer (mainly duplicates);

(b) use of reset only for window violations and P(R) errors, simply discarding out-of-sequence packets as duplicates;

(c) reset in all the above error situations.

In any case, and for generality, both losses and duplicates should be detected and recovered at the transport layer.

3.3.3 Flow control

Flow control within the packet layer is performed on a virtual circuit basis and uses the same tools as the data link layer; basically a window (credit) mechanism combined with the possible use of RNR packets.

When a RNR packet is issued, Data packets may still be outstanding in the reverse direction owing to their previous credit. Once accepted for processing by the packet layer, however, valid Data packets should not be discarded even if a RNR has been used, because no retransmission is supported.

The question then arises is the flow control effective, or even necessary, at the packet layer? The degree of effectiveness is influenced by the buffer management policy; also, because the use of RNR has always a delayed effect, the degree of anticipation in its use becomes an important factor.

Two alternatives are usually considered:

(a) buffers are reserved in advance on a virtual circuit basis according to the window sizes and a credit is never given in excess of the available resources. In such cases when a RNR is used, enough buffers still exist to accomodate the possible outstanding Data packets and a RR packet must then be used to reopen the window as soon as enough buffers are granted. This dedication of resources is usually highly inefficient and requires a large number of buffers, unless the number of active virtual circuits is severely reduced or if, as in TYMENET, flow control is character oriented [50]. With this scheme, however, congestion problems are

efficiently avoided;

(b) buffers are dynamically assigned on demand. Usually credit is given in excess of the currently available resources (overselling) on the founded expectation that they will exist when requested. Naturally, more complex flow control is required to avoid congestion and to provide an efficient and fair sharing of resources between virtual circuits, thus greatly improving the performance. The anticipation factor when using RNR becomes more important since its degree of dynamism may influence the frequency of use of level 2 flow control.

An alternative to the use of RNR and in some cases offering the advantages of a more dynamic and efficient window rotation is simply to delay acknowledgments. Thus, previous credit is fully used by the transmitter, but then gradually restrained while the window remains only partially opened.

CHAPTER 4

PACKET SWITCHING EXCHANGE - HARDWARE DESIGN

4.1 Design philosophy

The design aims of a Packet Switching Exchange for use in a local area environment can be summarized as follows:

modular construction - since the required number of exchange lines will vary considerably from one network to another, the exchange must be modular and capable of incremental growth;

standard access protocol - the exchange must present a standard X.25 access protocol and hence be compatible with manufacturers networking hardware and software;

good reliability - failure of a single unit should not cause a catastrophic failure but rather a degradation of the services offered and hence overall performance;

self monitoring - the current status of the exchange should be continuously monitored and, for diagnostic purposes, faulty components readily determined;

low cost - to be competitive with other local networking solutions, the exchange should have only a modest cost per line connected.

In an attempt to meet these aims, the design adopted is

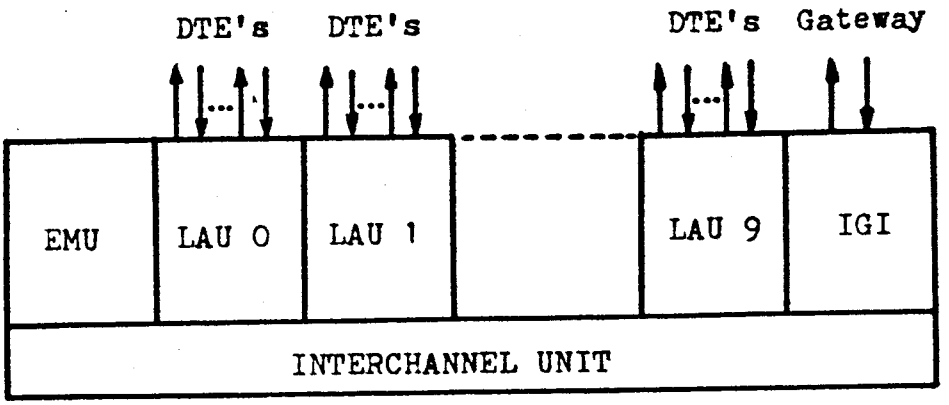
shown in Fig. 4.1. It is comprised of a readily extendable number of Link Access Units (LAU's), which perform the basic call processing functions of the exchange, an Inter-network Gateway Interface (IGI), similar to a Link Access Unit, to provide for communications with other wide-area networks, and an Exchange Management Unit (EMU) for monitoring and diagnostic purposes. The above units are then interconnected by means of an autonomous Interchannel Unit or parallel bus. A description of each is now given.

4.2 Link Access Unit

Each Link Access Unit can support a number of full-duplex links and each presents a standard X.25 interface. The design of the unit is based on a Z80 microprocessor [51] - several reasons determined this choice:

(a) support of a wide family of peripheral controllers (SIO, PIO, CTC, etc);
(b) powerful vector interrupt and daisy chain interrupt priority structure;
(c) availability of Z80 based microprocessor development facilities allowing extensive program debugging and real-time simulation of the exchange.

The organization of a Link Access Unit is shown in Fig. 4.2. Each unit has its own internal bus to which is connected a Z80 microprocessor, a readily varied quantity of PROM and RAM



EMU = Exchange Management Unit
LAU = Link Access Unit
IGI = Internetwork Gateway Interface
DTE = Data Terminal Equipment (Mainframe,mini,PAD,ING)

Fig. 4.1 - Exchange design

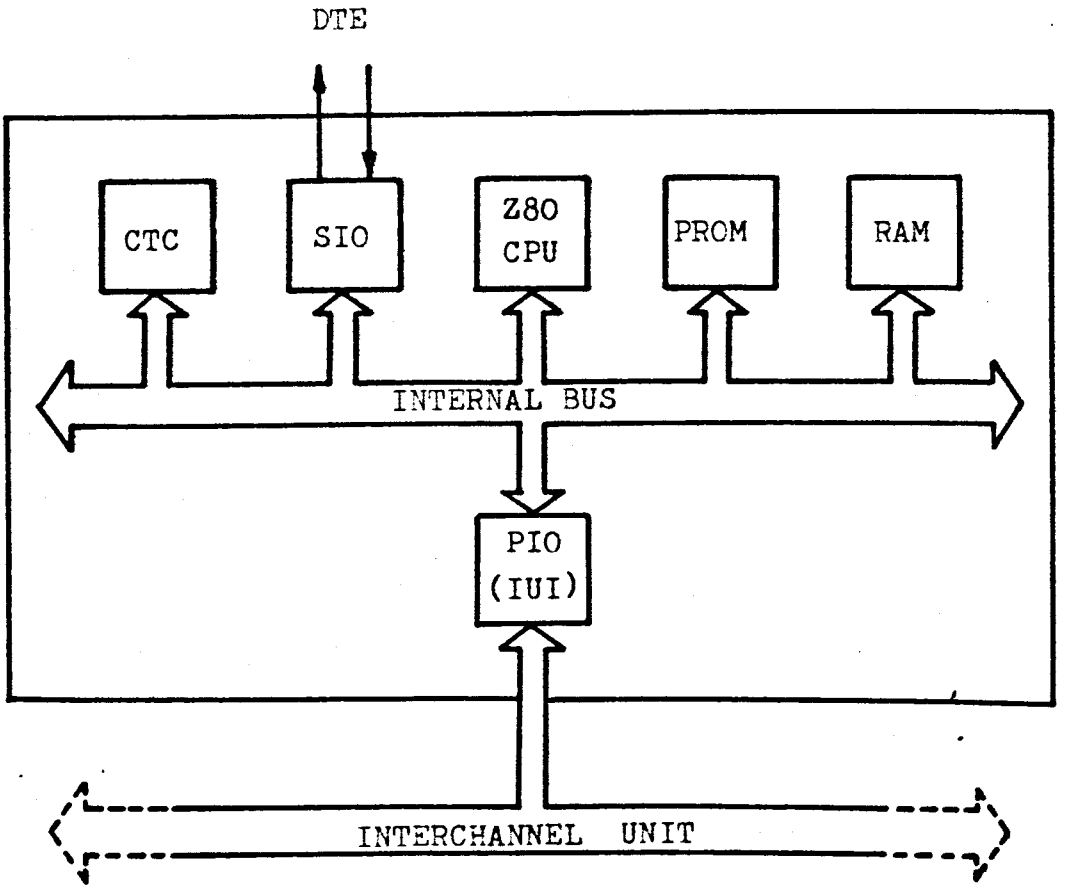


Fig. 4.2 - Schematic of a Link Access Unit

(currently 8K and 4K bytes respectively) to hold the X.25 control program and its associated tables and packet buffers, a counter/timer, one or more data link controller cards and an Interunit Interface (IUI) card which forms the interface between the unit and the Interchannel bus.

4.2.1 Serial link interface

The interface to the serial link is based on a Z80 SIO controller [52], which may be programmed to support a wide variety of communication protocols including the X.25 level 2 protocol HDLC. The Z80 SIO provides the basic framing and error detection features of the X.25 link access procedures but additional hardware and/or software must be used to perform the other functions.

Currently all link transfers are interrupt driven - mode 2 (vectored) is used with the SIO internal interrupt priority and vector modification logic enabled; as will be discussed in Chapter 7 link baud rates up to 50 Kbaud may be supported.

X.25 does not provide any field to deal with the length of the frame (packet) being transmitted or received. Hence, since length information is required during the frame/packet processing and transfer from one Link Access Unit to another, it was decided to include, for exclusive use within the exchange, an additional byte at the header of the frame to indicate the frame length.

On reception, the frame length is computed on a byte basis (RBYTE count) and, when the end of frame is detected, this is

inserted in the length byte preceeding the frame. When transmitting, however, the length byte is not transmitted but is simply used together with a TBYTE count in order to allow correct termination of the frame.

Clearly, the maximum buffer size requirement must be determined according to the maximum packet size supported across the interface. Hence, since the maximum data field currently implemented is 128 (default value in X.25) a total of 136 bytes is used. This includes the length byte, 2 bytes of level 2 header, 3 bytes of level 3 header, 128 bytes of user data, 1 byte to temporarily accomodate CRC1 prior to detecting the end of frame and 1 byte to deal with a possible overflow condition.

4.2.1.1 Frame transmission

Two interrupt service routines are used in the transmission process: the transmitter interrupt routine which is invoked on a byte by byte basis and the external/status interrupt routine which deals with the underrun/end of message situation. The Z80 SIO has a number of limitations in this respect and in particular:

(a) the number of bytes yet to transfer must be externally updated;

(b) CRC is automatically inserted by the SIO either following an underrun or at the end of a frame;

(c) abort transmission must be programmed after an underrun is detected;

(d) following transmission of an abort, the SIO defaults to the transmission of flags but no interrupt is then generated.

Moreover, the SIO must be programmed to insert the CRC, thus also allowing generation of the corresponding external/status interrupt. This must be done following the first byte loaded into the SIO since doing it in advance induces premature generation of the CRC ahead of the frame (underrun); doing it later will not protect against an early underrun condition and flags will be inserted prematurely.

Frames awaiting transmission are stored in a Link Output Queue (LOQ) and successive transmissions must be separated by at least a single flag.

End of transmission

The normal sequence of events during transmission is shown in Fig. 4.3. Up to the last byte, all transmitter interrupts (raised when the SIO transmit buffer becomes empty) are simply served by loading the next byte into the SIO. No underrun will occur if interrupts are served within the interval TØ of a byte transmission (Ø.8 ms at 9.6 Kbit/s).

Correct termination of a transmission requires:

(a) identification of a normal end of frame condition (TINTn), thus allowing the subsequent external/status interrupt to recognize a valid end of frame;

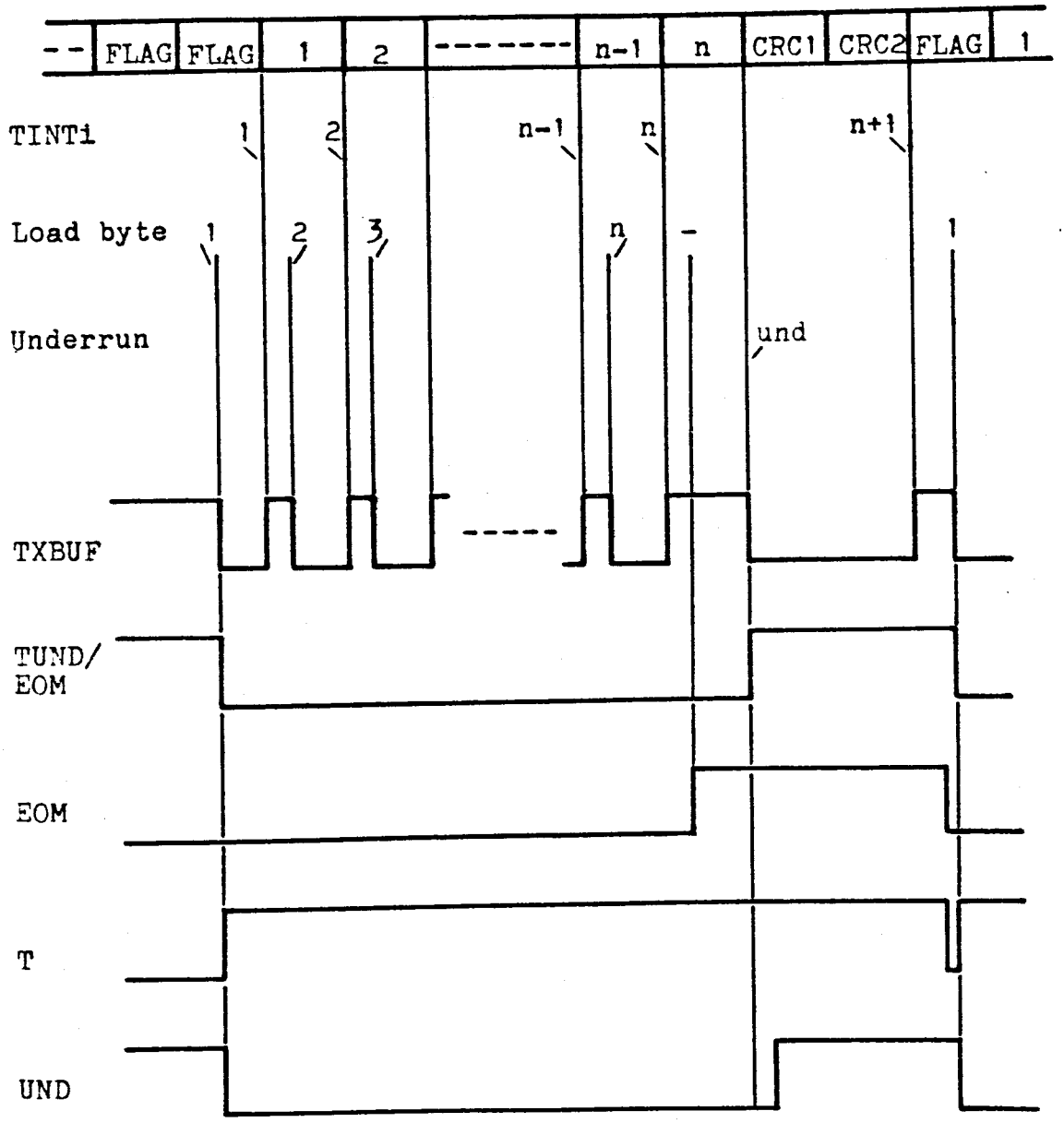


Fig. 4.3 - Frame transmission

(b) identification of the real end-of-frame transmission (TINTn+1), protected by the trailing flag, thus allowing the next transmission to begin correctly (otherwise no flag separates two consecutive frames if the first byte of the next frame is prematurely loaded into the SIO). Hence, TINTn+1 may be used to directly trigger the next transmission, thus avoiding transmission gaps.

TINTn and TINTn+1 may be distinguished from previous interrupts by the current TBYTE count, signifying the number of bytes yet to transmit. The SIO also provides a way of distinguishing between TINTn and TINTn+1, by means of two flags TXBUF (TXBUF=1 when the SIO transmit buffer is empty) and TUND/EOM (TUND/EOM=1 when the underrun is detected).

Three external flags are used - T to allow synchronization with externally initiated frame transmissions (T=1 during transmission), EOF to signal a valid end of frame (EOF=1) and UND to allow identification of underrun/end of message within the external/status interrupt (UND=Ø, TUND/EOM=1) and its service (UND=1).

However,the above sequence may be disturbed for two main reasons:

(a) in the case of cumulative delays and taking into consideration the interrupt priorities, the external/status interrupt may be served at the end and TINTn and TINTn+1 may eventually be merged into a single interrupt;

(b) it may be impossible to distinguish between TINTn and TINTn+1, using the internal SIO flags, if the setting of

the TUND/EOM flag is delayed due to status latching (although no risk exists of missing the interrupt as only a single transition is involved).

A general solution taking both aspects into consideration was adopted. It is based on a synchronization between the transmitter and external/status interrupts, by means of the UND and TXBUF flags. Normally, the real end of transmission will be detected by a transmitter interrupt following the external/status interrupt (if UND=1) or eventually by the external/status interrupt (if TXBUF empty). In the latter case, either TINTn+1 had previously been served (while UND=Ø) or had just been raised and then becomes or is made irrelevant (for example, begin to load next frame or issue a " Reset transmit interrupt pending " command).

Underrun

The occurrence of an underrun condition in the middle of a frame may now be easily detected (EOF=Ø) when the external/status interrupt is served. An abort must be generated, thus destroying whatever is in transmission (CRC, delayed byte, flag). It will be effective if programmed before the next flag is completely transmitted - and so a minimum protection of three bytes (2.4 ms at 9.6 Kbit/s) is then guaranteed.

The real problem concerns correct retransmission of the aborted frame - the SIO does not provide an interrupt when transmission defaults to flags, but rather when the abort is

issued, which is irrelevant since it cannot be used to begin correct retransmission of the aborted frame (no flag would be generated and therefore the frame would be lost).

One possible way around this problem is by means of a timing mechanism. However, another solution, independent of the baud rate and offering additional advantages was adopted:

(a) the transmitter interrupt pending is reset (SIO command);

(b) parameters concerning the aborted frame are updated, preparing its retransmission;

(c) the internal TUND/EOM latch is reset (SIO command), forcing the generation of two extra CRC bytes and the subsequent external/status and transmitter interrupts, and EOF is set (to prevent another abort being generated), as shown in Fig. 4.4.

This sequence of events is now similar to a normal end of transmission (EOF=1) immediately followed by retransmission of the aborted frame.

Two objectives are met with these procedures: firstly, no overheads are introduced in the transmitter interrupt routine, as retransmission of the aborted frame follows the pattern of a new transmission and secondly, the two extra CRC bytes inserted (and ignored by the receiver which is then in the hunt phase) give an additional and useful protection to the receiver when dealing with the abort detection.

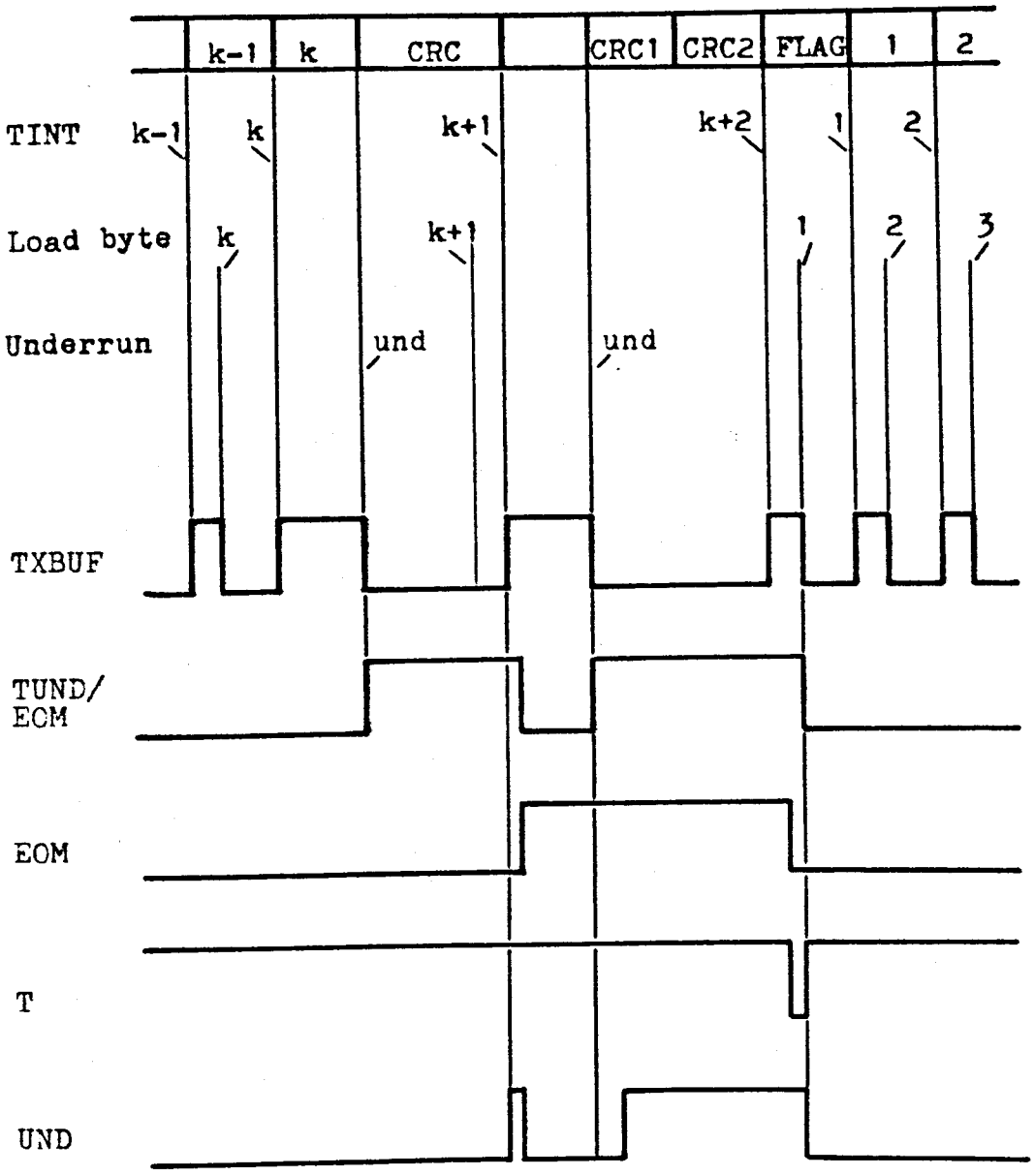


Fig. 4.4 - Underrun/ abort transmission

4.2.1.2 Frame reception

Three routines are relevant to the receiver process: the receiver interrupt routine which is called on a byte by byte basis, the special receive interrupt routine which is invoked on the occurrence of an end of frame or overrun and the external/status interrupt routine which deals with the flag synchronization (hunt phase) and the abort/idle detection.

The receiver process produces two new problems: firstly, a buffer must be granted before or at the moment a new frame begins to be assembled by the SIO and transferred to the receiver FIFO (triple buffered) and secondly, the receiver has no "a priori" knowledge about the length of the received frame nor does it have any control over the interval between successive frames (in the limit this could be a single flag).

Solving the length problem is trivial since, by providing a buffer with the defined maximum size (136 bytes), an overflow condition may be easily detected. If this happens, additional received bytes must be read (to clear the interrupt) but not transferred to memory. The frame will not be discarded, however, since the link level will follow the protocol rules concerning invalid frame length.

The other problem is more complex and, in fact, several solutions are possible. Due to the asynchronous nature of the interrupts and the uncontrolled gap between successive frames, any buffer request/grant mechanism not synchronized with the receiver interrupt routine may fail to grant a buffer during the critical interval when this is required. Any solution of this type must be rejected (certainly lost frames would trigger

frequent higher level recovery mechanisms, with a negative impact on the overall performance).

Consideration of a private pool would require additional data transfers inside the Link Access Unit and a reservation scheme, naturally depending on average requirements, could become critical in unusual conditions (succession of short frames) or lead to inefficient buffer utilization.

It was decided, therefore, to provide the receiver process with direct access to a buffer pool, only restricted for flow control purposes. This is discussed in Chapters 5 and 6.

In the absence of overrun errors or frame abort, end of reception is straightforward and is by means of End of Frame and CRC validation (flagged by the special receive interrupt). A valid frame (including the inserted length byte) is simply transferred to the Link Input Queue for further processing but if the frame has been corrupted, it is ignored and the buffer is returned to the pool.

Start of reception

Because the receiver has free access to the pool and also to minimize the number of pool accesses, a buffer is accessed from the pool at the beginning of reception rather than at the end of a previous reception. This requires identification of the first received byte, which may be done using the RBYTE count, also necessary for frame length and overflow detection, and signifying the available space in the current buffer.

An RC flag is also used to signal a reception in progress

(RC=1). Therefore RBYTE=0 refers either to the start of reception (RC=0) or overflow (RC=1). Reception may begin if a new buffer is granted (then RBYTE=buffer length and RC=1); otherwise, reception is abandoned and defaulted to the next frame (clear receiver FIFO and reenter hunt phase).

Overrun

Once an overrun occurs (and this only happens if a receiver interrupt cannot be served within the time corresponding to the transmission of four bytes - 3.2 ms at 9.6 Kbit/s), the frame is necessarily corrupted and, as in the case of a CRC error, it was decided to discard it since to ascertain the non corrupted bytes, for link control purposes, for example, would introduce high overheads.

Two solutions are possible - abandon reception and default to next frame reception (reenter hunt phase) or continue receiving the whole frame. The first solution seems attractive as it reduces time overheads following the overrun. However, it is possible that a new frame is already being assembled and present in the FIFO in which case it would be lost. Consequently, since an overrun is highly improbable, it was decided that the second alternative should be used. Moreover, by not issuing an Error Reset command (until the real end of the frame is detected) the internal SIO overrun flag is latched and may be used thereafter, thus generating special receive interrupts and avoiding additional overheads in the normal receiver interrupt routine.

Abort detection

Abort detection

An external/status interrupt occurs during flag synchronization (hunt phase) or abort detection. The SIO enters the hunt phase when programmed to do so or automatically following the detection of an abort.

When flag synchronization is achieved, the corresponding hunt interrupt could be used to report that the remote transmitter is active sending flags, thus preparing for frame reception. Not only is this not necessary, however, but may even induce errors. Correct reception of subsequent frames is independent of any action determined by the hunt interrupt, while a tight coupling between hunt and receiver interrupts could lead to misinterpretations due to the lower priority of the external/status interrupt.

Interaction with the abort interrupt is necessary but creates different kinds of problems. An abort/idle interrupt is irrelevant if no reception is in progress. If a reception is in progress, however, all actions must be taken at the commencement of abort detection (and not at the end) to minimize further interactions. This is distinguished by means of an external AB (abort) flag. However, the abort is not flagged (by the SIO) in the data stream (as for end of frame or overrun); frame reception is cancelled by the SIO until resynchronization is achieved but previous received bytes are not removed from the FIFO until read.

The problem is obvious - if an abort interrupt (lower priority) is eventually delayed until after the begining of a new reception, the header of the new frame will be wrongly assembled with the tail of the previous aborted frame and the new frame

will appear truncated.

Due to the early considered protection given by the transmitter (abort plus two CRC), ample time usually exists to serve the abort interrupt. In case of error, however, missing bytes in the header will probably induce protocol errors that will be detected at higher protocol levels.

4.2.2 The Counter/Timer

Two timing mechanisms are required: one to generate the SIO receiver/transmitter clocks and the other to allow implementation of the different time-out intervals required by the protocol.

Currently each SIO channel implements separate transmitter and receiver clocks. It is assumed that each station generates its own clock for transmission and that the receiver clock is extracted from the received data stream.

The requirements of the different processing levels within the protocol concerning the handling of time-out intervals are quite different. It was decided, therefore, to derive all time-out intervals from a basic time interval. This is currently every 100ms, when an interrupt is generated allowing the synchronization and update of all current active timers.

A Z80 CTC is used to generate the required timing signals. Channel 0 (timer mode) is used to generate the SIO transmitter clock and Channel 2 (timer mode) and Channel 3 (counter mode) are cascaded to generate the 100 ms interrupts.

4.2.3 Interunit Interface

The signals used to communicate with the Interchannel Unit are described in Fig. 4.5 (a). They are comprised of two 8 bit unidirectional parallel data paths D0-7 (one for input, one for output) and associated strobes (WRP, RDP) for the transfer of blocks of data between the Link Access Unit and the Interchannel Unit buffers; four address lines AD0-3 and associated control signals (WRSB, RDDB) to specify the required destination address and to control packet transfers, and finally two control signals indicating the Interchannel Unit buffers status (SB, DB).

The Interunit Interface is based on a Z80 PIO controller. Port A is programmed in the bidirectional mode and port B in the control mode, thus allowing status monitoring and generation of address and control signals as shown in Fig. 4.5 (b). All four PIO handshake signals (RDY and STB) are used to control the transfer of packets, byte by byte, between the PIO and the Interchannel Unit. The send and receive mechanisms are now described.

4.2.3.1 Packet transmission

The handshake signals 'A RDY' and '$\overline{\text{A STB}}$' are used to control the transfer of a packet between the PIO output port and the Source Buffer within the Interchannel Unit.

The circuit used to generate the PIO and the Interchannel Unit strobe pulses is shown in Fig. 4.6 (a) and the sequence of events, represented in Fig. 4.6 (b), is as follows:



INTERUNIT INTERFACE SIGNALS

DØ-7	Data Bus (tri-state input / output) : unidirectional data buses to transfer data between the PIO and the Source and Destination Buffers within the Interchannel Unit
AØ-3	Address Bus (tri-state output) : destination LAU address
WRP	Data Strobe (output) : write PIO Output port
RDP	Data Strobe (output) : read PIO Input port
WRSB	Begin packet transfer (output) : used by the Interchannel Unit to initiate a packet transfer across the bus, after loading the Source Buffer
RDDB	Packet transfer complete (output) : used by the Interchannel Unit to reenable packet transfers to the destination channel, after unloading the Destination Buffer
SB	Source Buffer status, within the Interchannel Unit (input) ; Ø=full; 1=empty
DB	Destination Buffer status, within the Interchannel Unit (input) : Ø=empty; 1=full

PIO CONTROL PORT

7	6	5	4	3	2	1	Ø
SB	DB	RDDB	WRSB	A 3	A 2	A 1	A Ø

Fig. 4.5 - Definition of the Interunit Interface control signals

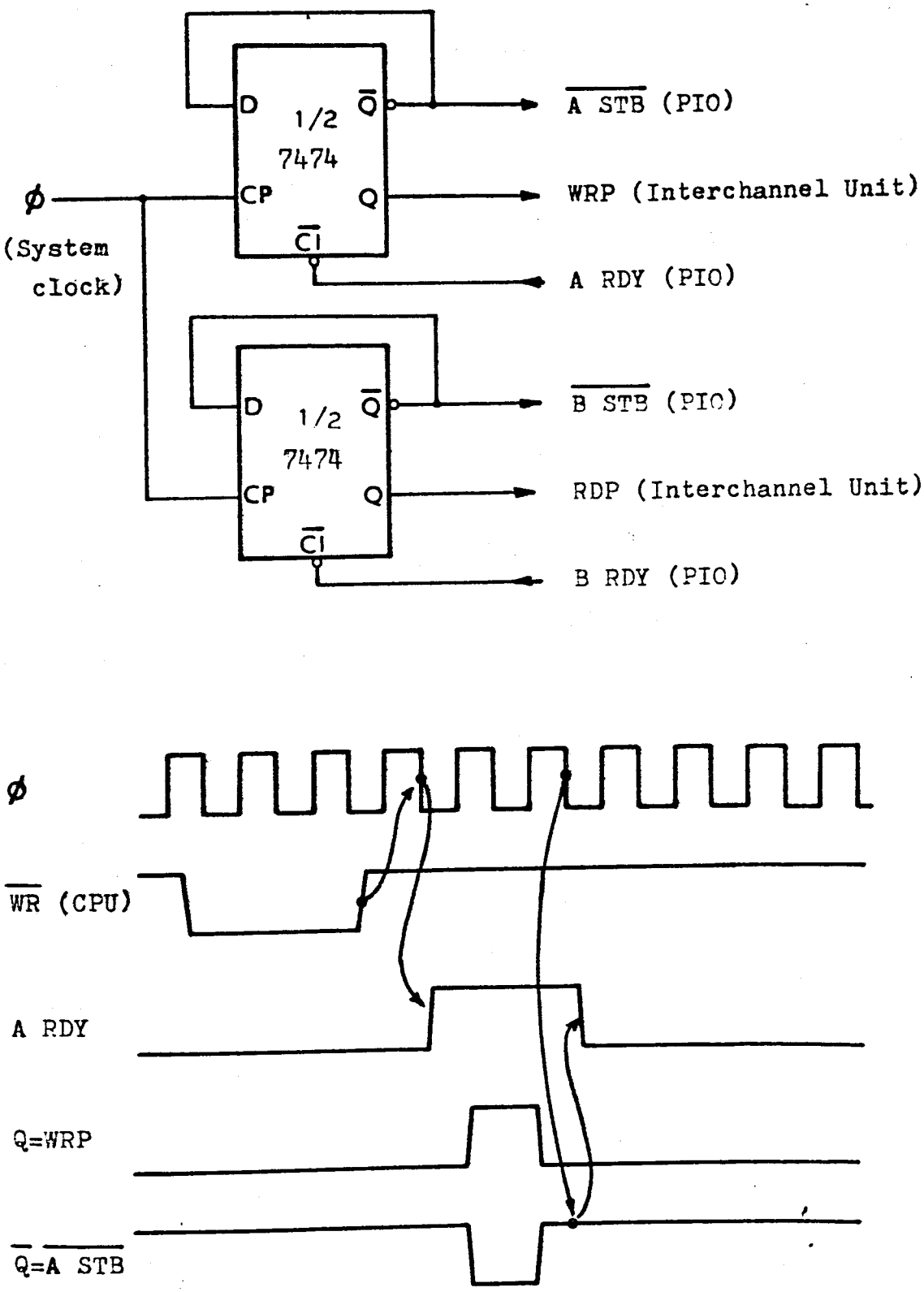


Fig. 4.6 - Generation of the Interunit Interface control signals

Normally 'A RDY' is low, inhibiting any output changes (Q=Ø, $\overline{Q}$=1). Following a CPU write, 'A RDY' is forced high, which allows Q and $\overline{Q}$ to change on the next clock pulse (D=$\overline{Q}$). Another change takes place again one clock period later, thus generating the PIO strobe pulse ($\overline{Q}$=$\overline{\text{A STB}}$), which again resets 'A RDY' - Q and $\overline{Q}$ are forced to their initial values and no more strobe pulses are generated until the next output cycle. The pulse generated in the Q output is used to gate data from the PIO output port into the Interchannel Unit Source Buffer (Q=WRP).

Because a minimum of 11 cycles separate two consecutive CPU output instructions (while the described handshake mechanism lasts for about 3 cycles) data is latched into the Source Buffer before a new byte is written to the PIO and, therefore, new data may be output without any further delay.

This has two consequences:

(a) no interrupts are required to enable a new transfer to the PIO;

(b) output block transfers, under CPU control, are possible.

The complete transmission process is summarized as follows and an associated timing waveform set shown in Fig. 4.7:

(a) when a packet stored in the Bus Output Queue is ready for transmission, the transmitter process within the Link Access Unit monitors the Source Buffer status;

(b) when the buffer becomes empty transfer to the Source Buffer can start;

(c) first, the packet length, together with the required

destination address is accessed;

(d) WRSB is reset and the LAU destination address set into the PIO control port;

(e) block transfer is then initiated;

(f) at the end of the block transfer, WRSB is set, as required by the Interchannel Unit control logic;

(g) the internal buffer within the Link Access Unit holding the packet is then released.

4.2.3.2 Packet reception

In many aspects use of the handshake signals 'B RDY' and 'B $\overline{\text{STB}}$' to control the transfer of data between the Destination Buffer within the Interchannel Unit and the PIO input port is similar to that described for transmission but in this case the pulse generated at the Q output is used to gate data into the PIO input port. Also the timing sequence is similar, but since the strobe pulse to the PIO is generated only after a CPU read instruction, it is necessary to perform a "dummy" read before each block transfer from the Destination Buffer.

A summary of the reception process is therefore as follows and a timing diagram is shown in Fig. 4.8:

(a) the receiver process within the Link Access Unit monitors the Destination Buffer status;

(b) when the buffer becomes full and a local buffer is available a transfer from the Destination Buffer can start;

(c) a "dummy" read is executed as explained above;

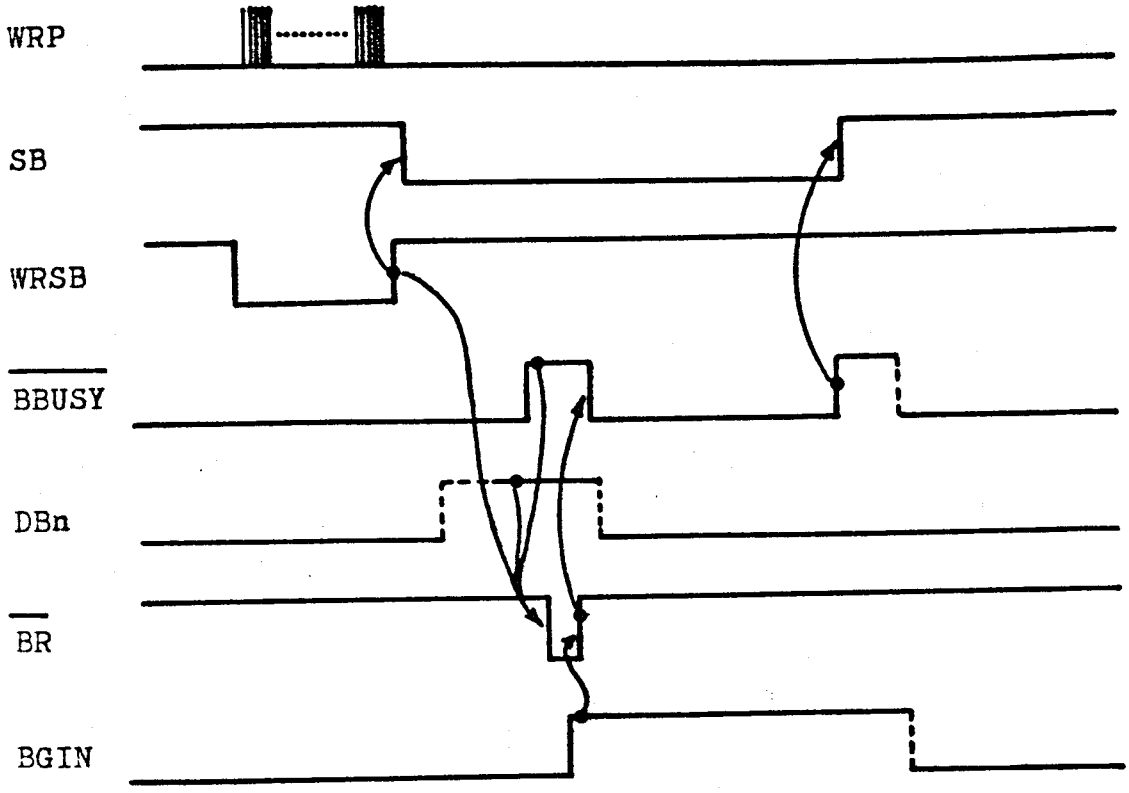


Fig. 4.7 - Packet transfer to the Interchannel Unit

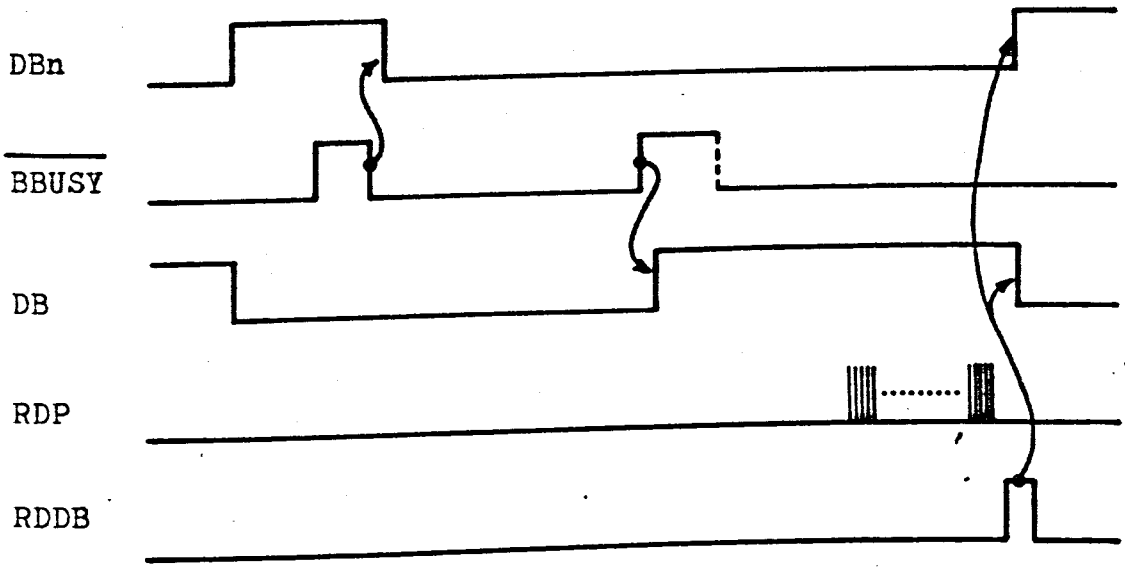


Fig. 4.8 - Packet transfer from the Interchannel Unit

(d) the first byte is read to determine the packet length;

(e) block transfer is then initiated;

(f) at the end of the block transfer, the RDDB pulse is generated by the PIO under program control, as required by the Interchannel Unit control logic;

(g) the packet is transferred to the Bus Input Queue within the Link Access Unit.

4.3 Exchange Management Unit

The Exchange Management Unit (EMU) is a separate autonomous unit connected to the Interchannel Unit and can, under the control of an operator, collect data relating to the call processing tasks within each Link Access Unit, monitor the overall performance of the exchange and indicate system fault conditions.

It is similar in terms of hardware to a Link Access Unit, except that instead of the serial link imterface card a conventional V.24/20 mA current loop interface is used to a VDU or TTY. This is based on a 8251 USART, initialized on power on (reset) under program control.

The control programs are held in 4K bytes of PROM - 2K bytes for a slightly modified version of the MOSTEK DDT-80 monitor and the remaining 2K bytes implementing the exchange monitor. The functions of the latter will be described in Chapter 7.

The timing requirements of the Exchange Management Unit are different from those in a Link Access Unit and a single time-out interval is used to protect a monitor command against possible

failures of the addressed Link Access Unit.

4.4 Interchannel Unit

As has been described, the complete exchange is comprised of a readily varied number of Link Access Units and an Exchange Management Unit. These are interconnected by means of the Interchannel Unit. The arbitration and synchronization mechanisms required for the access and transfer of packets across the bus within the unit are highly dependent on some general requirements the exchange should meet and these are now discussed.

4.4.1 Packet transfer requirements

The exchange must provide a virtual circuit service and hence the packet transport mechanism must ensure guaranteed packet delivery and acceptance without the need to default to an acknowledgment/time-out recovery mechanism. Also, to keep the overall throughput within acceptable limits, it is essential to minimize the occupancy time of each packet transfer.

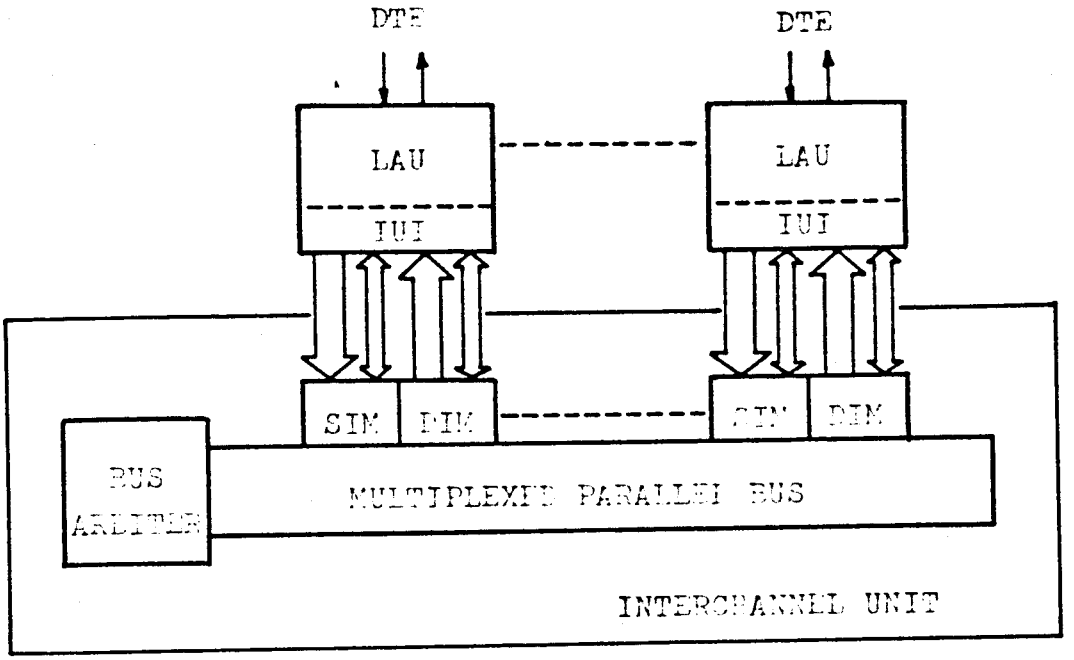
A packet transfer may be considered as being made up of three components: transfer of the packet from the Link Access Unit to the Source Unit, the actual transfer of the packet across the interconnecting bus and the transfer of the packet from the Destination Unit. Thus, to fulfill the previous non-loss requirement and to achieve synchronization between two communicating units, it is necessary to ensure that a transfer

takes place only when the receiving unit is ready to accept a new packet. Also, the occupancy time of the bus may be reduced to the transfer time, by providing packet buffering both at the source and destination and implementing a synchronized acquisition mechanism.

4.4.2 Hardware design

The Interchannel Unit is a separate autonomous unit which provides a fast (1.8 Mbyte/s) communications facility between the units which are connected to it. Essentially, it is comprised of a number of full-duplex parallel ports each supporting one input and one output channel as shown in Fig. 4.9. The signals used to control packet transfers across the bus are described in Fig. 4.10. Each channel contains a separate first-in first-out (FIFO) packet buffer and additional control logic, as shown in Fig. 4.11.

When the Source Buffer becomes empty, after a packet is completely transferred or following a reset, the SB latch is set, signalling to the Interunit Interface that a new packet may be loaded. After a packet is loaded into the Source Buffer, the WRSB pulse generated by the Interunit Interface is used to reset the SB latch (Source Buffer full), to latch the destination port address and to enable the bus acquisition control mechanism. The latter waits until no other transfer is currently in progress across the bus ($\overline{\text{BBUSY}}$=1) and the required destination channel buffer is empty (DBn=1) before making an interunit transfer request to the bus arbiter.



SIM = Source Interface Module
DIM = Destination Interface Module

Fig. 4.9 - Schematic of the Interchannel Unit

INTERCHANNEL UNIT BUS SIGNALS

D0-7	Data Bus (tri-state input / output) : used to transfer data between the source and destination modules within the Interchannel Unit
A0-3	Address Bus (tri-state output) : destination LAU address
DB0-15	Destination channel status (output) : 0=busy; 1=idle
BBUSY	Bus status (open collector output) : 0=busy; 1=idle
BR	Bus request (open collector) : active low
BGIN	Bus grant input (daisy chain)
BGOUT	Bus grant output (daisy chain)
CLOCK	Bus master clock
CLKIN	Packet transfer clock (tri-state output)

Fig. 4.10 - Definition of the Interchannel Unit bus signals

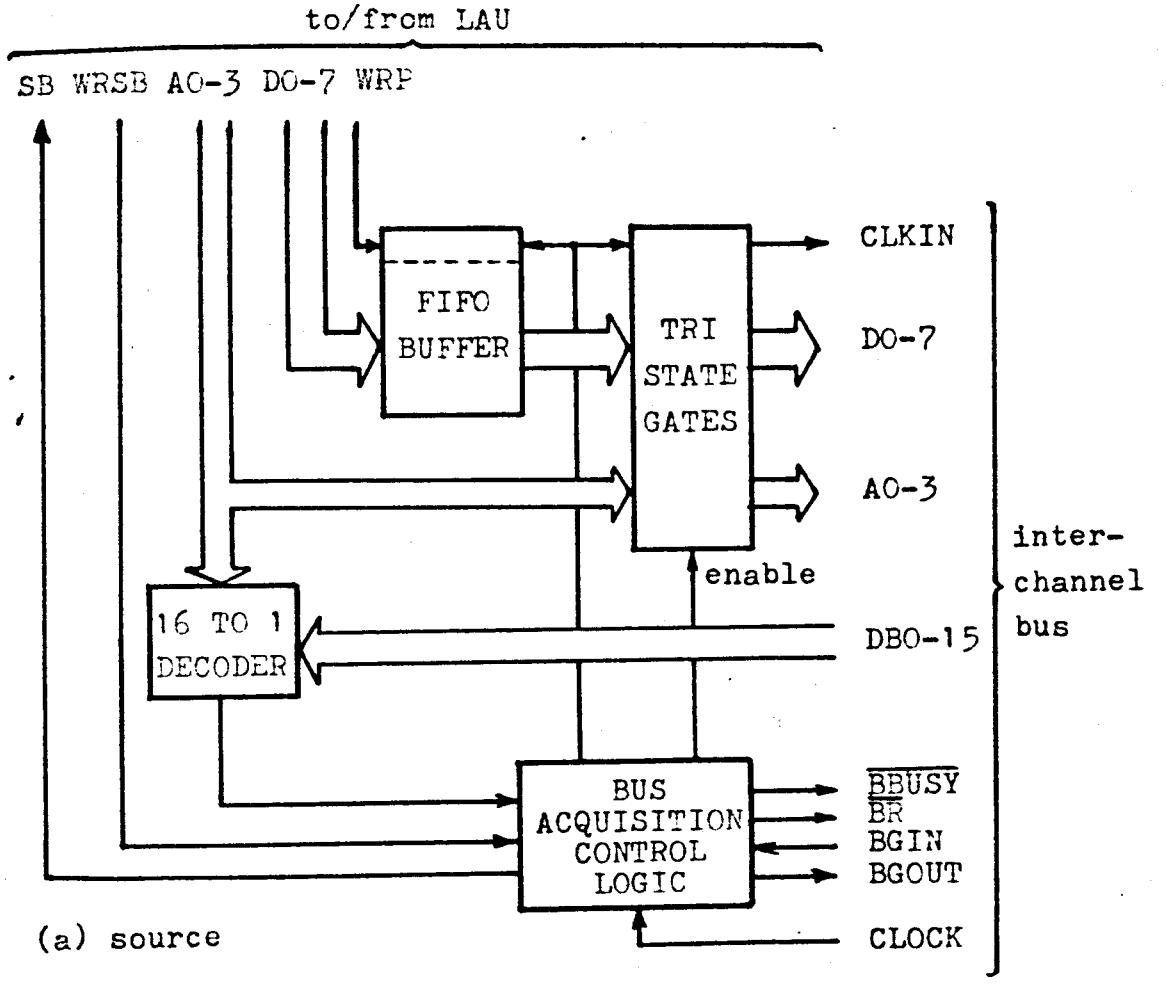


Fig. 4.11 - Schematic of source and destination interface modules

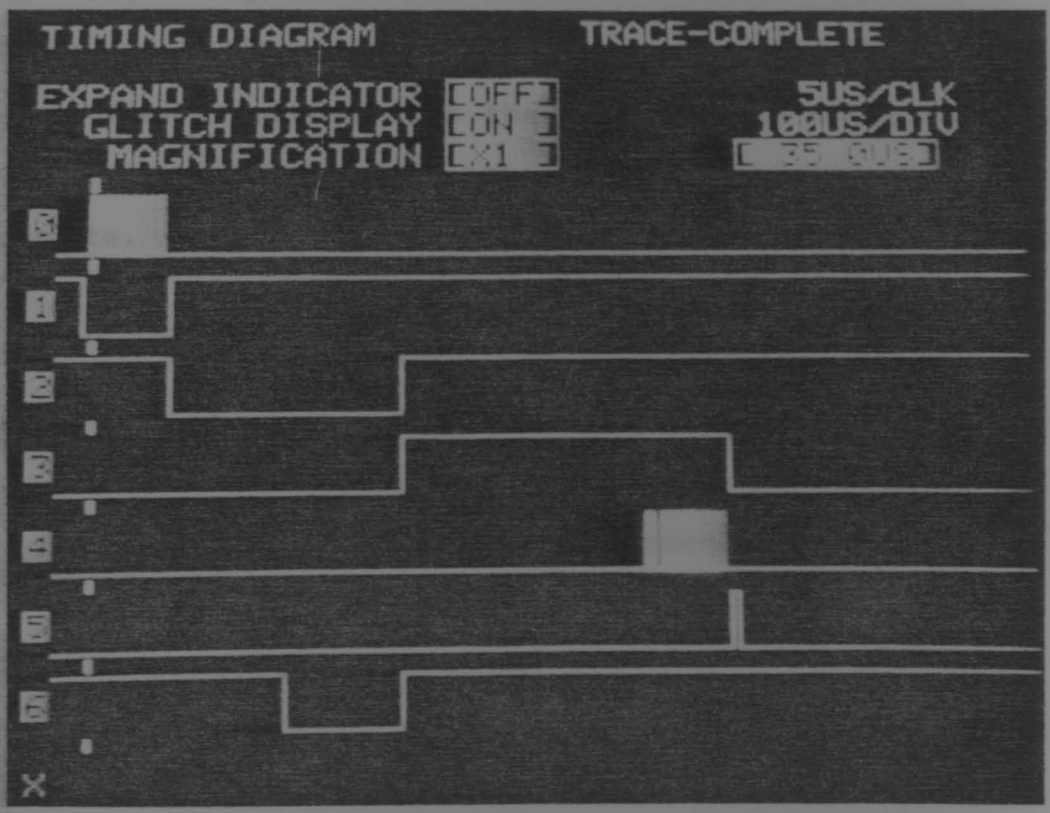
destination channel address onto the address bus and proceeds to transfer the packet contents across the bus using the bus clock. On completion, the Source Buffer becomes empty again and the Destination Buffer status becomes full.

The destination channel, after decoding its own address, enables the bus clock and data lines to its FIFO and resets DBn, thus inhibiting further transfers to the channel. Then, after the packet has been received, the bus returns to the idle state and the DB latch is set (Destination Buffer full). This signals to the Interunit Interface that a packet has been received and may now be read into an internal packet buffer. Finally, on completion of the transfer of the packet, the RDDB pulse generated by the Interunit Interface is used to reset the DB latch (Destination Buffer empty) and set DBn, thus reenabling further transfers to the channel. A timing diagram of a typical packet transfer is shown in Fig. 4.12.

4.4.3 Bus arbiter

Access to the internal parallel bus is controlled by a separate bus arbiter which is responsible for multiplexing packet transfers across the bus. A daisy chain bus arbitration scheme is used and, as will be discussed in Chapter 7, it ensures a reasonable and fair distribution of use of the interchannel highway.

The priority algorithm used to handle simultaneous requests is, therefore, determined by the position of the requesting module on the bus. All bus requests are OR-ed together and the



Trace 0 - WRP Data strobe (PIO output)

Trace 1 - WRSB Initiate packet transfer

Trace 2 - SB Source Buffer status (0=full; 1=empty)

Trace 3 - DB Destination Buffer status (0=empty; 1=full)

Trace 4 - RDP Data strobe (PIO input)

Trace 5 - RDDB Packet transfer complete

Trace 6 - BBUSY Bus state (0=busy; 1=idle)

Fig. 4.12 - Packet transfer across the Interchannel Unit

module on the bus. All bus requests are OR-ed together and the bus grant signal issued by the arbiter in response to a request is passed on from one unit to another (BGIN/BGOUT) until the first requesting module is reached. The latter assumes control of the bus (BGIN=1) and inhibits the grant signal (BGOUT=0) from being passed further down the highway. Any other requesting modules must therefore wait until the current transfer is complete.

CHAPTER 5

PACKET SWITCHING EXCHANGE - PROTOCOLS

In Chapter 3, X.25 was presented and discussed in a global perspective, oriented towards the interpretation of its more controversial or ill-defined aspects. In addition, however, a number of choices and strategies allowed by or inherent to the protocol must be discussed in connection with the implementation and the selected network topology and their effects on the overall network performance must be assessed.

X.25 only defines, in general,the procedures that must be adopted by a DCE (the interface with the subnetwork) and a DTE should follow similar or compatible procedures. Hence, when implementing the protocol in the exchange no inference is made about particular implementation details of the DTE's but rather it is assumed that they broadly adhere to the general protocol rules.

In this chapter some implementation decisions will be discussed together with their effects on such factors as resource allocation, flow control strategy and the internal (LAU/LAU) protocol. The software organization within the exchange will then be discussed in Chapter 6.

5.1 Link management

As stated earlier, set up and disconnection of a logical link using the Asynchronous Response Mode procedures (LAP) must be separately performed in both directions. This creates possible synchronization problems that are discussed in Appendix A which influence their implementation as follows:

(a) once a link is set-up in one direction, Information frames may be transmitted in that direction on the assumption that set-up in the other direction is in progress or has already been completed;

(b) following a Link Access Unit reset or a physical (power down) or logical link failure, further set-up of the link (SARM) is preceeded by a logical disconnection (DISC) for synchronization purposes;

During a link disconnection (DISC), the following procedures are adopted:

(a) once a DISC is acknowledged, both the Primary and the Secondary are assumed to be logically disconnected. This is intended to avoid deadlocks that could arise if one station was kept indefinitely waiting for the other to send a DISC;

(b) since a DISC may have to be transmitted or retransmitted by the remote Primary, while a local Primary set-up may then be required (thus possibly leading to loop conditions involving DISC and SARM in opposite directions), a single time-out interval is utilized to delay a set-up following a

local disconnection. However, this condition is cleared if, in the meantime, a remote SARM or DISC is received.

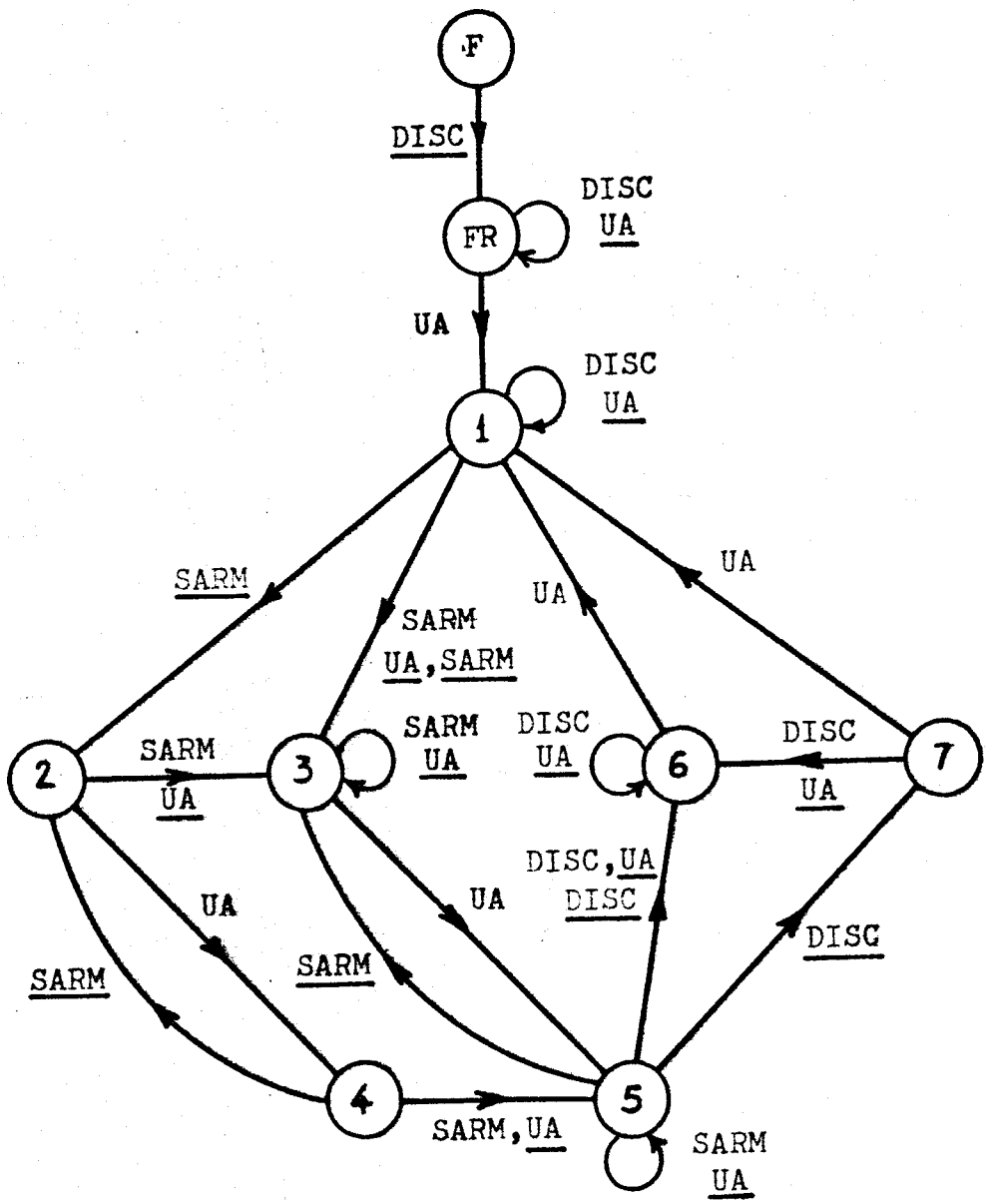
The above determine a link state transition description which requires a tight coupling between Primary and Secondary states as represented in Fig. 5.1. For clarity, handling of error conditions and of disconnection during the set-up phase are not represented.

5.1.1 Acknowledgment strategy

Once a logical link is in the Information transfer phase, mainly Information and RR supervisory frames are exchanged across the X.25 interface. The use of RR is necessary, of course, in case of unbalanced flow of Information frames, since frequent use of piggyback acknowledgment is thus not possible. Use of supervisory RNR and REJ frames is reserved for exceptional conditions and is not relevant to the following discussion.

Even when piggyback acknowledgment is possible, X.25 allows the use of RR frames to perform the acknowledgment function. Clearly this unrestricted use of RR frames reduces througput, increases processing overheads and creates additional buffer requirements. However, advantages of using RR frames result from their short length compared to the average length of an Information frame. This results in early acknowledgments and therefore buffers are released sooner and window rotations are faster.

The advantages of using RR frames are only fully



State	Primary	Secondary
F	Failure	
FR	Failure Recovery	
1	Disconnected	Disconnected
2	Connection/Reset	Disconnected
3	Connection/Reset	Connected
4	Connected	Disconnected
5	Connected	Connected
6	Disconnection	Disconnected
7	Disconnection	Connected

Note: frame transmissions are underlined

Fig. 5.1 - Implemented link state transition

accomplished, however, if they are given higher priority over Information frames and although this is not difficult to guarantee, its consequences must be assessed.

Since the N(R)'s are submitted for validation at the receiver, their normal sequence cannot be disturbed, for example, by the possible insertion of high priority RR frames ahead of Information frames already processed and queued for transmission.

One way of ensuring that the N(R)'s will be transmitted strictly in sequence is to delay the complete processing of a frame until its transmission is possible. This is neither practical nor efficient due to the fact that level 2 processing is done concurrently with other tasks and hence this would certainly create not only high processing overheads at the beginning of a transmission, but also transmission gaps, thus compromising the link efficiency.

Alternatively, a frame may be processed and queued but allow its control field (N(R)) to be later updated (on the fly) at the moment of transmission. This is the ideal solution if the link management is transparently handled by the hardware with automatic generation of supervisory frames and insertion of the Information frame header [53]. Doing it under program control, however, as is the case in the present implementation, means that virtually all the frame processing would still be defaulted to the beginning of transmission with enormous overheads in the transmitter interrupt routine. Not only is updating the N(R) relatively complex due to the fact that the control field contains several subfields, but also further frame decoding is required to insert the N(R), determine priorities, etc.

The solution adopted, therefore, which avoids some of the

above problems, is as follows:

(a) once a frame is processed at level 2, it is queued for transmission, without further update of its control field;

(b) transmission is carried out in a first-come, first-served basis;

(c) frames are transmitted without gaps (if the Link Output Queue is not empty);

(d) piggyback acknowledgment is preferred to the use of RR, unless this is impossible or use of RR is required by the protocol rules.

Clearly, this reduces overheads in the interrupt routines and interaction with the level 2 frame processing but, because the N(R)'s do not generally reflect the current V(R) at the moment of transmission, acknowledgments are delayed with two main components:

(a) delay due to piggyback effect (average Information frame length);

(b) delay due to non update of N(R) (average queue length).

This problem is further addressed in Chapter 7.

5.1.2 Buffer allocation

After transmission, Information frames must be kept in a retransmission list until acknowledged. This requires some form of frame reprocessing following transmission, which also creates possible contention problems.

In the absence of errors in the acknowledgment mechanism, a received N(R) points to Information frames already transmitted and hence the release of buffers through the retransmission list is straightforward. Errors in the acknowledgment mechanism, however, are considered in the protocol and a particular case arises if the received N(R) points to an Information frame which is still awaiting transmission. Hence, to avoid contention problems, an Information frame should be referenced in the retransmission list only after it has actually been transmitted and hence the above situation should be handled as a detectable error.

Other contention problems may arise when retransmission is required and previously processed Information frames are still in the Link Output Queue. This is possible during REJ recovery and thus it is necessary to avoid Information frames (buffers) being referenced twice in the Link Output Queue. Other potential problems are associated with buffer releases during time-out recovery and these are also discussed in Appendix A. The obvious way to avoid contention in the above cases is to delay any retransmission until all current frames are removed from the Link Output Queue.

We have then assessed:

Firstly, the need to reprocess frames after transmission.

This must be included in the link level processing and synchronized with the processing of received frames since the received N(R)'s are validated against the transmitted N(S)'s.

Secondly, the need to avoid contention problems, thus requiring a clean identification of different phases of processing of Information frames.

To avoid unnecessary transfers of pointers between queues, the following mode of operation has been adopted:

In the Link Output Queue, three types of frames referenced by pointers and counters are considered as shown in Fig. 5.2 (a):

(a) frames already transmitted but not yet reprocessed by the link level (NIF3-NIF4);
(b) a frame in transmission (NIF4-NIF5= Ø/1);
(c) frames waiting transmission (NIF5).

A single queue (Packet Output Queue) holds all Information frames under processing within the link level as shown in Fig. 5.2 (b):

(a) Information frames transmitted, reprocessed but not yet acknowledged (NIF1-NIF2);
(b) Information frames in the Link Output Queue, waiting transmission and/or reprocessing (NIF2);
(c) Information frames transferred from the packet level and thus waiting link level processing (NIFØ).

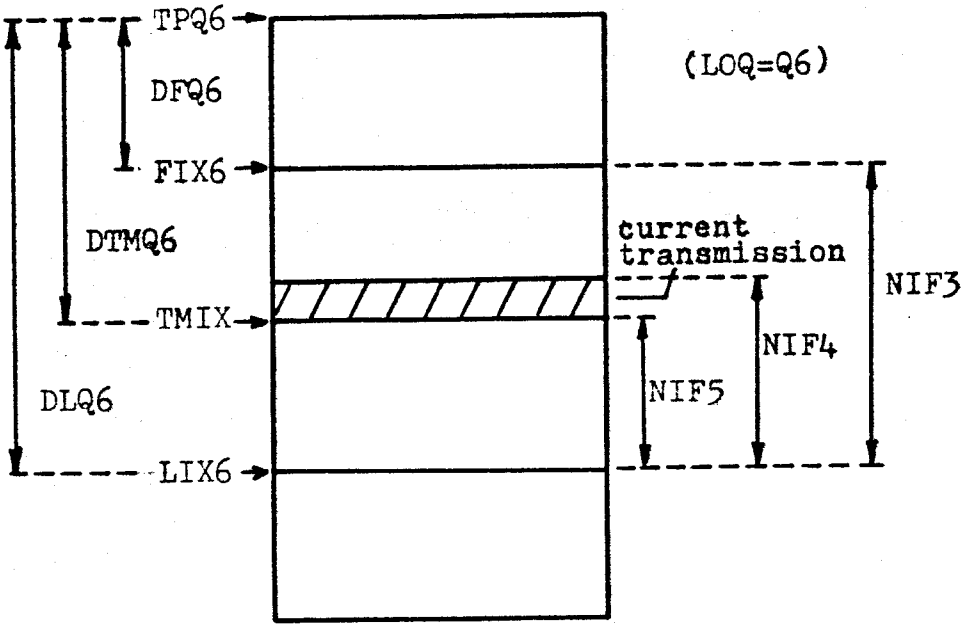


Fig. 5.2 (a) - Structure of the Link Output Queue

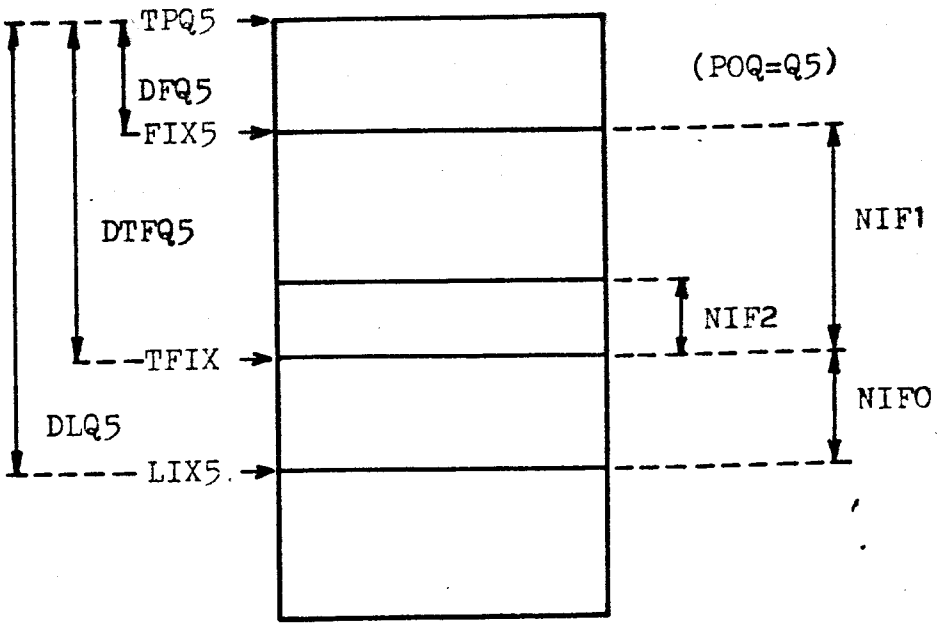


Fig. 5.2 (b) - Structure of the Packet Output Queue

5.2 Flow and congestion control

Flow control is mainly a resource management mechanism (implemented according to some protocol rules [54]), concerning the fair and efficient allocation of the subnetwork resources to achieve the best possible performance even in the situation of network congestion. Past experience has shown [55] that packet switching networks, if no congestion control is exercised, are highly susceptible to degradation and even to deadlock problems, as shown in Fig 5.3 by a typical network transfer curve.

Although a flow control strategy must always be considered in the perspective of the whole network, particular and complementary schemes have been envisaged to solve the problem of congestion according to some local or global congestion measure; for example network access, internodal, end-to-end.

A survey of flow control schemes may be found in [54,56] and such problems as resource allocation, deadlock avoidance, routing and congestion control are discussed for example in [14,57,58,59,60,61].

Flow control in the exchange must be viewed in the context of the X.25 protocol and the exchange topology. We recall that X.25 flow control at the packet level is constrained by the fact that Data packets received from the DTE's cannot be discarded once accepted at the link level and submitted for packet level processing.

This naturally determines that any flow control policy at the network access point having direct and immediate effect, has to be carried out at the link level. On the other hand, X.25 allows flow control to be exercized on a virtual circuit basis

but the only way to render it effective is to relate the rotation of the X.25 windows with some internal window(s), thus making it possible to restrict the total number of packets per call that are buffered in a node and/or outstanding along the virtual circuit.

Three examples of flow control in X.25 networks are particularly relevant for the present discussion and are now briefly described:

GMDNET

In GMDNET [55] a global flow control is based on the structure of a buffer pool into buffer classes, as shown in Fig. 5.4, and packets have access to a buffer class according to the number of hops they have covered in the network. Input traffic is further restricted (by means of an Input Buffer Limit), thus greatly improving the throughput performance. This is combined with an individual virtual circuit flow control, both node-to-node and end-to-end, which is important in avoiding performance degradation in the case of limited sink rates.

The virtual circuit flow control is based on a window mechanism, with the windows dynamically adjusted up to a maximum value; acceptable values have been shown to be Wn=2 (node-to-node) and We=h+1 (end-to-end), where h is the number of hops in the virtual circuit.

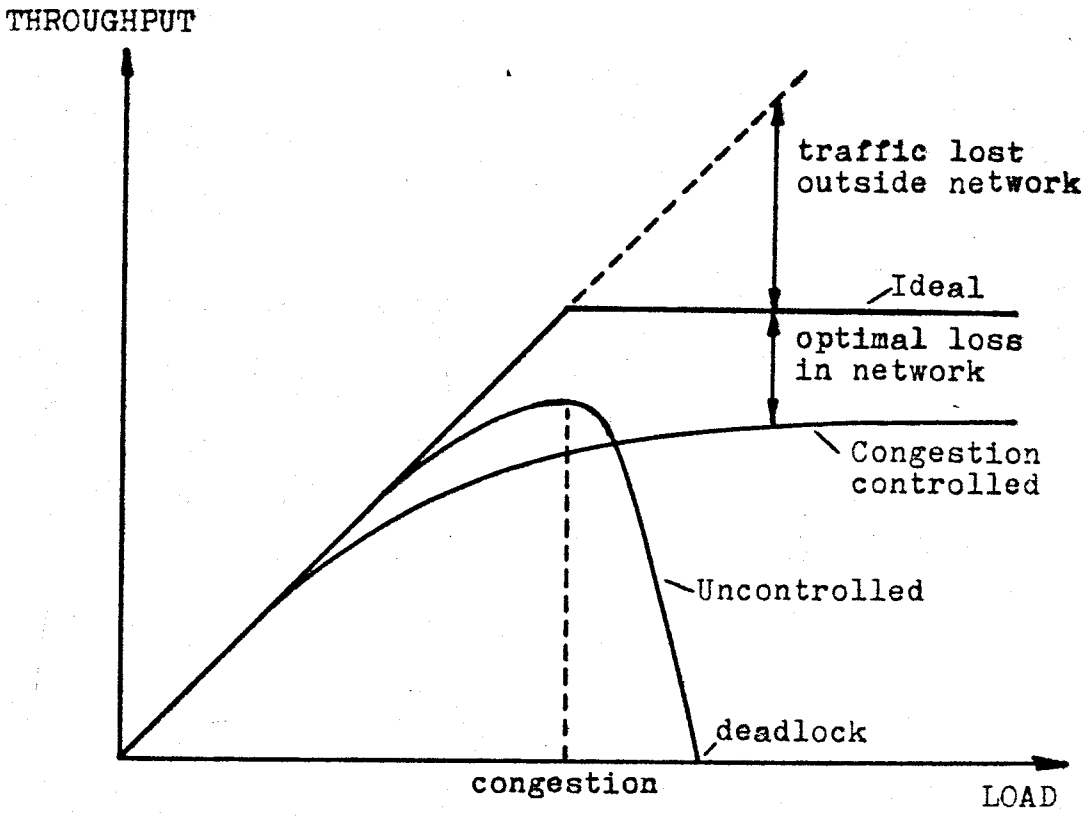


Fig. 5.3 - Packet switching throughput characteristics

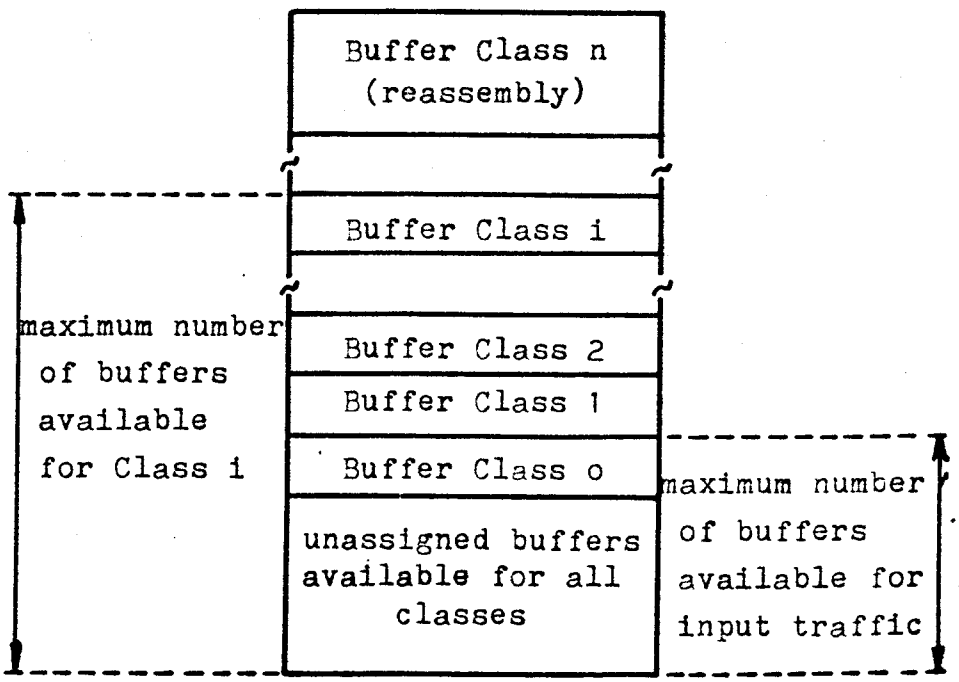


Fig. 5.4 - Structured buffer pool

Important results derived from the study of several Input Buffer Limit schemes [55,62,63,64] showed that:

(a) an optimum threshold value exists for each buffer pool that maximizes the throughput and prevents degradation;

(b) a too high value (in the limit no control) introduces long delays and leads to very rapid throughput degradation;

(c) a too low value leads to a more conservative performance (small delays and almost no throughput degradation) but maximum throughput is reduced since resources are underutilized;

(d) for the optimum Input Buffer Limit, transit traffic is kept flowing only with a fraction of the available buffers.

DATAPAC

In DATAPAC [65] the virtual circuit service is achieved by means of a subnetwork transport service on top of a low level datagram service. Under congestion, packets may be discarded, but resequencing and recovery from losses/duplicates is performed on a subnetwork end-to-end basis. The subnetwork flow control is selective (virtual circuit) and based on an internal window rotation, synchronized with the X.25 windows at the source and destination network access interfaces [66].

When the destination DTE acknowledges oustanding packets (implicitly updating the credit) the destination node rotates the internal window accordingly (internal acknowledgments are delayed until the rotation of the external window). The source node

rotates the source X.25 window only if the internal window can, thereafter, accomodate all the packets the source DTE is authorized to send. When the three windows have the same size and no packet conversion is internally done, then the rotation of the windows has end-to-end significance.

TRANSPAC

In TRANSPAC [67] the declared throughput class negotiated during the call set-up is compared to the real measured throughput and used to dynamically allocate resources to the virtual circuits. In case of congestion, and according to some pre-defined thresholds, new calls are rejected, then credit is restricted and finally some calls are even selectively cleared.

5.2.1 Exchange congestion

Having considered the implications of using X.25, let us now examine the constraints introduced by the exchange topology and the bus transport mechanism.

We shall analyse in the first place how congestion may develop, the reasons for that to happen and its possible effects, beginning with a simple example involving two Link Acces Units - LAU1 and LAU2 with traffic from LAU1 to LAU2.

As transfer rates across the bus are much higher than the serial link baud rates, packets will initially move from LAU1 to LAU2 with average small queues in LAU1. They will begin to

accumulate in LAU2 if the rate at which packets are accepted by DTE2 is lower than the rate at which they are generated by DTE1. However, if the number of established calls is small, level 3 flow control will certainly be effective, as the total number of packets in transit is naturally limited (a lack of credit simply causes a window overflow in DTE1). The problem presents a new dimension, however, when more DTE's are considered, as the number of calls established across any X.25 interface is now likely to increase.

Thus, in general, accumulation of packets in the direction of the serial links will be due to at least one of the following reasons: abnormal traffic from other LAU's (even if transient), local recovery actions (Information frame retransmission, link reset) or DTE flow control.

Moreover, a new phenomenon may take place: following the described saturation of a LAU, packets will begin to accumulate in the Bus Output Queue of other LAU's if they still hold packets for the saturated LAU. This means that congestion backs up to other LAU's, possibly leading to some temporary performance degradation if this blocking prevents other traffic from moving to non congested LAU's. The whole situation is difficult to quantify, as it is highly susceptible to instantaneous traffic distributions among the LAU's (order of the packets in the queue, for example).

We need a deeper view into the nature of the traffic within the exchange to assess possible deadlock conditions and to determine the most suitable flow control strategy.

5.2.2 Deadlocks

The only possible deadlock condition within the exchange is of the store-and-forward type. Generally, a direct store-and-forward deadlock occurs when two nodes in a network are full of packets to each other and no buffer may be granted for packet reception in the corresponding node-to-node link [14]. However, in the exchange, due to the implemented bus transport mechanism, a deadlock may occur in a slightly different way, as will be now described.

A LAU may be blocked (new packets cannot be loaded into the Source Buffer within the Interchannel Unit) and yet it may not be congested; this should therefore be considered as an indication of congestion elsewhere and not the only criteria for deadlock assessment. Also, a temporary lack of buffer resources is not always a useful indication of a deadlock, provided that potentially releasable buffers still exist in the LAU (for example, those containing Information frames to be transmitted across the X.25 interface).

For a deadlock to occur between two LAU's the following conditions must be simultaneously present:

(a) both Source Buffers must be full with packets addressed to the other LAU;
(b) both Destination Buffers must be already full with other packets;
(c) no buffers may ever be granted for transit traffic (no buffers are releasable).

This is shown diagrammatically in Fig. 5.5. Thus, it is a necessary, although not sufficient condition of deadlock in the exchange that a LAU has all its buffers occupied with transit traffic to other LAU's.

We must, therefore, assess how the Bus Output Queue may grow: the main stream of packets for that queue comes from the serial link; however, as will be discussed later, there are circumstances where a packet creates a loop within the exchange, when it is internally returned, after processing, to the source LAU.

The problem that we face is, therefore, how to control the growth of the Bus Output Queue and what strategy - detection, avoidance or prevention - to adopt concerning deadlocks. This is now discussed in connection with the proposed and implemented flow control strategy in the exchange.

5.2.3 Flow control strategy

We first recall two aspects of particular importance in the adopted flow control strategy:

(a) no packet retransmission is supported by the exchange and hence no packets should be internally discarded if the virtual circuit service is to be preserved (unless the corresponding call is cleared);

(b) every path within the exchange is of single hop length (h=1).

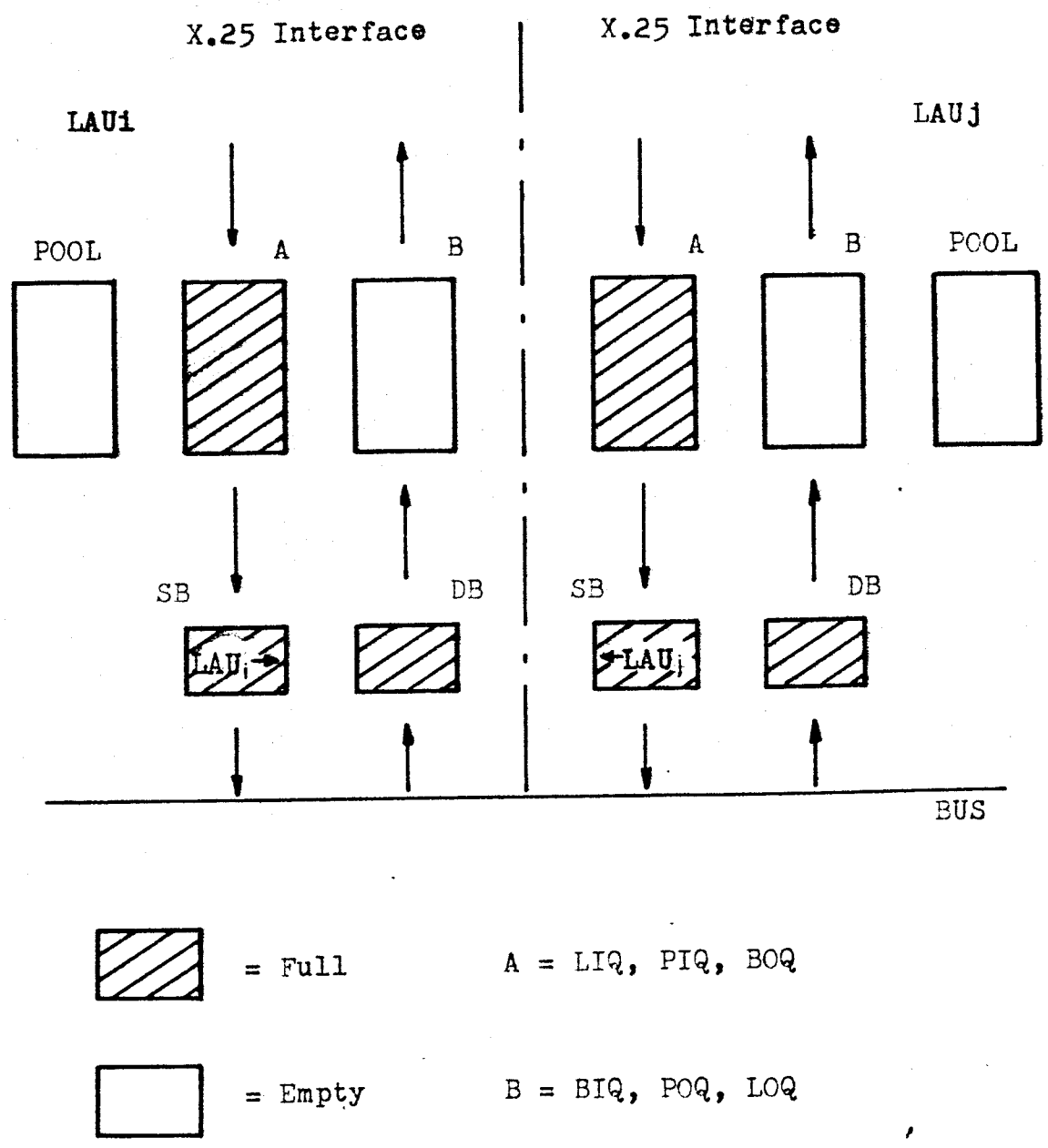


Fig. 5.5 - Exchange deadlock

Flow control in the exchange is exercised at two levels - at the network access, corresponding to the link layer of the protocol, and on a selective virtual circuit basis, corresponding to the packet layer of the protocol.

5.2.3.1 Virtual circuit flow control

The previous considerations determined the implementation of a very simple mechanism at the packet layer - window rotations for each virtual circuit will be performed on an end-to-end basis by transparently transferring, across the exchange, Data and supervisory packets with their send and receive sequence numbers (P(S) and P(R)) unchanged. Under these circumstances the default X.25 window (W=2) constitutes a good solution to match the number of hops (h=1) in each virtual circuit (W=h+1).

There is another argument in favour of the implemented strategy. In multihop networks every node may act as a source, a destination or a relay for packets in transit between other source and destination nodes. A node may then wish to flow control some virtual circuit(s) in order to avoid unfair access to resources by some transit traffic. However, in the exchange, all transit traffic inside a LAU has a local nature (source/destination). Then the problem of fair allocation of resources among different virtual circuits may be transferred to the DTE's, as in any case they must regulate the rate at which new calls are generated and the sharing of bandwidth among concurrent calls.

5.2.3.2 Network access flow control

The network access flow control is more critical to implement as it is directly related to the buffer management policy and should be considered in connection with the potential deadlock conditions.

Basically an Input Buffer Limit scheme, usually referred to as "free buffer allocation" [55,63], was adopted: as soon as the number of available buffers falls below a certain limit (NØ), a node enters a busy condition and some traffic restrictions are introduced. Normally transit traffic is given higher priority, as more resources have already been used by transit traffic and it usually has a higher probability of delivery.

Hence, we allow transit traffic to enter a busy LAU (otherwise all flow within the exchange could halt), while input traffic will be restricted; that is, Information frames will be discarded (they are naturally protected by level 2 retransmission mechanisms).

This restriction of input traffic does not mean that frame reception should be completely suspended; on the contrary, supervisory frames and the control field of Information frames are still required to acknowledge outstanding Information frames in the opposite direction, thus contributing to the release of additional buffers. Hence, frames must be received, processed and only then the corresponding buffers should be released (thus, only Information frames are "really" discarded). As reception under these circumstances does not lead to cumulative buffer requirements (rather it is as if buffers were reused by the receiver process), a maximum number of buffers, say Nr, must be

assigned for reception during a busy period.

By conveniently structuring the buffer pool, both these objectives (transit traffic to flow with higher priority while keeping a fixed level of frame reception) may be met. When entering the busy condition NØ-Nr (NØ>Nr) buffers may be granted to handle transit traffic and during that condition all free buffers in excess of Nr will be exclusively assigned for that purpose.

An additional condition was introduced: to avoid persistent succession of busy/non busy transitions, a certain level of buffer regeneration (hysteresis) is required before returning to a normal non busy condition (N1>NØ). This is naturally achieved as soon as packets are transferred across the bus and transmitted Information frames across the link are acknowledged.

If the busy condition persists, however, transit traffic naturally degrades due to lack of credit in the reverse direction (stopped due to input traffic restriction), which incidently is an indirect way of congestion control. Moreover, if the congestion backs up to other LAU's, they themselves enter a busy condition and also begin to flow control input traffic.

5.2.3.3 Deadlock avoidance

The described flow control scheme is in the first place intended to prioritorise transit traffic as a measure to keep the overall performance within acceptable limits (ideally with no degradation, past the threshold of congestion). Moreover, it eliminates the main cause of deadlock (that associated with input

traffic overflowing the Bus Output Queue) and in this respect it provides a deadlock prevention mechanism.

However, the second cause of deadlock must now be considered. As it is associated with a form of internal looping, at least three LAU's are required to produce a dealock (thus constituting, in some way, a form of indirect store-and-forward deadlock). Additionally, it is necessary that two LAU's become completely full with transit traffic and simultaneously (and unlikely) with their Source Buffers loaded with packets addressing each other. To reach this situation it is further necessary that the rate at which these loop packets enter the LAU is higher than the rate at which transit traffic leaves the LAU.

The detection of a deadlock could, therefore, be based on a time-out mechanism, thus possibly keeping the whole exchange degraded for a reasonable period of time. On the other hand, prevention schemes would introduce severe overheads, eventually requiring the exchange of internal control packets and a more complex channel state description.

We consider that the described situation is highly improbable and even constitutes an indication of severe degradation, thus justifying the implementation of a deadlock avoidance scheme - whenever a LAU becomes full with transit traffic (a threshold is reached in the Bus Output Queue), it is simply reinitialized at the packet level by clearing all current calls (this is discussed in the next section).

5.3 Internal protocol

The packet level of X.25 is responsible for the management of virtual circuits and, due to the end-to-end nature of a virtual circuit, this management clearly assumes different aspects in the DTE's and the LAU's.

A LAU does not generate any calls and only initiates call clearances for recovery purposes or if a call cannot be established. On the other hand, routing and flow control are required on a network wide basis. Thus, in addition to the X.25 interface protocol an internal protocol must also be established.

The internal protocol adopted depends, of course, on the network topology. An obvious choice is to keep the highest possible compatibility between both protocols at least at the packet level, since any packet manipulation inside the network must be hidden from the X.25 interfaces.

In the proposed implementation the internal protocol is simply an extension of the X.25 packet level and assumes that the exchange provides a reliable transport of packets across the internal parallel bus. An essential aspect of the implementation is the internal channel assignment scheme. Since the exchange provides a single hop path between any pair of LAU's, logical channels set-up at the X.25 interfaces must be mapped at each LAU to an internal logical channel in order to establish and maintain the virtual circuit.

5.3.1 Internal logical channel assignment

Although well defined rules exist for the set-up of logical channels at the X.25 interfaces, the same is not readily applicable to the internal channels - mainly because each LAU is potentially interconnected with any other LAU, as in a fully connected network. Several possible numbering schemes were considered and are now discussed.

A global number scheme on the bus is not suitable especially because it is prone to network wide collisions - it is not easy to ensure an ordered assignment of channels between multiple LAU's.

Another scheme considered was based on the reservation of a well defined set of channels for each LAU pair. In particular, a channel number could be formed as a basic number and a prefix, identifying the connected pair. The prefix could simply be the destination LAU address at the source or the source LAU address at the destination.

The collision problem is now reduced (to the pair) but the number of channels/LAU pair is not easy to determine. Two limits are obvious:

(a) the number of channels/LAU pair = number of X.25 logical channels. This is highly redundant (about 90 % on average if 10 LAU's are considered);

(b) the number of channels/LAU pair = 1. This is highly restrictive and nothing in the protocol allows this assumption (in particular we may consider several serial links managed by a single LAU or a LAU acting as a gateway

interface or a tandem unit).

Thus, another mechanism which overcomes all the above drawbacks was adopted. During call set-up, the two LAU's involved in the call exchange information concerning the logical channels just assigned at each X.25 interface. This does not violate the channel independence at both X.25 interfaces since this mechanism is completely transparent to the DTE's.

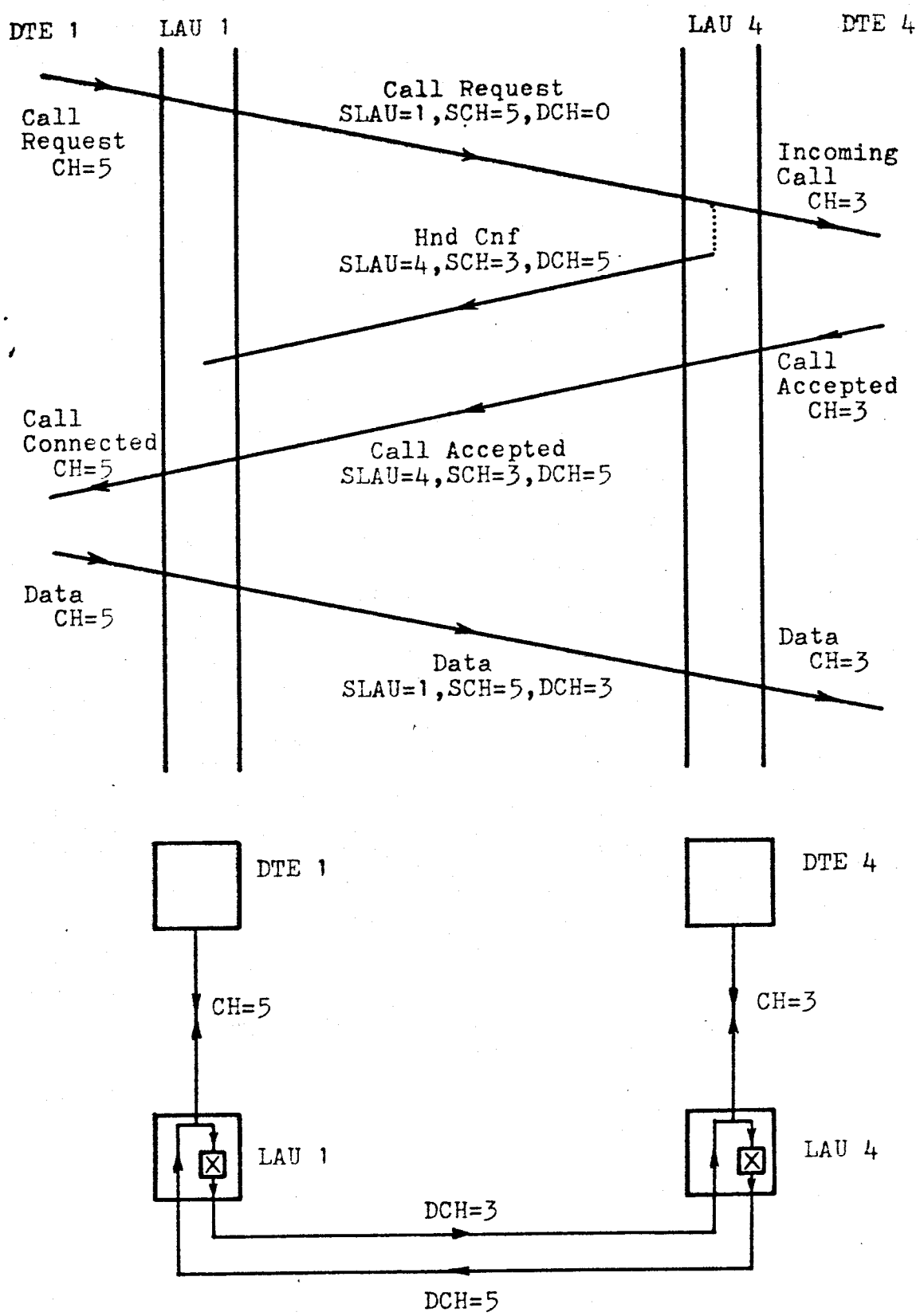
A sequence of operations using the adopted scheme is illustrated in Fig.5.6. The calling LAU transfers information concerning its address (source LAU) and its X.25 logical channel (source channel) in the Call Request packet. This is inserted in the two fields normally reserved for the frame header and unused at the packet level. At the called LAU, if the call may be forwarded (Incoming Call) the newly assigned X.25 logical channel is mapped to the remote LAU and the remote channel, and is then returned to the calling LAU, which then completes its own call mapping. Once this handshake is completed, destination channels may be directly addressed.

In conclusion:

(a) an internal channel is an extension of the X.25 channel; hence, no concern exists about the number of internal channels to implement;

(b) a combined state accounting for the link and bus interface conditions may be defined;

(c) no channel mapping is required when receiving a packet from the remote LAU since the destination channel is directly addressed;



SLAU = Source LAU
SCH = Source Channel
CH = X.25 logical channel
DCH = Destination Channel

Fig. 5.6 - Call set-up channel mapping

(d) no internal collisions arise during call set-up, because no channel is internally addressed (assigned) by the calling LAU.

This scheme has, however, some implications that must be analysed. A logical channel at the X.25 interface cannot be reused by the DTE before its extension is cleared. Also, a call collision at the X.25 interface will result in both calls being cleared - the Incoming Call, as determined by the protocol, and the Call Request due to the remote assignment of the same channel to the Incoming Call. This is not serious as a collision is restricted to the last available channel in the X.25 interface.

The possible reception of packets (Clear Request or protocol error) from the DTE before the channel handshake is completed must also be considered and the appropriate action delayed.

5.3.2 Virtual circuit management

The call set-up mechanism is purely end-to-end although an extra packet (Handshake Confirmation) must be generated by the remote LAU for internal channel handshake purposes, as described.

If the call cannot be accepted by the remote LAU (busy, no X.25 logical channel available, etc) a Handshake Reject packet is returned. The latter is similar to a Clear Indication (and will result in a Clear Indication being sent to the local DTE) but facilitates the handling of exceptional conditions (protocol error or DTE Clear Request) during the channel mapping phase.

As far as the call clearance is concerned, it is

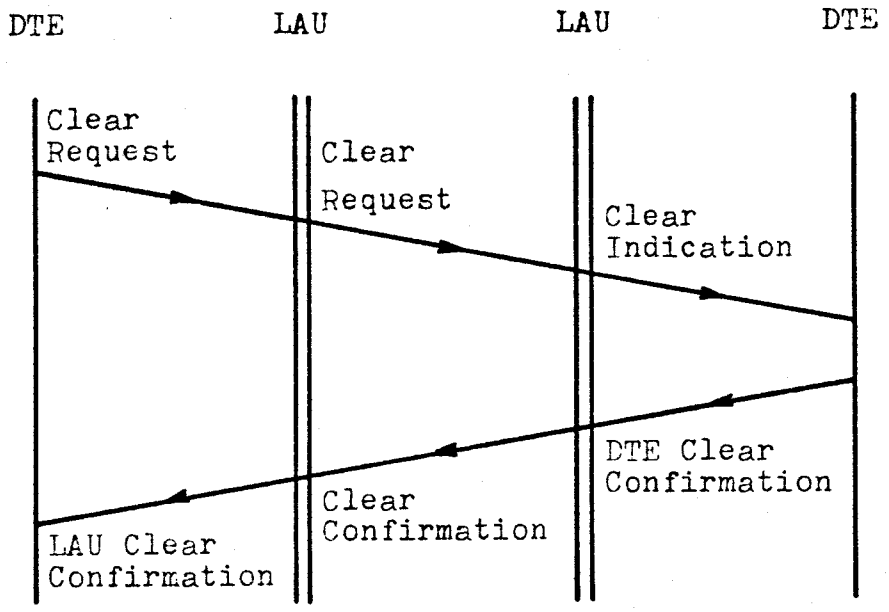
advantageous to consider two situations: for DTE initiated clearances, no local LAU Clear Confirmation should be generated before the remote LAU confirms the clearance. An automatic way of achieving this aim is to let the clearance have an end-to-end significance as shown in Fig. 5.7 (a). For LAU initiated clearances, to avoid similar problems (it would be impossible to synchronize Clear Confirmation packets from the DTE and the remote LAU) the clearance must first propagate to the remote LAU as shown in Fig. 5.7 (b).

In the data transfer phase all packets, other than those related to call clearance or protocol errors, are transparently transferred by the exchange. This is a condition arising from the implemented flow control mechanism. No supervisory packets are generated, no sequence number validation is performed and so naturally no Reset packets are generated by the LAU's. Thus, a call reset has pure end-to-end significance.

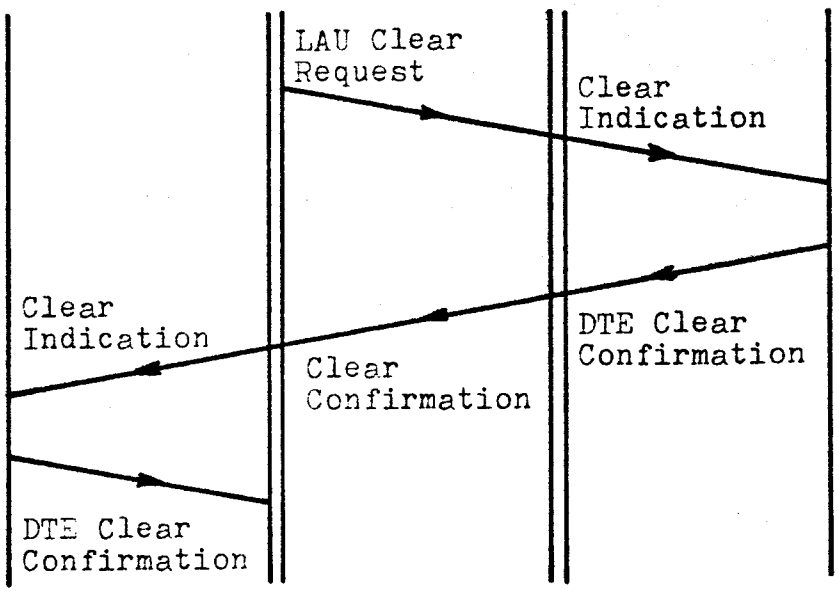
5.3.3 Restart

A DTE Restart Request is intended to clear all the calls in progress at the X.25 interface. These calls must also be remotely cleared, normally by individually clearing all the calls. This may be partially avoided, however, if Restart packets are internally exchanged between two LAU's to clear all calls currently established between them.

This means that several Restart packets may have to be generated for different LAU's although in number equal or less to the total number of Clear packets otherwise required. Also, each



(a) DTE clearing



(b) LAU clearing

Fig. 5.7 - Call clearing procedure

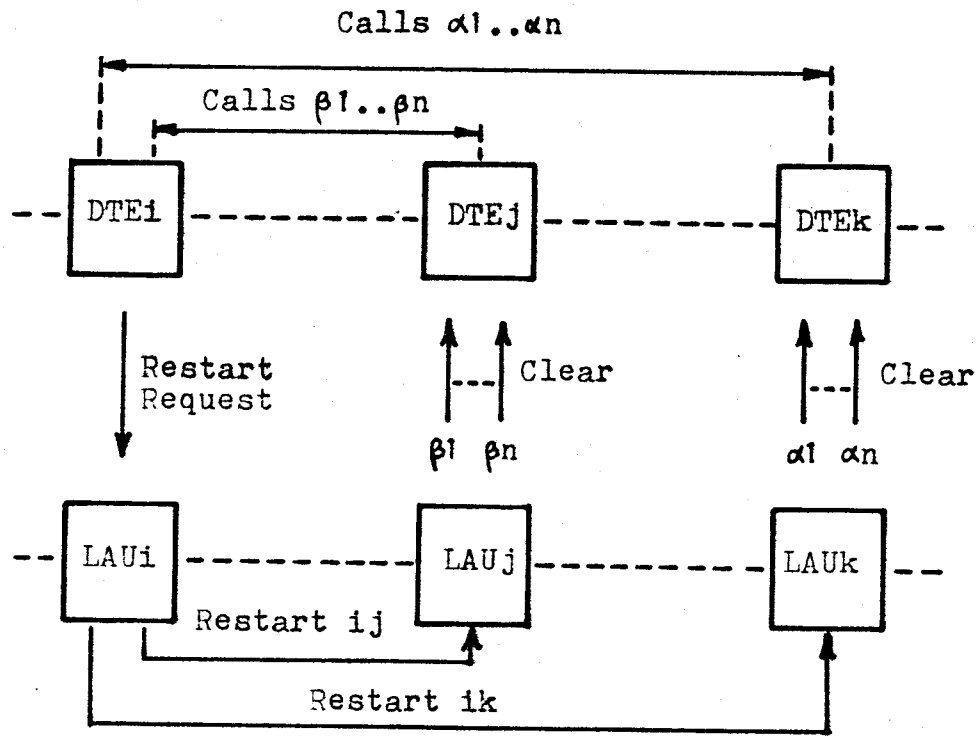
LAU receiving such a Restart packet has to search all calls related to the remote LAU and locally generate, at the X.25 interface Clear Indication/Confirmation packets as shown in Fig. 5.8 (a).

Since there is not a one-to-one relationship between a Restart and these Clear packets, it was decided to generate immediate Restart Confirmation packets either locally or remotely as indicated in Fig. 5.8 (b). This has the following implications:

(a) at the local X.25 interface, new calls may be generated by the DTE after receiving the LAU Restart Confirmation (the DTE sees all the channels in the ready state). They will be forwarded by the local LAU, however, only if they are addressed to a remote LAU that has completed the Restart procedure, thus ensuring that no channel contention will arise. Moreover, this does not make the establishment of new calls depend on the synchronization of the multiple Restart Request/Restart Confirmation packets;

(b) at the remote LAU's, all corresponding calls are presumed to be broken and subsequently only local clearances will take place.

The use of internal Restart packets is also convenient under other circumstances associated with LAU initialization and synchronization - power on, manual reset, console command (EMU), LAU/LAU failure or a link failure that must be recovered at a higher level.



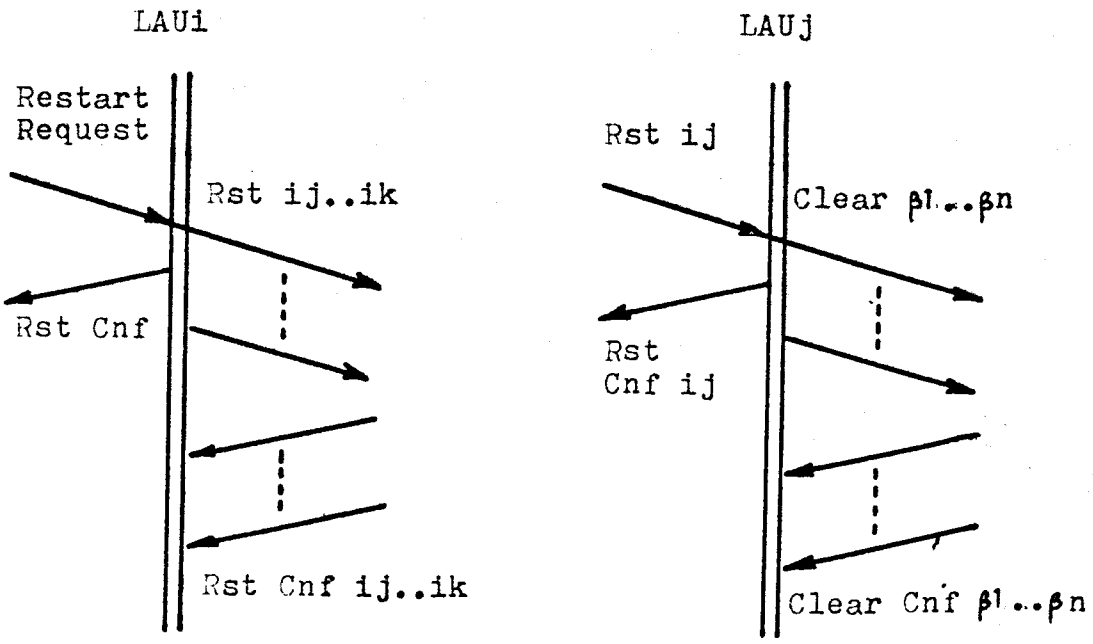


Fig. 5.8 - Restart procedure

5.3.4 Recovery mechanisms

We shall consider recovery actions other than those resulting from detected protocol errors and which are based on the implementation of a time-out mechanism. Three types of packets across the X.25 interface must be protected by a time-out. However, because retransmission of these packets is not convenient, as discussed, the following actions are taken:

(a) Incoming Call failure - the call is locally and remotely cleared by the LAU in the normal way;

(b) Clear Indication failure - a Clear Confirmation is first sent to the remote LAU (if the clearance had been remotely initiated), then a Diagnostic packet is sent to the local DTE and the channel returns to the ready state; further recovery actions, if required, are defaulted to a later stage, when the channel is reassigned;

(c) Restart Indication failure - recovery is defaulted to the link level (disconnection) as discussed in Section 5.4.

Concerning the LAU/LAU interface, several time-out mechanisms are implemented.

Call Request packets during the channel mapping phase and LAU/LAU Restart packets are protected by a time-out interval (small compared with that implemented at the X.25 interface) and the absence of the expected response means that either the remote LAU failed (or is simply OFF) or an abnormal condition has occurred. All calls to the remote LAU are then locally cleared and a long term recovery is initiated: periodic Restart packets

are generated until a response is obtained and while this process is in progress, no calls to the remote LAU will be accepted.

A recovery mechanism ("garbage collection") is also used to monitor the activity in all channels with calls currently outstanding to remote LAU's. The absence of any expected response during the call set-up or clearance phases or of any packet transfer during the data phase is considered as a remote LAU failure and the above resynchronization mechanism is enabled.

5.3.5 Channel state description

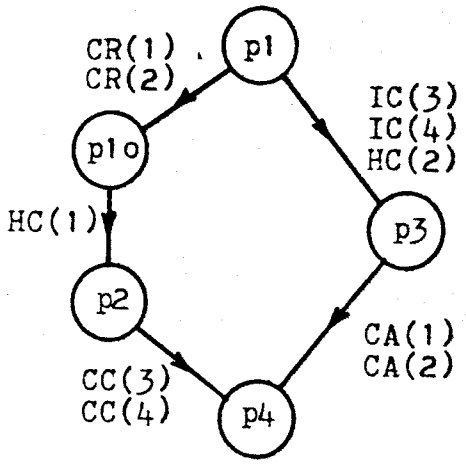
The previous discussion naturally leads to the consideration of a single state description and state transition on a logical channel basis, within a LAU. However this may be overruled by more general conditions (Restart, for example), in which case a global LAU state must be considered.

The basic set of channel states defined in the protocol (p1 to p7) and referring only to the X.25 interface was thus enlarged to account for protocol and time-out error conditions as well as internal control and recovery procedures. A virtual circuit is therefore characterized by a single state as represented in Fig. 5.9 but, when relevant, it may be described by two substates referring to the X.25 and LAU/LAU interfaces.

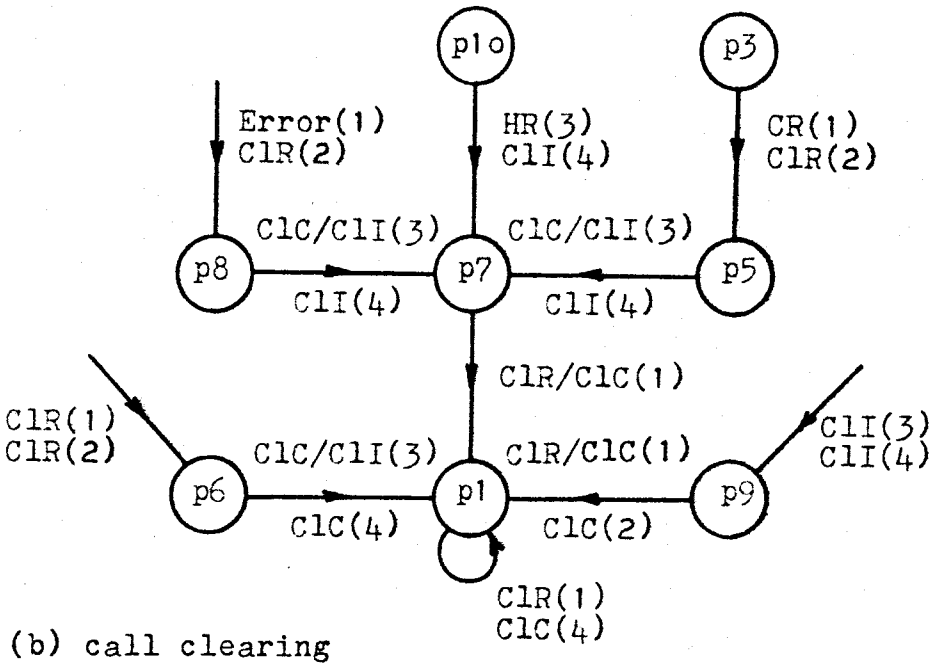
A state transition diagram thus including the activity in both interfaces is shown in Fig. 5.10. Only the call set-up and clearing phases are represented and for clarity some state transitions are omitted and instead the enabling condition is considered (for example, transitions to state p8 as a result of a

p1	ready (channel not assigned)
p2	local DTE Call Request/ remote channel mapping complete
p3	remote DTE Incoming Call
p4	data transfer (call established)
p5	call collision/ Incoming Call being remotely cleared
p6	local DTE Clear Request/ call being remotely cleared
p7	local LAU Clear Indication/ call remotely cleared
p8	call being remotely cleared (local LAU clearance/ error)
p9	remote Clear Request
p10	local DTE Call Request/ channel mapping phase
p11	Clear Indication failure (p7)
p12	Incoming Call failure
p13	Clear Indication failure (p9)
p14	initiate local LAU clearance/ call remotely cleared
p15	confirm local DTE clearance/ call remotely cleared

Fig. 5.9 - Logical channel state definition



(a) call set-up



(b) call clearing

CR=Call Request
IC=Incoming Call
CA=Call Accepted
CC=Call Connected

ClR=Clear Request
ClI=Clear Indication
ClC=Clear Confirmation
HC =Handshake Confirmation
HR =Handshake Reject

Packet transfer: (1) from local DTE
(2) to remote LAU
(3) from remote LAU
(4) to local DTE

Fig. 5.1o - Implemented logical channel state transition

protocol error or to state p9 as a result of a remote call clearance).

5.4 Interface between the Link and Packet layers

Normally, the exchange of packets across the level 2/level 3 interface should be enough to ensure that the link and packet layers are independently performing their functions in the hierarchy. Moreover, each level has its own recovery mechanisms which, in particular, may be called as a result of failures or recovery actions at lower layers; for instance, a failure at the link layer (meaning that the logical link is not operational) would probably induce, at the packet layer, a time-out recovery.

This is not, however, the best approach to the problem and a more efficient and quick signalling between layers is necessary. Without compromising its efficiency, this signalling must be kept to a minimum and it must not create interference between layers. It must then be performed exclusively at the interface between layers.

5.4.1 Link level failure

The only requirement for level 3 packet transfer is that the logical link must be set-up and in the information transfer phase. This places the problem of whether or not the link must be set-up before any information transfer is expected. The role of the DTE's and the LAU's is quite different in this respect.

Several strategies are possible in what concerns the DTE's, for example:

(a) the DTE may set-up the link only for specific call sessions, thus enabling communications at the packet level and above. Before this happens it may "listen" to the LAU (and accept or refuse any attempts to set-up the link) or not (receiver disabled, for instance);

(b) the DTE may set-up the link with anticipation (for instance, after enabling the receiver and transmitter), thus creating the conditions for remotely initiated packet level communications.

The role of the LAU's is more predictable. Once the link is set-up and because generation of calls is external to the LAU's, a link disconnection should not be initiated by a LAU unless irrecoverable errors so determine. The link must therefore be kept set-up until the DTE disconnects it. On the other hand, possibility of setting-up the link is not disallowed by a LAU.

In view of the above considerations the following approach is adopted in the LAU's:

(a) the packet layer assumes the link is set-up. No explicit request is made and packets are simply exchanged across the link/packet interface;

(b) existence of Information frames (packets) within the link layer is an implicit request for link set-up. If the link is still logically disconnected, a set-up will be attempted and, if successful, packets will begin to flow on the

serial link. If not, level 2 enters a "failure" state and this information is signalled to level 3;

(c) only when this failure is explicitly signalled does the packet level assume that the link is not operational and all current calls are cleared. After doing so, the whole process may repeat.

This is a conservative approach. The LAU as a part of the subnetwork may continuously receive remote calls and must handle them. The safest procedure therefore is to reject new Incoming Calls only after a new link set-up is attempted and fails. This signalling is also extended to include any level 2 disconnection during the information transfer phase.

5.4.2 Packet level failure

Signalling in the reverse direction may also be required. A DTE may wish, for example, to disconnect the link at the end of the current call(s). Normally this facility will not be required by a LAU.

There is, however, a situation that must be considered in a LAU: a level 3 Restart failure renders all logical channels unusable. Since Indication packets are not normally repeated, the avoidance of call blocking is not easily achieved by any packet level mechanism. We propose therefore that this condition should be removed from level 3 with an action performed outside level 3.

This is achieved by signalling a level 3 error condition and level 2 will then attempt a logical disconnection of the link.

Then, provided this disconnection is interpreted in the sense that has been considered here, it in fact means that all current calls will be cleared at both the DTE and the LAU. This was just the purpose of the Restart, of course, and when the link is later successfully set-up, there is no need to default to additional recovery actions at the packet layer (although they are not ruled out).

CHAPTER 6

PACKET SWITCHING EXCHANGE - SOFTWARE

In the previous chapters the main processing tasks inside a LAU were presented and discussed. In this chapter their software structure is presented together with the implementation of the scheduling algorithm utilized to achieve concurrency.

6.1 Concurrency control

In general, four basic stages of packet processing within a LAU must be considered:

(a) transfer of a frame to/from the serial link;
(b) processing of a frame (link level);
(c) processing of a packet (packet level);
(d) transfer of a packet to/from the parallel bus.

Each processing function is implemented as a separate process and hence, due to the possible accumulation of packets at different stages of processing, these processes communicate by means of queues, as shown in Fig. 6.1. This figure only shows the main streams of flow inside a LAU, whilst other forms of process communication associated with the outcome of particular events (a time-out, for example) will be considered in more detail within

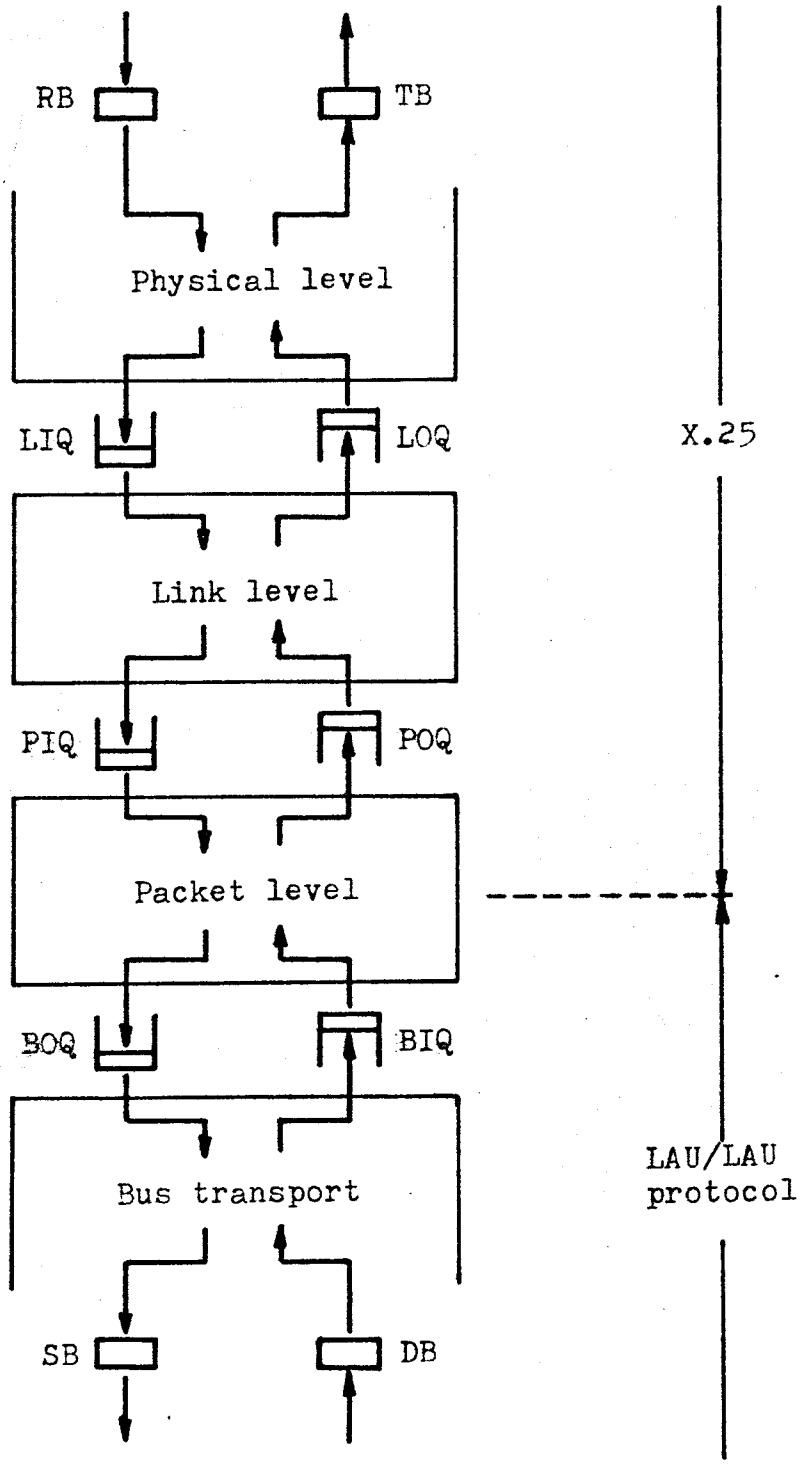


Fig. 6.1 - Interlevel communication

each processing level.

In addition, three global processes are utilized: an Initialization process, which performs basic initialization functions following a LAU reset; a Buffer Manager process which controls the granting of buffers to requesting processes; and a Timer Manager process which provides synchronization signals for the implementation of the various time-out mechanisms required by the protocol.

6.1.1 Scheduling algorithm

The choice of a scheduling algorithm and associated process priorities cannot be separated from the nature of the processing required and the possible patterns of traffic inside the exchange.

An overall objective is to maintain a smooth flow of packets through the exchange without keeping the link or the bus idle for long periods whenever packets already exist inside a LAU. Two questions are of interest: the priority assigned to each process and the time interval a process is allowed to run before being preempted by another process.

Short time-slices assigned to each process would certainly mean that some processes would simply preempt themselves to complete the current task, before others were ready to run. If, on the other hand, high priority processes were allowed to process several (possibly all) entries on their queues, groups of packets would reach the Link and Bus Output Queues in bursts, which is not the ideal arrival distribution (packets then compete

for bandwidth rather than processor or buffer resources). Moreover, accumulation of packets in intermediate queues tends to starve the output processes, thus increasing average packet delays inside a LAU.

Under these circumstances it was decided:

(a) to assign the same priority to all processes (except for the link process which is interrupt driven);

(b) to schedule the processes in a round-robin manner;

(c) to allow each process to access and process a single entry on its queue (if the queue is not empty) during each scheduling cycle.

This has several consequences: no queueing of packets will take place on intermediate stages and the arrival of packets at the output queues follows a more uniform pattern, since no accumulation of packets is enforced other than that determined by external traffic conditions. The proposed algorithm allows, on average, lower and more uniform bus transfer delays, a higher probability of completing bus transfers when the bus transmitter process is scheduled and likely lower processor idle times/packet processed. Moreover, the time slice assigned to each process, although variable, is bounded and in fact it is an important figure in assessing the exchange performance. This is discussed in Chapter 7.

6.2 Link Access Unit Initialization

Following a manual or power on reset, each LAU is completely initialized. This includes:

(a) setting the interrupt mode;

(b) initialization of the buffer pool and queues (counters, pointers);

(c) initialization of link and packet level state information;

(d) initialization of the logical channel tables (state and remote mapping) and the remote LAU tables (state, number of calls, etc);

(e) enabling the generation of Restart packets to all other LAU's, for LAU synchronization;

(f) initialization of all information concerning the Exchange Management Unit control functions;

(g) reset and initialization of peripheral controllers (PIO, CTC, SIO);

(h) initialization of time-out counters and link receiver and transmitter registers and counters.

When partial reinitialization is required (link level, for example) some of the above functions are locally activated (for example reset and initialization of the SIO).

6.3 Buffer Manager

Although several levels of processing are carried out inside a LAU, a single pool of buffers, shared by all processes is utilized and packets are transferred between queues by simply transferring buffer pointers between them.

To reduce processing overheads, buffers are of single fixed size, as determined by the maximum packet length supported. Moreover, prior to its release, a buffer may be reused by a process holding it, thus reducing pool accesses and allowing a tight coupling between the outcome of an event and the corresponding action (reception of a packet and generation of a response, for example). Possible inefficiency in the use of buffer storage space is reduced by the fact that shorter frames/packets are less frequent (control packets) or of shorter life (supervisory frames).

A dynamic buffer allocation scheme was implemented, as determined by the described flow control mechanism. Buffers are assigned on demand to requesting processes since:

(a) buffer requirements are easily predictable, as a result of the cyclic nature of the processing (no new processes are ever created or destroyed);

(b) no complex buffer reservation schemes are necessary, since no process places high and permanent demands. Under steady conditions and considering only the packets in transit through the LAU, the average rate at which new buffers are required is the same as the rate at which buffers are released.

Moreover, a priority rule must be implemented in order to restrict traffic during busy periods (flow control).

6.3.1 Buffer requirements

We may now summarize the buffer requirements placed by the link level process:

Firstly, a buffer is required for the generation of SARM/DISC commands. As stated by the protocol, only one SARM or DISC is allowed outstanding until a UA is received or a time-out occurs. On the other hand, it is advantageous to guarantee that a SARM or DISC is generated when required, irrespective of any possible buffer shortages. For these reasons, a private buffer pool (one buffer) is utilized for exclusive generation of SARM/DISC control frames.

Secondly, for acknowledgment or recovery purposes, response frames may have to be generated following the receipt of a command. Although on average a small number of buffers would be required, its number is not easily determined since it depends on traffic and frame length distributions. Instead of using a private pool for this purpose, therefore, it was decided to reuse, when possible, the same buffer to generate the response. The only situation when this is not possible is when an Information frame is accepted and generation of a RR supervisory frame required (this is next discussed), and this mainly occurs when traffic is unbalanced and hence when buffer shortages are less probable.

The buffer reqirements placed by the packet level are of two

main types: firstly, for the generation of multiple Clear or LAU/LAU Restart packets and these may be handled in a structured way according to the channel number or the LAU address. Secondly, during call set-up, an extra buffer is required to return channel mapping information to the calling LAU; since during a busy period the call should not be accepted (thus being directly cleared) this extra buffer requirement is thus easily accomplished.

Thus, the highest buffer demands are from the link receiver process (external offered traffic) and from the bus receiver process (exchange carried traffic). Less demanding requests are made by the link level process (for RR generation) and the packet level process during call set-up. Quite infrequent requests result from LAU/LAU control procedures and packet level time-out recovery.

6.3.2 Buffer pool structure

Since there are mainly two classes of traffic (input and transit) the pool is structured into two segments as shown in Fig. 6.2 (a) - NOFB means the number of free buffers generally available for all traffic and NOGB means the number of buffers granted (reserved) for transit traffic and used during a busy period only.

While in a not busy condition, NOGB=NGBØ but once the busy condition is entered, NOFB=Nr buffers are assigned for link reception (Nr<=NFBØ) with the remaining NOCB=Ng buffers assigned for transit traffic. We therefore define NØ=NCBØ+NFBØ and a busy

condition is entered when the total number of unused buffers, N, falls below N0 (N<N0 or NOFB<NFB0).

In general, the link receiver process has free access to the pool while NOFB>0 and the bus receiver process while NOGB>0. All other requests are only served during non busy periods; otherwise default actions are taken by the corresponding processes - for example, discard of Information frames and generation of a single RNR frame at the link level, call set-up restrictions at the packet level and delay of all packet level control or recovery actions, for which buffers have been requested.

A Buffer Manager process is therefore implemented to provide the following synchronization functions:

(a) to assess a busy condition as a result of the asynchronous requests of the link receiver process;

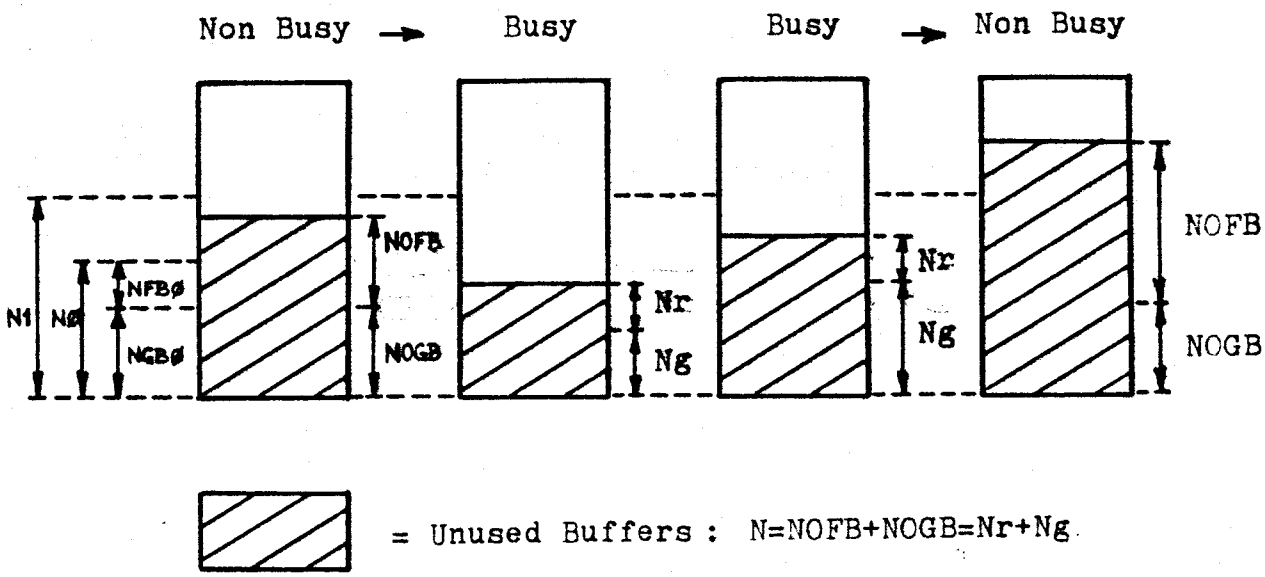
(b) to distribute the available buffers (Nr/Ng) during a busy period;

(c) to assess the conditions that allow the removal of the busy condition (N>=N1).

Moreover, detection of a busy condition is also performed during each synchronized access to the pool.

To avoid contention in the access to the pool and to minimize the transfer of received frames to the Link Input Queue, the Buffer Pool and the Link Input Queue share the same circular entry space as shown in Fig. 6.2 (b) and the following rules are observed:

(a) the receiver process accesses the pool from the top;



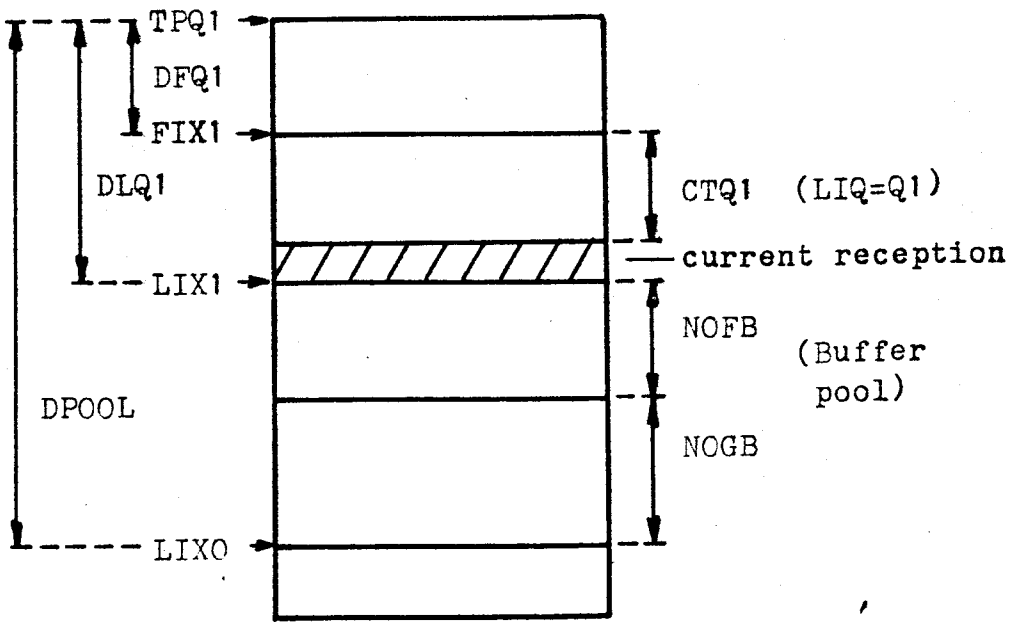


Fig. 6.2 - Buffer pool structure

(b) synchronized access by the other processes is from the bottom;

(c) the release of buffers follows the same rule.

In the current implementation the size of the pool is NBUF=15 and the busy/non busy thresholds are: NØ=6 (NGBØ=4; NFBØ=2) and N1=8. These correspond to Ng=3 and Ng=6 (2Ø and 4Ø % of the pool size respectively), at the moment of the transitions, considering Nr=2. The choice of NFBØ=2 is intended to avoid the loss of a second frame closely following a previously received but not yet processed frame; for example, two frames may be separated by a single flag, corresponding to Ø.8 ms at 9.6 Kbaud, which is usually insufficient to process a received frame and release the buffer.

6.3.3 Buffer transfer routines

Four general purpose Buffer transfer routines are used:

GETBUF - with entry parameters identifying the queue and the queue entry, which accesses the queue and returns the corresponding buffer in the current buffer (CURBUF);

PUTBUF - with similar entry parameters, which transfers to the queue the buffer referred to in CURBUF;

GTPOOL - which uses GETBUF to access the pool and tests a possible non busy/busy transition as a result of the pool access;

RELBUF - which uses PUTBUF to return a buffer to the pool

assigning the released buffer to NOFB (not busy) or to NOGB (busy).

All queues are of similar size and can support the maximum number of buffers (thus avoiding overflows) ; this does not place particular storage overheads since the number of queues is limited - a total of 6*32=188 bytes of RAM is required to hold all queue tables.

6.4 Timer Manager

The Z80 CTC is programmed to generate interrupts at 100 ms intervals. This interval is then used as the basic timer unit to generate the various time-out signals necessary within each processing level.

At the link level a single time-out interval (1 s) is required. The corresponding time-out counter is updated and tested for the time-out condition provided that it has been enabled by the link level (timer running).

At the packet level several time-out mechanisms are required:

(a) to protect Indication packets across the X.25 interface (15 s);

(b) to protect Call Request (channel mapping phase) and Restart packets across the LAU/LAU interface (1 s);

(c) to initiate LAU/LAU recovery actions (30 s).

All these timers are free running, updated every 100 ms and thus periodically generate the corresponding time-out signals. These are later used to trigger internal mechanisms on a channel or LAU basis as explained in Section 6.6.

6.5 Link level processing

Several solutions have been proposed for the implementation of the link level of X.25 [68,69,70]. Perhaps the most widely referred to being the one described in [71], which uses the concepts of processes and monitors of Concurrent Pascal.

In our implementation, the link level process is comprised of five main procedures which are called in a strict sequential order corresponding to a hierarchy of functions that takes into account the tight-coupling existing between Primary and Secondary components.

As has been described, three different frames (SARM, DISC and Information) may be issued by the Primary component either to transfer information or perform link management functions. An identification of the conditions (enabling predicates) that determine state transitions, therefore becomes a central aspect of the implementation. This is represented in Fig. 6.3 and shows that Primary functions may be enabled in a variety of independent ways, thus appearing distributed over several procedures.

This information, complemented with the general protocol rules, was used to define the relevant link state information as shown in Fig. 6.4 and the link level procedures concerning frame reception described in Table 6.1.

COMMAND	FUNCTION	ENABLE PREDICATE	COMMENT
SARM	SET-UP	Reception of SARM (PRIM=0)	Remote Set-up
		INF frame available for transmission (PRIM=0; LF=0)	Local Set-up (See ** for LF=1)
	RESET	Reception of CMDR (PRIM=1)	Remote Reset request
		Reception of invalid Response (PRIM=1)	Local Reset
		Time-out/INF frame (RETCNT=RETMAX; RICNT<RIMAX)	Local Reset (Max Nbr INF retransm)
	SET-UP/ RESET	Time-out/SARM (DIS=0; RETCNT<RETMAX)	SARM retransmission (See * for DIS=1)
DISC	DISCONNECTION	Reception of DISC (PRIM=1 or SEC=1)	Remote Disconnection (may be delayed - *)
		Packet level failure (L2/L3 Interface signal)	Local Disconnection (may be delayed - *)
		Error (complete loss of protocol context)	Local Disconnection (may be delayed - *)
		Time-out/INF frame (RETCNT=RETMAX; RICNT=RIMAX)	Local Disconnection (INF failure)
		Time-out or UA received/SARM (DIS=1)	* Local Disconnection delayed by SARM
	SYNCHRONIZATION	Reception of Command (LF=1)	Link failure/Remote Primary active
		INF frame available for transmission (PRIM=0; LF=1)	** Disconnection before Set-up
	DISCON/ SYNCHR	Time-out/DISC (RETCNT<RETMAX)	DISC retransmission
INFORMATION	PACKET TRANSFER	INF frame available and Information transfer enabled (IAL=1)	INF transfer when Primary Set-up and no restrictions: window overflow, recovery, Remote Busy, Reset

Fig. 6.3 - Link Primary functions

	State	
LF	Link Failure	LF=1: Link failed (time-out) or was down
DIS	Disconnection	DIS=1: real Disconnection enabled/in progress
PRIM	Primary state	PRIM=0: Primary disconnected/connecting PRIM=1: Primary connected/disconnecting
SEC	Secondary state	SEC=0: remote Primary disconnected/disconnecting SEC=1: remote Primary connected/connecting
SEC0 SEC1 SEC2	Secondary substates	SEC0=1: CMDR sent SEC1=1: RNR sent SEC2=1: REJ sent
COM	Command	COM=1: SARM or DISC sent
SD	SARM/DISC	SD=0: SARM sent; SD=1: DISC sent (if COM=1)
IAL	INF enabled	IAL=1: INF transfer enabled (⇨ COM=0, PRIM=1)
WAIT	Wait state	WAIT=1: Command transfer temporarily disabled
REC	Recovery	REC=1: Information frame time-out recovery
PS	Poll Sent	PS=1: Information frame with P=1 transmitted
RBY	Remote Busy	RBY=1: RNR received and not yet cleared
PGB	Piggyback	PGB=1: Piggyback acknowledgment required
	Timer	
T2OUT	Time-out	T2OUT=1: Link level time-out
T2ON	Timer on	T2ON=1: Link level timer is running
	Sequence number variables	
MAXNS	Current maximum N(S) outstanding; (may be ≠ last transmitted)	
MINNS	Last released Inf frame; (may be ≠ last acknowledged)	
LASTNR	Last received N(R)	
SEQNBR	Receive/Send State Variables V(R)/V(S)	
	Retransmission counters	
RETCNT	Current number of retransmissions of SARM/DISC/INF	
RICNT	Current number of Resets due to repeated time-out of same INF	

Fig 6.4 - Link level state variables

Table 6.1

COMMAND RECEIVED	LINK STATE					
	LF=1	LF=0				
		DIS=1	DIS=0			
			P/S=00	P/S=01	P/S=10	P/S=11
SARM	Send DISC	---	Send UA SEC=1 V(R)=0 Send SARM	Send UA SEC=1 V(R)=0 Send SARM	Send UA SEC=1 V(R)=0	Send UA SEC=1 V(R)=0
DISC	Send UA Send DISC	Send UA SEC=0	Send UA SEC=0 Clear Set -up delay	Send UA SEC=0 DISCN	Send UA SEC=0 DISCN	Send UA SEC=0 DISCN
INFORMA TION	Send DISC	---	DISCN (protocol error)	NRI Process INF (Tab 6.1.1)	DISCN (protocol error)	NRI Process INF (Tab 6.1.1)
INVALID COMMAND	Send CMDR	Send CMDR	Send CMDR	Send CMDR	Send CMDR	Send CMDR

RESPONSE RECEIVED	LINK STATE		
	COM=1	COM=0	
		PRIM=1	PRIM=0
UA	Process UA (Tab 6.1.2)	RESET	---
SUP	---	Process N(R)/ SUP (Tab 6.1.2)	---
CMDR	log CMDR reason	log CMDR reason RESET	---
INVALID RESPONSE	---	RESET	---

P/S : Primary/Secondary state

NRI : validates and processes N(R) in INF frame (Tab 6.1.2)

DISCN : enables link Disconnection (DIS=1), signals Disconnection to level 3 (LD=1) and clears Information queue (POQ)

RESET: initiates a link Reset (SARM transmission)

Table 6.1.1

SEC0 / SEC1 / SEC2		INFORMATION			
		Poll Received P=1		Poll received P=0	
		N(S)=V(R)	N(S)≠V(R)	N(S)=V(R)	N(S)≠V(R)
0	00 BUSY	Release SEC1=1;SEC2=0 Send RNR(F=1)	Release SEC1=1 Send RNR(F=1)	Release SEC1=1;SEC2=0 Send RNR	Release SEC1=1 Send RNR
	01 BUSY				
	11 BUSY			Release SEC2=0	Release
	10 BUSY				
	00 NOT BUSY	Accept SEC1=0;SEC2=0 Send RR(F=1)	Release SEC1=0;SEC2=0 Send RR(F=1)	Accept SEC2=0;PGB=1	Release;SEC1=0 SEC2=1;Send REJ
	01 NOT BUSY				Release
	11 NOT BUSY			Accept SEC1=0;SEC2=0 Send RR	
	10 NOT BUSY				Release;SEC1=0 SEC2=1;Send REJ
1	-- ---	Send CMDR	Send CMDR	Send CMDR	Send CMDR

Table 6.1.2

UA			
WAIT=1	WAIT=0		
	SD=0 / DIS=0	SD=0 / DIS=1	SD=1
--- (SARM/DISC in LOQ)	STP;RETCNT=0 COM=0;RBY=0 PRIM=1;IAL=1 Reset Sequence Variables Reset POQ	STP;RETCNT=0 COM=0;RBY=0 PRIM=1 TDIS	STP;RETCNT=0 COM=0;RBY=0 SEC=0;DIS=0 PRIM=0;LF=0 WAIT=1;START (delay set-up)

	N(R)/INF	RR	REJ	RNR
RBY=1	---	ACK;ENDREC RBY=0 IAL=1 (WAIT=0, NIF1<window)	ACK;ENDREC RBY=0 MAXNS=MINNS STP(WAIT=0);RETR	ACK;ENDREC RBY=1 RETCNT=RICNT=0 IAL=0 (NIF1>0)
RBY=0	ACK			

STP : Stops level 2 timer (T2ON=0)

START : Starts level 2 timer (T2ON=1)

TDIS : Initiates DISC transmission

ACK : releases acknowledged Information frames (if REC=0)

ENDREC : reenables Information transfer (IAL=1) and defaults transmission to the first unacknowledged frame (if valid F=1 received)

RETR : enables Information frame transmission after LOQ is cleared of Information frames (IAL=1; WAIT=0); otherwise WAIT=1.

It should be noted that a WAIT state was created to temporarily restrict the generation of new commands; this is mainly intended to avoid buffer contention problems and to deal with some aspects of the recovery mechanism and the link management. Moreover, to avoid Primary blocking due to repeated transmission failures, for example, a time-out mechanism associated with the physical transmission of commands (and not with their acknowledgment) is implemented during this state.

6.5.1 Link level procedures

A brief description of the main procedures now follows:

Interface with the packet level

This tests the packet level requirement for a link disconnection signalled through the interlevel interface and, if signalled, enables the link disconnection routine, which clears all Information frames from the Packet Output Queue.

Reprocess transmitted frames

Transmitted frames are reprocessed in order to:

(a) release buffers holding response frames;

(b) release a buffer holding a SARM/DISC command;

(c) transfer to the retransmission list buffer(s) holding Information frame(s) and update the maximum outstanding send sequence number (MAXNS);

(d) update variables associated with the WAIT state.

Process received frames (LIQ)

Frames received across the serial link are processed as follows:

(a) length, address and control fields are decoded and validated;

(b) appropriate actions are taken according to Table 6.1 and include: response generation, Primary/Secondary coupling (link management functions, for example), error detection and enabling of recovery actions (link reset, for example);

(c) transfer of valid Information frames to the packet level (PIQ).

Time-out (level 2)

In general, except for the cases referred to in the text, the level 2 timer is kept running only when a command is outstanding and is stopped (and eventually restarted) after a valid acknowledgment is received. A maximum number of retransmissions (3) is supported, after which more powerful recovery procedures are initiated.

Thus, following a time-out, one of the following actions is taken:

(a) retransmission of the timed-out frame is enabled, unless one of the remaining conditions applies;

(b) reset of the link is initiated if retransmission of an Information frame failed;

(c) disconnection of the link is initiated if it had been delayed by an outstanding SARM or if repeated Information frame failures have not been recovered by link resets (b);

(d) link level reinitialization is enabled (failure routine) following command (SARM/DISC) or transmission failures (the latter during the WAIT state);

(e) the set-up restriction introduced after a Primary disconnection is cleared, if present.

Information Manager

The Information Manager controls the processing of Information frames and their transfer to the Link Output Queue. It:

(a) assesses the conditions that, during the information transfer phase, enable/disable the flow of Information frames (window overflow, time-out recovery, remote busy, reset);

(b) assesses the need for a link set-up, as explained in Section 5.4;

(c) assesses the need to generate a supervisory RR frame when acknowledgment piggyback is impossible (no Information frame available or information transfer not allowed).

6.6 Packet level processing

Processing of received packets is based on two main event/state tables: one for the X.25 interface (Table 6.2) and the other for the LAU/LAU interface (Table 6.3).

Most of the protocol errors are detected across the X.25 interface and therefore they do not propagate into the exchange. Moreover, the internal protocol and the synchronization of states between LAU's allows a simplification of the Table 6.3 as follows: in some states it is either impossible for the reception of a packet (Call Accepted in state p4, for example) or its reception is irrelevant since it must be discarded (Data in state p6, for example); in both cases no further action is taken (the packet is discarded and no state change takes place).

Three types of state information are used as shown in Fig. 6.5. These are:

(a) general state information concerning global actions to be performed or that are in progress;

(b) information concerning each logical channel, which is stored in three tables, corresponding to the channel state and its mapping to the remote LAU and the remote logical channel;

(c) information concerning each LAU/LAU interface, namely

Table 6.2

X.25 Interface State	Packet received from DTE			
	Rst Request (Channel=Ø)	Rst Confirm (Channel=Ø)	Other Packet (Channel=Ø)	Other Packet (Channel≠Ø)
DCERST=1	DCERST=Ø TONRST=Ø	DCERST=Ø TONRST=Ø	---	---
DCERST=Ø	RST Rst Cnf (POQ)	DCERST=1;RST Rst Ind (POQ)	DCERST=1;RST Rst Ind (POQ)	Process pack (Tab 6.2.1)

RST : returns all logical channels to the ready state (p1) and initiates remote clearance of all calls currently established.

Table 6.2.1

Channel State	Packet type					
	Call Req	Call Acc	Clear Req	Clear Cnf	Data	Inv Pack
p1	p1Ø *Call Rq (BOQ)	p7 Clear Ind (POQ)	*p1 Clear Cnf (POQ)	p7 Clear Ind (POQ)	p7 Clear Ind (POQ)	p7 Clear Ind (POQ)
p2	p8 Clear Rq (BOQ)	p8 Clear Rq (BOQ)	p6 *Clear Rq (BOQ)	p8 Clear Rq (BOQ)	p8 Clear Rq (BOQ)	p8 Clear Rq (BOQ)
p3	p5 Clear Rq (BOQ)	p4 *Call Ac (BOQ)	p6 *Clear Rq (BOQ)	p8 Clear Rq (BOQ)	p8 Clear Rq (BOQ)	p8 Clear Rq (BOQ)
p4	p8 Clear Rq (BOQ)	p8 Clear Rq (BOQ)	p6 *Clear Rq (BOQ)	p8 Clear Rq (BOQ)	*p4 *Data (BOQ)	p8 Clear Rq (BOQ)
p5/p6 p8/p1Ø	p8 --- ---	p8 --- ---	p6 --- ---	p8 --- ---	p8 --- ---	p8 --- ---
p7	*p7 --- ---	*p7 --- ---	p1 --- ---	p1 --- ---	*p7 --- ---	*p7 --- ---
p9	p7 Clear Cf (BOQ)	p7 Clear Cf (BOQ)	p1 Clear Cf (BOQ)	p1 *Clear Cf (BOQ)	p7 Clear Cf (BOQ)	p7 Clear Cf (BOQ)
p11	p1 Diagnost (POQ)	p1 Diagnost (POQ)	p1 --- ---	p1 --- ---	p1 Diagnost (POQ)	p1 Diagnost (POQ)
p12	p8 Clear Rq (BOQ)	p8 Clear Rq (BOQ)	p6 *Clear Rq (BOQ)	p8 Clear Rq (BOQ)	p8 Clear Rq (BOQ)	p8 Clear Rq (BOQ)
p13	p11 Clear Cf (BOQ)	p11 Clear Cf (BOQ)	p1 Clear Cf (BOQ)	p1 *Clear Cf (BOQ)	p11 Clear Cf (BOQ)	p11 Clear Cf (BOQ)
p14/p15	p7 Clear Ind (POQ)	p7 Clear Ind (POQ)	p1 Clear Cnf (POQ)	p7 Clear Ind (POQ)	p7 Clear Ind (POQ)	p7 Clear Ind (POQ)

(1) new channel state (*pi = same state)
(2) packet transferred (*pack = same packet; --- = packet discarded)
(3) packet destination : local DTE (POQ) ; remote LAU (BOQ)

Table 6.3

LAUi state		Packet received from LAUi			Packet from EMU
RSTI	ONI	Rst Indicat	Rst Confirm	Other Packet	
Ø	Ø	ONI=1 Rst Cnf (POQ) BUSRST	RSTI=ONI=COMI=1 Rst Rq (BOQ) BUSRST	RSTI=ONI=COMI=1 Rst Rq (BOQ)	Process Monitor Request
	1			Process Pack (Tab 6.3.1)	
	COMI				
1	Ø		ONI=1	ONI=1	
	1		RSTI=COMI=Ø ONI=1;TONI=Ø		

BUSRST : Initiates local clearance of all calls with LAUi.

Table 6.3.1

Channel State	Packet type					
	Call Con	Clear Ind	Clear Cnf	Data	Hnd Cnf	Hnd Rej
p1,p7,p9	---	---	---	---	---	---
p11,p13	---	---	---	---	---	---
p14,p15	---	---	---	---	---	---
p2	p4 *Call Cn (POQ)	p9 *Clear In (POQ)	--- --- ---	--- --- ---	--- --- ---	--- --- ---
p3	--- --- ---	p9 *Clear In (POQ)	--- --- ---	--- --- ---	--- --- ---	--- --- ---
p4	--- --- ---	p9 *Clear In (POQ)	--- --- ---	*p4 *Data (POQ)	--- --- ---	--- --- ---
p5	--- --- ---	p7 *Clear In (POQ)	p7 Clear In (POQ)	--- --- ---	--- --- ---	--- --- ---
p6	--- --- ---	p1 Clear Cf (POQ)	p1 Clear Cf (POQ)	--- --- ---	*p6 Clear Rq (BOQ)	p1 Clear Cf (POQ)
p8	--- --- ---	p7 *Clear In (POQ)	p7 Clear In (POQ)	--- --- ---	*p8 Clear Rq (BOQ)	p7 Clear In (POQ)
p10	--- --- ---	--- --- ---	--- --- ---	--- --- ---	p2 --- ---	p7 Clear In (POQ)
p12	--- --- ---	p9 *Clear In (POQ)	--- --- ---	--- --- ---	--- --- ---	--- --- ---
Incoming Call accepted:		p3 *In Call (POQ) Hnd Cnf (BOQ)		Incoming Call rejected:		--- Hnd Rej (BOQ)

(1) new channel state (*pi = same state)
(2) packet transferred (*pack = same packet; --- = packet discarded)
(3) packet destination : local DTE (POQ) ; remote LAU (BOQ)

State

DCERST	DCERST=1: Restart Indication sent to local DTE
TONRST	TONRST=1: timer running for Restart Indication
SCN1	SCN1=1: recovery of logical channel(s) is enabled (channel time-out, remote LAU Restart or failure)
SCN2	SCN2=1: recovery of remote LAU(s) is enabled (LAU failure, initialization or synchronization)
BF	BF=1: remote LAU(s) failure while calls in progress

Counters

NRCH	number of channels currently in use (state ≠ p1)
STMC	number of channels with local timer protection (states: p3, p7, p9)
TONC	number of channels with remote timer protection (states : p2, p4, p5, p6, p8, p10)
SPCC	number of channels in recovery (SCN1=1) (states: p11, p12, p13, p14, p15)
BSCC	number of LAU's in recovery (SCN2=1)

Channel flags

CHTIM1	CHTIM1=1: channel has local timer protection
CHTIM2	CHTIM2=1: channel has remote timer protection
CHTON	CHTON=1 : timer is running (⇨ CHTIM1=1 / CHTIM2=1)
CHSCN	CHSCN=1 : channel is in recovery

LAUi state

ONI	ONI=1 : LAUi is ON; ONI=0: LAUi is OFF / failed
RSTI	RSTI=1: Restart sent (COMI=1) / enabled (SCNI=1)
SCNI	SCNI=1: Restart enabled (⇨ RSTI=1 and COMI=0)
COMI	COMI=1: Response expected (Restart / call set-up mapping)
TONI	TONI=1: LAUi timer is ON (⇨ COMI=1)
BFI	BFI=1 : LAUi failed while calls were in progress

Fig. 6.5 - Packet level state variables

number of calls in progress (total and in channel mapping phase), remote LAU state and Restart reason. These in turn are stored in four tables.

6.6.1 Packet level procedures

The packet level is comprised of five main procedures:

Interface with the link level

In the event of a link failure or a logical link disconnection, signalled through the interlevel interface, remote clearance of all calls in progress is activated. This is carried out by LAU/LAU Restart packets.

Process packet received from the DTE (PIQ)

Packets received from the serial link are processed as follows:

(a) the packet header is accessed and validated (minimum length, logical channel number range, and packet type);
(b) the channel state is decoded and the action and the new state are determined from the corresponding event/state table (Table 6.2).

Process packet received from the remote LAU (BIQ)

Packets received across the interchannel bus are processed as follows:

(a) the packet header is accessed;

(b) the channel state is decoded and the action and the new state are determined from the corresponding event/state table (Table 6.3);

(c) Exchange Management Unit commands are handled by a Monitor procedure. This is described in Chapter 7.

Timer

The packet level timer mechanism is different from that at the link level. Firstly, time-out intervals at the packet level are less critical and therefore are usually larger (by one or two orders of magnitude) and less precisely defined. Secondly, the current channel numbering is flexible enough to allow the implementation of up to 255 channels, which could introduce severe processing overheads if timer registers were kept and periodically updated and tested for each channel.

Instead, a timer is kept permanently running in the X.25 interface and at periodic intervals (T3=15 s) all logical channels requiring a time-out protection are scanned in order to synchronously update their timing conditions (start timer, time-out detection). Clearly, with this scheme time-outs are only detected in the interval [T3, 2*T3] or on average 1.5*T3.

However, this is not a severe restriction provided that T3 is accepted as an useful minimum upper bound.

Similar independent mechanisms are implemented in the LAU/LAU interfaces, for packet protection and recovery actions.

Scan

The scan procedure has two components:

(a) once a special condition arises (possibly involving several channels as a result of time-outs or LAU/LAU Restart), all the affected channels are first determined and later orderly scanned to generate the corresponding Clear Indication/Clear Confirmation packets;

(b) following a Restart at the X.25 interface or a link failure or as a result of LAU/LAU recovery mechanisms, all LAU's previously identified for recovery are orderly scanned to generate the corresponding LAU/LAU Restart packets.

6.7 Link and bus transfers

A description of the link and bus transfer processes was presented in Chapter 4. For link transfers and other interrupt driven processes, use of the Z80 vector interrupt mode and the daisy chain interrupt structure, allows the generation of an indirect call to the interrupt service routine corresponding to

the current highest priority interrupt source. For this purpose a table of start addresses is kept in memory (ROM) and this table is accessed by means of a pointer formed by the CPU Interrupt Page Address Register (loaded during LAU initialization) and the vector supplied by the interrupting device.

Since the Z80 CPU allows one level of context switching (by means of the alternate set of general purpose registers), this feature is used to reduce overheads in saving and restoring the program status at the beginning and end of each interrupt service routine (EXX and EX AF,AF' instructions). This naturally precludes the use of nested interrupts (unless more extensive status saving is performed) but is readily accomplished by reenabling the interrupts at the end of the service routine only (EI followed by a RETI instruction).

6.8 Implementation details

All programs have been written in Assembly language and developed and tested using the editing, assembling and debugging facilities of a Z80 based Microcomputer Development System, running under the CP/M Operating System. These are presented in Appendix B.

Each protocol level was developed and tested independently in the Microcomputer Development System and a communications board was added, which supported up to six serial links (three Z80 SIO controllers) and two independent timers (based on the Intel 8253) to simulate these functions in the exchange hardware.

Initially the interrupt routines were developed utilizing a

loop transmission around a single SIO. Later two similar Microcomputer Development Systems were connected through the exchange - they featured the same level 2 version as in the exchange and an oversimplified version of the packet level, mainly providing some facilities for call generation. However, it was possible to carry out significant tests of the exchange, in particular its recovery mechanisms. Incidentally, the extensive error detection schemes of X.25 proved helpful in the localization of run-time program bugs.

A parallel investigation was being carried out concerned with the implementation of X.25 software for use in a single-user system. This included the implementation in a high level language (Pascal) of the X.25 protocol and a Transport protocol closely following the Transport Service proposed by the British Post Office [72]. Additional facilities (printer server, mail box, file transfer) were also being provided.

Finally, therefore, two Microcomputer Development Systems were used as the environment for more complete tests of the exchange, which also included the use of the facilities provided by the Exchange Management Unit. The facilities provided in the latter are discussed in the next chapter.

A photograph of the development facilities used to test the prototype exchange is shown in Appendix D.

CHAPTER 7

EXCHANGE MONITORING AND PERFORMANCE EVALUATION

7.1 The Exchange Management Unit

The design of the Exchange Management Unit (EMU) and the set of functions it supports was derived from the following basic requirements:

(a) to monitor the activity within the exchange;
(b) to detect failures and malfunctions;
(c) to provide maintenance and debugging aids;
(d) to collect statistical information.

In addition, it was felt that the EMU should provide an easy interface with the operator and the interference with the normal processing activities within each LAU should be kept to a minimum to avoid performance degradation. Also, possible malfunctions of the EMU itself and in particular operator errors, should not leave a LAU in an uncontrolled or ill-defined state.

In an attempt to meet these aims, therefore, the EMU is implemented as an independent unit which is connected to the same Interchannel Unit as the LAU's. The various control functions are performed by each LAU as a result of commands entered at the EMU operator terminal. These are subsequently issued by the EMU and mainly operate on an enquiry/response basis.

7.1.1 Exchange Management Unit commands and their functions

Each command is usually addressed to a specific LAU. For synchronization purposes, however, some commands may use a global address mode and hence they are converted by the EMU into a sequence of identical, individually addressed, commands, since there is no support in the Interchannel Unit for packet broadcasting.

Each command specifies a well-defined action to be carried out by the addressed LAU. For simplicity, commands are identified by a single character and a set of parameters defining possible options (a zero default value is assumed, however, when not entered at the console). The list of commands and their functions are shown in Fig. 7.1.

Each LAU then takes appropriate actions as a result of each command, but the EMU may retrieve the information gathered as a result of past commands without affecting the current state of each LAU. Retrieval of this information is only actioned on request by the EMU. If more than one packet is required to convey the response, an internal handshake mechanism regulates the exchange of several commands/responses until the whole response is transferred.

7.1.2 Implementation details

The EMU software is known as the X.25 monitor and is implemented as an extended version of a MOSTEK DDT-80 monitor [73]. Several features of DDT-80 are utilized which include USART

P p1	Poll the specified LAU to determine its current state (active/failed)
I p1	Initialize the specified LAU; that is, clear all calls currently in progress
S p1	request the specified LAU to store the current State of its call processing variables in its internal buffer storage area
D p1,p2,p3	retrieve and Display either the stored (p2=Ø) or the current (p2=1) state of the call processing variables relating to the specified LAU with options as specified by p3
A p1,p2	request the specified LAU to trace all link Activity for p2 seconds and to store the traced sequence in its internal buffer storage area
T p1,p2,p3	Trace either the stored (p2=Ø) or current (p2=1) sequence of link activity on the specified LAU for p3 seconds (p2=1 option only)
R p1,p2	display either current (p2=Ø) or total (p2=1) statistical Records (number of calls, Information frames, Data packets, time-outs, etc) gathered by the specified LAU

p1=specified LAU address

Fig. 7.1 - EMU commands and their functions

initialization, command and parameter handling procedures and I/O drivers and routines. A complete listing of the X.25 monitor is presented in Appendix C.

Two modes of operation are implemented (X.25 and DDT-80) and swapping between modes is achieved by a single command (X). X.25 is the basic mode and DDT-80 runs as part of the X.25 Input process.

The X.25 monitor is comprised of five processes which are scheduled in a round-robin manner to allow concurrent activity. Each is now described.

Console Input

This process includes the procedures to input the command identifier and parameters and also a procedure that decodes and starts execution of the command. While in the X.25 mode, a command may be aborted at any time by the Kill (K) command.

A command is first validated and error messages are displayed if either the command identifier is not implemented or if the address mode (individual or global) is not supported by the command. Addresses 00 to 09 are reserved for the LAU's and address 10 for the Internetwork Gateway Interface and any other value results in a default to the global address.

The command execution phase is initiated by the EMU creating a command packet (Monitor Request) according to a standard format as shown in Fig. 7.2. Each packet header has six bytes and the command is identified by the corresponding ASCII code. The handshake field is 00 when the command is first issued by the EMU

and is set to Ø1 if the EMU repeats the command to retrieve additional information gathered by the addressed LAU. A timer is then started.

A single Transmit Buffer is used to hold the packet before it is transferred to the Source Buffer within the Interchannel Unit and hence in the case of global commands the same copy of the packet may then be used.

Packet transmission

This process transfers a packet from the Transmit Buffer to the Source Buffer within the Interchannel Unit and, for commands using the global address mode, initiates further transfers until all the specified LAU's have received the command. This is achieved simply by transferring the unchanged copy of the packet from the Transmit Buffer and, prior to each interunit transfer, the new destination address is written to the PIO control port.

Packet reception

A received packet is first validated. This includes a check for minimum length, packet type (Monitor Confirmation), command field (current command) and that the command is outstanding to the addressed LAU. Further actions depend on the type of the commnad and the address mode.

An area of storage within the EMU memory is used to assemble and/or temporarily store the received packets until display of

the received response(s) is carried out.

In the case of global commands, short responses ("Command Accepted" or "Command Rejected") are expected and are sequentially stored in order of arrival. Only when all expected responses are received (timer stopped) or a time-out occurs is the command terminated and the display process enabled. If any response is missing, a local message is generated reporting possible LAU failure.

On the other hand, when an individual command is outstanding, all the storage area is reserved for assembling the response (a long sequence of activity messages, for example). Moreover, the handshake mechanism is considered in the case of the Display and Trace commands. This operates as follows:

(a) if the handshake field is ØØ, all currently gathered information in the addressed LAU has been transferred and the command is terminated (except during Trace);
(b) if the handshake field is Ø1 the command is repeated by the EMU, thus also carrying the handshake field set to Ø1;
(c) if the handshake field is Ø2 (during Trace only) this simply reports end of tracing by the addressed LAU thus terminating the command.

Timer

This process detects a time-out when a command is outstanding. The command is then terminated and the failure(s) reported, as explained.

It also generates timing signals that determine the regular repetition of Trace commands by the EMU (at intervals of 1 s). Repeated Trace commands also carry the handshake field set to Ø1 and this timer mechanism is disabled when a response is received with the handshake field set to Ø2.

Console output

Display of the responses received for global commands is enabled when all responses have been assembled and then follows an order determined by the LAU addresses. In the case of individual commands, however, display is enabled as soon as the response begins to be assembled (Trace ,for example).

In either case display may be stopped/resumed by issuing a Halt (H) command from the console.

When display of the current LAU response is complete the next LAU is selected (if any). Finally, some parameter reinitialization takes place to prepare for the next command.

7.2 Execution of Monitor commands by a Link Access Unit

The received Monitor Request packet is multiplexed with other packets within a LAU. Initially the packet header is saved as for other packets, but then separate processing takes place. The relevant information determining the appropriate action is accessed through the four data fields following the header.

A response(Monitor Confirmation) is generated for each

received command and is structured to a standard format as shown in Fig. 7.3. If the command is unknown a "Command Rejected" message is generated. Otherwise the actions carried out as a result of the command are as follows:

Poll (P)

A "Command Accepted" message is generated and returned.

Initialization (I)

A "Command Accepted" message is generated and then the following action takes place:

(a) at the link level a logical disconnection is performed, but the transmitter and receiver remain enabled;
(b) at the packet level all current calls are remotely cleared, as described.

State (S)

A "Command Accepted" message is generated followed by the update of LAU state information. This is organized in two parts:

(a) general information concerning the link and packet level states, pointers and counters related to the various queues

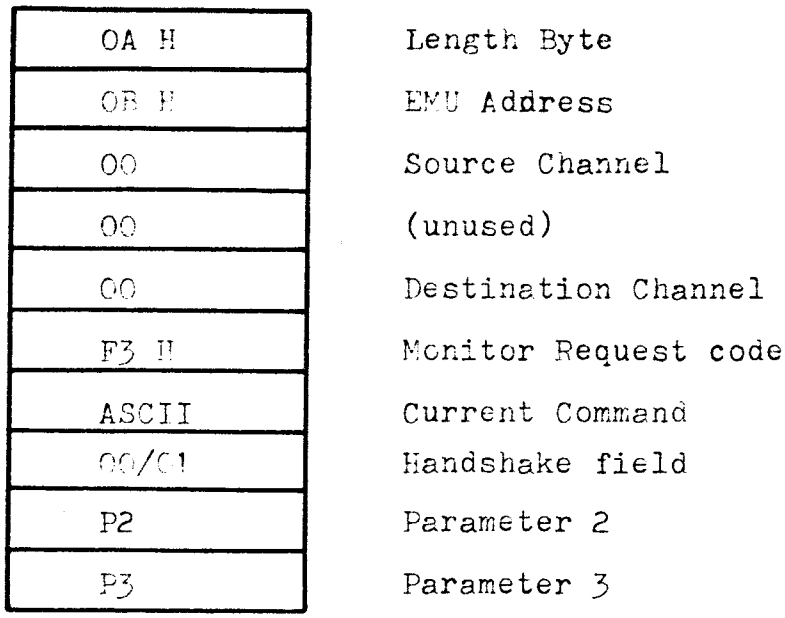


Fig. 7.2 - Monitor Request packet format

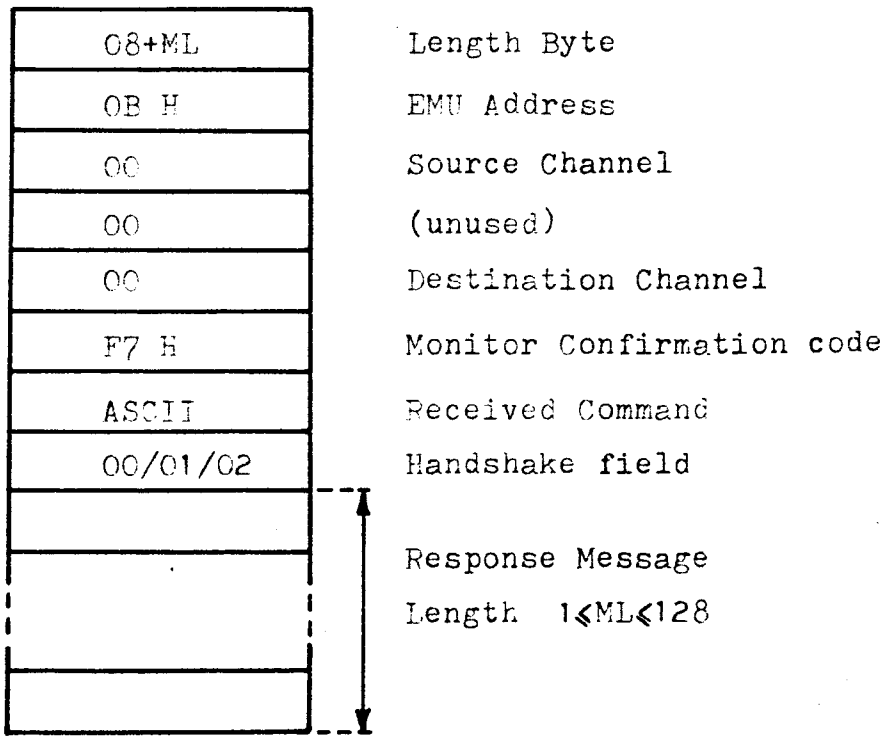


Fig. 7.3 - Monitor Confirmation packet format

and the state of the remote LAU's. A State Buffer stores this information;

(b) information concerning the state and mapping of all logical channels, which is stored in a Map Buffer.

Activity (A)

A "Command Accepted" message is generated and initialization of the Activity function is enabled. The timer is started (for the specified interval), Activity and Trace pointers and counters are reset and the transfer of Activity/Trace messages to the Activity Buffer is enabled.

At the end of the specified interval or if a buffer overflow occurs (currently 384 messages may be stored), the transfer of messages is disabled, which ensures that the command is always terminated by the LAU within a well-defined period.

Display (D)

If the "stored state" option is specified, the information stored as a result of the last State command is retrieved and returned, using the described handshake mechanism.

If the "current state" option is specified, the above is preceeded by the actions described for the State command (thus updating the "stored state").

Trace (T)

If the "stored activity" option is specified, retrieval of the activity stored as a result of the last Activity command begins. This may be simultaneous (delayed) with the execution of the Activity command.

If the "current activity" option is specified, the Activity/Trace function is first reset as for the Activity command (thus updating the "stored activity") and information retrieval begins.

Record (R)

For each recorded event current and total values are kept, which are returned as specified by the Record command option. Up to 64,000 possible outcomes of an event may be recorded without overflow.

The records are kept for both directions of flow and were chosen so that overall performance figures could be derived. For example, total traffic and how it is balanced, overheads due to control frames or packets (number of Information frames as a percentage of the total number of frames, number of Data packets as a percentage of the total number of packets), efficiency of the acknowledgment mechanism (number of RR frames as a percentage of the total number of Information frames), exception conditions (RNR, REJ, CMDR), time-outs (level 2), call clearances due to protocol errors (level 3), etc.

7.2.1 Example Dialogue

An example of a typical dialogue between the Exchange Management Unit and the Link Access Units is now described and illustrated in Fig. 7.4. Some comments have been added to clarify the meaning of the various messages.

A Call set-up between two DTE's (Call Request issued by DTE1 and addressed to DTE2), following a power-on of the exchange has been simulated. This therefore requires the set-up of both logical links; moreover, for synchronization purposes, this is preceeded by a logical disconnection of the links, as explained.

Poll commands are first used to test the active presence of the LAU's in the bus.

In order to synchronize the tracing functions, Activity commands are then issued for both LAU's, specifying the required trace interval (20 s in this example).

Once these commands are acknowledged, Trace commands are issued, to retrieve the information gathered or being gathered by the LAU's (the "stored activity" option is specified).

The use of the Display command to retrieve the current channel state and mapping information is then illustrated. Following the Call set-up, channel 7 in LAU1 (selected by DTE1 in the Call Request) and channel 1 in LAU2 (selected by LAU2 in the Incoming Call) are internally mapped and both are in state p4 (data transfer state). The remaining channels are in state p1 (ready state).

Finally the Record command retrieves the statistical information gathered during the simulation period.

A photograph of the prototype exchange and the monitor

```
>P 1              ;poll LAU 1

*LAU Ø1
CMND AC          ;EMU command accepted

>A 1,2Ø          ;initiate trace activity on LAU 1 for 2Ø seconds

*LAU Ø1
CMND AC          ;EMU command accepted

>T 1,Ø           ;trace stored activity on LAU 1

*LAU Ø1
RDIS             ;DISC frame received (DTE synchronization)
SUA              ;generate UA frame
SDIS             ;generate DISC frame (LAU synchronization)
TRES CF 63       ;UA frame (CF=63) transmitted
TCOM CF 43       ;DISC frame (CF=43) transmitted
RUA              ;UA frame received (DISC acknowledged)
RSAR             ;SARM frame received (DTE Primary set-up)
SUA              ;generate UA frame (Secondary connected)
SSAR             ;generate SARM frame (LAU Primary set-up))
TRES CF 63       ;UA frame (CF=63) transmitted
TCOM CF ØF       ;SARM frame (CF=ØF) transmitted
RUA              ;UA frame received (Primary connected)
RINF CF ØØ       ;Information frame (CF=ØØ) received
SRR  CF 21       ;generate Receiver Ready frame (CF=21)
CREQ CH Ø7       ;Call Request (CH=Ø7) received from local DTE
TRES CF 21       ;Receiver Ready frame (CF=21) transmitted
CCON CH Ø7       ;Call Connected (CH=Ø7) received from remote DTE
SINF CF 2Ø       ;send Information frame (CF=2Ø)
TINF CF 2Ø       ;Information frame (CF=2Ø) transmitted
RRR  CF 21       ;Receiver Ready frame (CF=21) received

>D 1,Ø,1         ;display the current state of calls on LAU 1

*LAU Ø1

CNB ØØ Ø1 Ø2 Ø3 Ø4 Ø5 Ø6 Ø7   ;eight logical channels implemented
CST ØØ ØØ ØØ ØØ ØØ ØØ ØØ Ø4   ;channel state (ØØ=ready; Ø4=data)
LAU Ø1 Ø1 Ø1 Ø1 Ø1 Ø1 Ø1 Ø2   ;remote LAU address
RCH ØØ ØØ ØØ ØØ ØØ ØØ ØØ Ø1   ;remote LAU logical channel

>R 1,1                        ;display the total records on LAU 1

*LAU Ø1
     REC   TRM                ;received/transmitted (from/to DTE)
FRM ØØ Ø6 ØØ Ø6               ;frames
INF ØØ Ø1 ØØ Ø1               ;Information frames
COM ØØ Ø2 ØØ Ø2               ;SARM/DISC frames
RR  ØØ Ø1 ØØ Ø1               ;Receiver Ready frames
RNR ØØ ØØ ØØ ØØ               ;Receiver Not Ready frames
REJ ØØ ØØ ØØ ØØ               ;Reject frames
CMR ØØ ØØ ØØ ØØ               ;CMDR frames
PAK ØØ Ø1 ØØ Ø1               ;packets
CAL ØØ Ø1 ØØ ØØ               ;Call Request/Incoming Call
DAT ØØ ØØ ØØ ØØ               ;Data packets
RST ØØ ØØ ØØ ØØ               ;Restart packets (X.25 interface)

IVR ØØ ØØ                     ;invalid response frames
TM2 ØØ ØØ                     ;link level time-outs
LFL ØØ ØØ                     ;link level failures

CER ØØ ØØ                     ;calls cleared by LAU (error)
TM3 ØØ ØØ                     ;channel time-outs
PFL ØØ ØØ                     ;packet level failures
```

Note: only the dialogue with LAU 1 is represented

Fig. 7.4 - Example dialogue between the Exchange Management Unit and a Link Access Unit

terminal is shown in Appendix D.

7.3 Performance evaluation

As has been stated, the implemented Packet Switching Exchange is not intended as an alternative to existing local networks of the bus or ring type, but rather its main purpose is that of providing a higher level of networking in a local environment, including possible interconnection to public networks. The objective of the present analysis, therefore, is not to establish direct performance comparisons with ring and bus networks which are discussed for example in [74,75]. As will be apparent, however, under certain conditions, the overall characteristics and performance of the exchange may compare favourably with other solutions.

7.3.1 Central exchange characteristics

The exchange, being topologically a star network, clearly shares with other star networks their potential advantages and disadvantages. However, due to its modular, distributed organization, the implemented exchange overcomes many of the conventional disadvantages of star networks. For example, because of its modularity, the reliability of the exchange is increased in the sense that a failure in a LAU only affects the attached controlled link(s). Moreover, with the EMU, isolation of the failed unit and its replacement is facilitated. Also, the

exchange offers a total maximum bandwidth equal to the sum of all link bandwidths which is only limited by the parallel bus capacity (1.8 Mbyte/s) and the responsiveness of each LAU.

As a star network it offers the advantage of easy control of the access to the transmission media (point-to-point), thus allowing the use of available conventional interfaces and protocols. Moreover, flow control techniques are well understood (as in store-and-forward networks) and control of congestion may be enforced by the subnetwork, possibly at different protocol levels.

The use of a central exchange has the possible disadvantages of higher cable costs although this is compensated both by the possible use of existing cable pairs which may be available for speech and also by the use of inexpensive interfaces to the media. The amount of cable required is clearly a function of the geographic scope and the traffic distribution. The use of emerging broadband techniques, however, offering dedicated channels for point-to-point communications is compatible with the use of an exchange topology and may therefore constitute in the future an adequate and economically justifiable solution.

7.3.2 Analytical models

HDLC controlled data links have been studied in the literature and different models have been proposed and analysed using Queue Theory. Generally, the basic assumption is that messages to be transmitted over the link arrive at each station according to a Poisson distribution. However, as we shall see,

balanced traffic as this not only corresponds to the most demanding processing tasks but also to the best use of the full-duplex links. We shall refer all the derived values to a packet size of No bytes.

7.3.3.1 Throughput

35 μs are necessary to service a normal transmitter or receiver interrupt. Overheads are incurred, however, at the beginning and end of transmission as well as at the beginning and end of reception. Considering the same packet length for each direction, the total processing time to transmit and receive a packet as a function of No and partially absorbing the overheads into the No term is 0.07*No + 0.10 ms.

The processing at level 2 depends on the traffic distribution but here we assume:

(a) balanced traffic;
(b) acknowledgment of Information frames is by means of piggybacking;
(c) each Information frame acknowledges a single Information frame in the reverse direction.

Under these conditions an average 0.75 ms are required for the processing of one Information frame in each direction.

As far as level 3 is concerned we consider the processing associated with the transfer of Data packets and, assuming one Data packet processed for each direction, a total of 0.5 ms is

incurred; however, additional overheads should be considered during the call set-up phase.

Finally, the transfers to/from the bus introduce an additional processing time of 0.01*No + 0.25 ms, since 5 μs are necessary to transfer a byte between the LAU and the Interchannel Unit.

Summing up all the terms a total time Tp=1.6 + 0.08*No ms is required for the processing of two packets, one for each direction of flow.

In this analysis no overheads were considered in what concerns recovery mechanisms and permanent functions associated with the EMU (update of statistical records, for example), the latter introducing overheads of 3 to 6 %.

The average Tp is a function of the packet length and therefore not only the average value of No is important, but also its distribution. Two types of packets should be considered - Data and control (mainly supervisory). The first will be of maximum length 128 bytes (plus header), and maximum processor efficiency (minimum processor time/number of bytes transferred) is achieved if the average length of Data packets is kept near the maximum supported. Control packets have lengths of 6 or 7 bytes within the exchange.

As the exchange does not generate supervisory packets and considering only the data transfer phase, then it is the end-to-end use of supervisory packets that influences the packet length distribution. For example, in highly unbalanced traffic (per virtual circuit), RR packets are necessarily used and usually one should be generated for each Data packet. We may then assume that packets will be of two typical lengths and two extreme average

lengths are of interest - above 100 bytes (efficient piggyback) and around 50 bytes (no piggyback).

A general stability condition is now derived, thus placing some bounds on the link baud rate.

For an average packet length of $\overline{No}$ bytes and a link baud rate of f Kbaud, the average time to transfer a packet across the link is Ti=8*No/f ms. This means that a maximum interarrival rate $\lambda max=1/Ti$ may be achieved and generally $\lambda=\lambda max*Y$ with Y<=1 being the link efficiency (useful load).

Thus, assuming a possible maximum link efficiency (Y=1) we may state the stability condition as

$$Tp<Ti$$

or

$$1.6 + 0.08*\overline{No} < 8*\overline{No}/f$$

This leads to a maximum link baud rate, corresponding to the maximum throughput of the exchange, as a function of $\overline{No}$ and an upper bound $fmax=8*\overline{No}/0.08*\overline{No}=100$ Kbaud may be readily determined. For example, for $\overline{No}=50$ bytes, f must be further restricted to about 70 Kbaud, unless the link is permanently used below its maximum useful capacity.

7.3.3.2 Delay

To assess the delay profile of the exchange a possible model for the processing inside a LAU will be derived. The scheduling algorithm adopted implies that during a scheduling cycle the equivalent to a packet is processed in each direction. As far as the Interchannel Unit transport mechanism is concerned, two facts are relevant:

(a) the bus capacity is very high, which means that packets transferred to the Bus Output Queue are in general immediately absorbed by the Source Buffer, without additional queueing;

(b) a packet is transferred from the Destination Buffer on each scheduling cycle and processed at the packet level, which means that no queueing takes place at the Bus Input Queue.

Thus, queueing within each LAU will mainly take place firstly at the Link Input Queue due to general packet processing (Tp) and secondly at the Link Output Queue due to packet transmission (Ti). We shall consider a simplified model in which we assume that all traffic entering a LAU leaves the exchange through the same LAU, thus reproducing a balanced traffic pattern.

Two servers are considered as shown in Fig. 7.5. The first with a mean service time Tp (including all the processing tasks) and the second with a mean service time Ti, which is passive in terms of processor time and only reflects the time required to

times for the tandem.

In our model i=2 and the equivalent service time is Te=max (Tp,Ti). We have, however, established a condition that should hold for any Y and that was Tp<Ti which leads to Te=Ti. Hence, the general stability condition is $\lambda*Ti<1$ (which naturally ensures $\lambda*Tp<1$).

Now $\rho 1=\lambda*Tp=Y*\rho 1max$, with $\rho 1max=Tp/Ti$ and similarly $\rho=\lambda*Ti=Y$, which is simply the utilization factor of the link.

For each link (f=fo) and a known average packet length (No) $\rho 1max$ is constant. This, together with the link utilization factor Y, are the useful parameters that we use to characterize the delay profile of the exchange.

Then

$$W=\frac{Y*Ti}{2*(1-Y)}=\frac{Y*Tp}{2*(1-Y)*\rho 1max}$$

$$W1=\frac{Y*\rho 1max*Tp}{2*(1-Y*\rho 1max)}=\frac{Y*(\rho 1max**2)*Ti}{2*(1-Y*\rho 1max)}$$

$$W2=W-W1$$

In Fig. 7.6 normalized values of W and W1 are represented as a function of Y with $\rho 1max$ as a parameter.

It can be concluded that for $\rho 1max\rightarrow 1$ (Tp→Ti) all queing takes place at the first server (W2→0). Conversely, for $\rho 1max\rightarrow 0$ (Tp<<Ti) then W2→W (the interarrival distribution at the second queue tends to be exponential) and W1→0.

We may now establish the total delay of a packet inside the

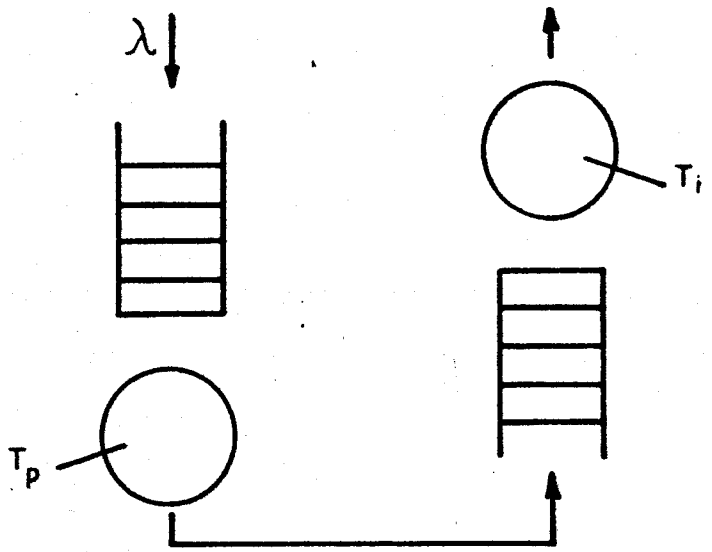


Fig. 7.5 - Queueing model of a Link Access Unit

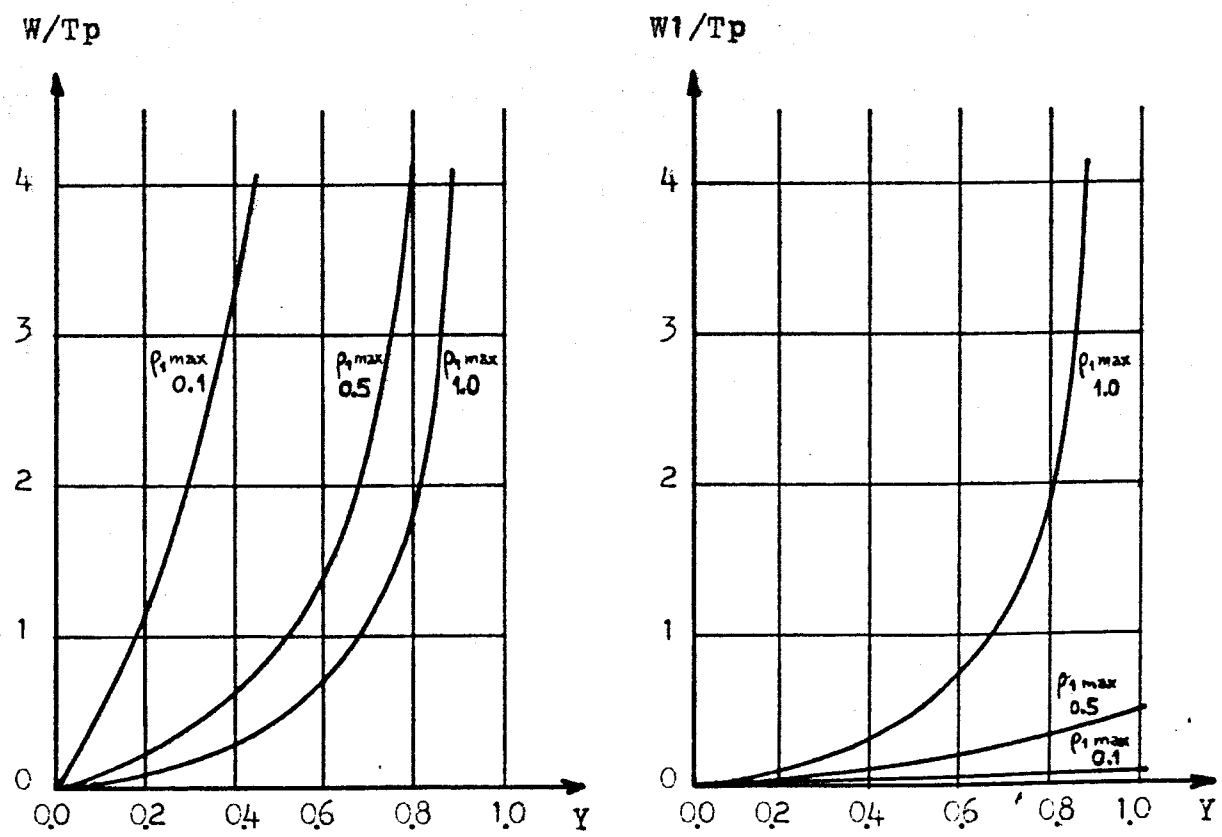


Fig. 7.6 - Exchange delay profile

exchange corresponding to the total time of a packet inside a LAU in our model.

Clearly, T=T1+T2=Tp+Ti+W, but we shall measure the total delay introduced by the exchange between the last byte received in an incoming link until the first byte transmitted in an outgoing link and thus Td=Tp+W while the total end-to-end delay (host-to-host) is Tt=2*Ti+Td.

However, a corrected value of the service time must be considered when determining Td and Tt. In fact, the receiver and transmitter interrupts will be served with probability (1-ρ1) while the processor is idle and hence that fraction will not contribute to the total packet delay.

Then

$$Tp=1.6 + 0.01*No + \rho 1*0.07*No$$

and, for example, for No=50 bytes, the minimum delay of the exchange (ρ1→0) is 2.1 ms.

The average number of packets in the LAU is, using Little's formula,

$$Np=\lambda*T=Y*(1+\rho 1max+Y/(2*(1-Y))$$

Np does not account for the Information frames already transmitted and waiting acknowledgment in a retransmission list. We may now determine the buffer holding time corresponding to those Information frames. We shall consider, as an approximation, that this time is similar for both directions of the link. Hence, we shall determine it in the case of the acknowledgments being

generated by the exchange and thus orderly queued for transmission and therefore, ignoring the last processing time of the received acknowledgment, the holding time is

$$Th=W+Tp+Ti=T$$

This simply means that the average number of Information frames in the retransmission list (Nh) is approximately the same as the remaining number of frames or packets inside the LAU.

7.3.3.3 Conclusions

The previous analysis was mainly intended to give some insight into the overall performance of the exchange and in particular to determine stability conditions and a useful throughput measure for the exchange. Also some figures for the delay and queue lengths in a LAU may be obtained as a function of the link utilization factor and the ratio Tp/Ti.

We may consider an example, corresponding to a link operating at fo=50 Kbaud, which seems to be a typical situation of interest according to the early assumptions. For packets of constant length No=50 bytes, then Ti=8 ms and Tp=5.6 ms (ρ1max=0.7). For a link efficiency of Y=0.8 we conclude that

$$Np=3,\ Nh=3$$

$$W=16\ ms,\ Td=21\ ms,\ Tt=37\ ms$$

which means that in this case the exchange introduces about 55 % of the overall end-to-end delay (note that a window of 7 ensures that no window overflow will occur).

Similarly, the maximum useful throughput (Y=1) of the LAU is 125 packets/s (per direction), thus allowing a total maximum throughput of 1250 packets/s (length 50 bytes) for an exchange comprised of 10 LAU's (each packet is processed in a source and destination LAU), which is well below the capacity of the parallel bus.

In general, these values should be considered as approximate as the processing time is particularly sensitive to the link utilization factor and/or the relation of traffic intensities in both directions. This may result in level 2 processing overheads whenever piggyback of acknowledgment is not possible; however, this effect is negligible for high loads where the most critical performance of the exchange is of interest.

We must also consider the accuracy of the model in more realistic conditions, where traffic within the exchange enters a LAU from various different source LAU's. Due to the implemented bus transport mechanism and the scheduling algorithm, the model is still a good representation of the reality, provided that the internal and external packet interarrival rates are similar.

The most demanding case is when a LAU is flooded with packets from other LAU's; this may even induce some temporary accumulation of packets in other LAU's and if the condition persists ($\rho>1$) congestion of the exchange begins to develop.

Finally, the practical effect of working near the maximum utilization factor, where the delays would increase without limit is, in the first place, a natural traffic reduction from

the DTE's (window overflow due to delayed acknowledgments) which may be completed by explicit level 2 flow control from the exchange. In either case, however, the model will not apply.

The improvement of the exchange throughput may be assessed by reducing the processing time Tp. Use of DMA will obviously reduce the overheads of the frame transmission and reception, and thus Tp becomes practically independent of the packet size. Further reductions may be achieved using HDLC controllers that handle transparently level 2 of X.25. On the other hand this introduces a high dependency (through Ti) of the performance measures on the packet size.

CHAPTER 8

DISCUSSION AND CONCLUSIONS

After first discussing the role of X.25 in a local area network environment, the major portion of the thesis has been concerned with the implementation of a modular Packet Switching Exchange based on the X.25 protocol. This investigation has given rise to some additional aspects, however, and hence some of these are now briefly discussed.

Moreover, the work described is part of a combined research programme, and therefore a broader view of the proposed architecture is also considered. This identifies some of the problems and additional facilities that should be provided in order to give users of the network a more complete range of services and facilities.

8.1 Broadband techniques

One of the potential criticisms concerning the use of a centralized Packet Switching Exchange is the inherent cable costs to link each system to the exchange. Providing the required baud rates and distances are modest, however, it is perfectly feasible to utilize existing cable pairs that are currently present for telephony functions.

As an alternative, however, the current developments in

broadband techniques mean that it is likely that in the future coaxial cables will be utilized to provide a complete range of network interfaces to support a wide variety of user requirements including voice, data and perhaps video. Clearly, therefore, it will be possible to utilize a proportion of the cable bandwidth to support individual point-to-point connections between a central packet switch and a number of distributed computing resources and devices.

It is therefore proposed to further investigate the technical and economical problems of using broadband techniques within the proposed network architecture.

8.2 Multilink Link Access Units

As was discussed in Chapter 7, (assuming the links are interrupt driven), the current design of a LAU is capable of supporting a single serial link with baud rates up to in the order of 50 Kbaud. Alternatively, a LAU may readily control several serial links, with a collective baud rate in the same range; for example, two at 9.6 Kbaud, four at 4.8 Kbaud and eight at 1.2 Kbaud.

Assuming the latter, therefore, it is clearly possible to meet the requirements of a small exchange using just a single self-contained LAU. In this case, it is envisaged that a single LAU could be readily implemented in a typical microcomputer system comprised of a CPU board, supporting vectored interrupts at the peripheral level, up to 64 K of memory and one or two communications boards, each controlling up to six serial links.

An additional VDU interface could then be incorporated for diagnostic and monitoring functions, thus constituting an inexpensive packet switch and hence an inexpensive way of interconnecting a range of X.25 based computers and supporting devices currently available from a number of computer manufacturers.

Some of the implementation aspects inherent in such a unit are now considered.

Each serial link must, of course, be separately controlled, thus requiring independent status information for each controlled link. Similarly each packet level logical channel must be separately controlled and hence, since the same set of logical channel numbers is used across each serial link X.25 interface, logical channels must be further internally distinguished within a LAU by the use of a link identifier.

One first implementation choice could be to have a complete separation of traffic in all links, by providing separate queues and possibly independent buffer pools. However, this separation is in many aspects not kept at the packet level, where for example, traffic from different sources is multiplexed to common destinations - either another LAU or another link within the same LAU.

Hence, input traffic may be multiplexed in a single input queue. This would require, for example, the inclusion of an additional field in the frame header, identifying the incoming link which is inserted by the corresponding link receiver process.

For the purpose of level 3 channel assignment, the link identifier could be internally considered as an extension of the

LAU address or of the logical channel number using the currently unassigned "logical channel group number" field in the packet header.

In what concerns output traffic, the Link Output Queues should be independent in order to allow easy and efficient control of frame transmission over separate links. It is also advantageous to keep separate retransmission lists, to reduce processing overheads (buffer release, retransmission, etc). Therefore packets should also be demultiplexed by the packet level, in separate Packet Output Queues.

Similarly, for general buffer requirements, based on the experience gained from using a single link LAU, the most demanding would be by the link level and link handler processes. For example, the existence of several retransmission lists and possibly the use of dissimilar link baud rates, would require the provision of additional buffers to temporarily hold unacknowledged Information frames. Moreover, although the total input throughput may be comparable, and hence the total number of packets/s, the possible simultaneous reception and transmission over separate links immobilizes buffer resources for longer periods.

In addition, it is also necessary to discuss whether to flow control each link separately, thus requiring some form of buffer reservation per link or to use a global flow control policy irrespective of the current share of buffer resources among the different links.

All these aspects should be considered in the assessment of the scheduling algorithm and the buffer management scheme.

8.3 Additional network components

As has been outlined in Chapter 2, the implemented Packet Switching Exchange is the central component of the proposed architecture. However, an essential feature of the architecture is the support of a wide range of user requirements that must be provided by a set of additional components.

Firstly, although the current trend is probably to the future support of Virtual Terminal protocols [83,84], there is still a likely requirement for unintelligent character mode devices to be interconnected to the network. This may be achieved by means of one or more Packet Assembly/Disassembly devices (PAD's) and currently a set of protocols (X.3, X.28, X.29) exist to integrate a PAD into an X.25 based network.

A PAD appears to the exchange as a multiple call packet mode device. It can therefore be situated either locally with a group of terminals and hence remote from the exchange or within the exchange itself. The latter would clearly require separate communications lines between the exchange and each terminal device. Moreover, a different LAU software interface would be required to handle necessary protocol conversions thus affecting the exchange modularity. Hence, whenever possible, a remotely situated PAD would seem advantageous, thus providing an X.25 interface to the exchange, while all terminal handling functions were kept transparent to the exchange.

A second important requirement is for the support of network-wide communications between a number of separate local networks, based for example on Ethernet, Cambridge ring, etc. This is, of course, a network interconnection issue.

Networks are interconnected by means of gateways, this broadly meaning the collection of hardware and software required to interconnect two or more data networks and thus being either implemented in a single separate device, in separate devices attached to each network (half-gateways) or simply as software in conventioanl packet switches.

The problems raised by network interconnection have been widely discussed [85,86,87,88] in what concerns such issues as addressing and routing, protocol level of interconnection, flow and congestion control, accounting, internet services, etc.

It is now accepted that all network dependent functions should be hidden from the user-oriented layers, and hence a useful basis for compatibility should be found at least at the transport level of the protocols. In this sense a gateway should interface to each local network as a host computer in that network. A step in that direction is achieved by the support by local network manufacturers of X.25 interfaces to public data networks. An interesting example is given by the evolution of a Transport Service Byte Stream protocol [89], compatible with the British Post Office Transport Service, tightly coupled with X.25, that has been adopted for use with the Cambridge ring.

This suggests that the exchange should provide an X.25 interface to the different local networks and, whenever required, any protocol conversions should be performed by a gateway unit within each local network.

A third requirement is for the interconnection to public networks. Although the X.75 standard protocol has now been adopted for the interconnection of X.25 public data networks, it is likely that a private network will be connected to a public

network by means of the X.25 protocol, thus appearing to the public network as a multiple call DTE.

It is proposed, therefore, that this should be provided by the implementation of a separate gateway providing such functions as flow control, addressing and routing, call charging, etc. The latter is then interfaced to the exchange through an Internetwork Gateway Interface (IGI), similar to the LAU's early described. Any traffic to be carried outside the local network served by the exchange is then recognized by each LAU and internally routed to the Internetwork Gateway Interface.

8.3.1 Network address scheme

Assuming connections are required from the local network to a public network, this may be accomplished by the use of a single address in which additional addressing information handled by the gateway is carried in the user field of Call Request packets or alternatively embedded in a message header at the transport layer. Alternatively, the local network could be assigned a set of addresses, possibly having a common prefix.

Then, for internal purposes the local network addresses may be hierarchically structured [18] in the following fields:

X (several digits) - identifies the whole network served by the exchange (could be the single assigned address, as in the first alternative above);

Y (one digit) - identifies the LAU (ØØ to Ø9); internally the IGI is assigned address 1Ø and the EMU is assigned address

11;

W (one digit) - identifies the serial link controlled by each LAU (up to 10);

Z (several digits) - reserved for all devices served by the same serial link (single device, terminals connected through a PAD, devices connected through an internetwork gateway).

This structure exploits the use of the exchange modularity and the address numbering scheme inherent to X.25. Although each separate local network (ring, bus, etc) may have its own internal addressing scheme (broadcast, addressing by process name [28], etc), network-wide addressing, as supported by X.25, should be by means of a physical address identifier.

As an example of the described addressing scheme, the routing function (during call set-up) could be carried out as follows:

(a) if the called and the local X fields are different, the call is routed through the IGI;

(b) if the X fields are equal, then the call is internally routed to the LAU specified in the called address (Y) and then to the specified link (W);

(c) the Z field is handled by external components (PAD's, Internetwork Gateways, etc).

8.4 Conclusions

Although at the time the work discussed in this thesis started there was only the standard X.25 protocol, standardization within Local Area Networks is now earnestly being pursued and indeed a first draft - IEEE 802 - has recently become available [25]. Considerable argument is now arising in particular in what concerns access methods since these may have a strong impact on emerging "de facto" standards such as Ethernet, for example.

Another relevant aspect of the IEEE 802 reference model is the establishment of a correspondence to the OSI reference model at the physical and data link layers and also the offering of two types of link services: connectionless and connection oriented. The latter is procedurally similar to HDLC; however, and due to the broadcast nature of rings and buses and the transparent address recognition at the media access control layer, it has in fact an end-to-end nature.

Hence, within a system, multiple links must be set-up to provide connections with other interconnected systems, which is, in some aspects, a network layer function. This does not rule out the possibility, however, of multiplexing calls at the network layer over established link connections. Moreover, the candidacy of the packet level of X.25 to provide network layer functions for local networks is still an open question.

Two conclusions seem important: first, link layer procedures of the HDLC type are now proposed within local area network architectures and secondly, current standardization does not preclude compatibility with the OSI reference model at the

network layer and above.

This clearly strengthens the relations between X.25 and local area networks which will not only be concerned with interconnection to public networks. It seems likely, therefore, that much hardware and software support currently offered for X.25 networks will find wider application within the local network environment.

On the other hand, market forces and research interests are likely to result in the wider acceptance and support of different network types that may even coexist within the same organization. However, the lack of any agreed software standards for use with these networks severely compromises their usefulness and growth potential.

The network topology described in this thesis is intended to overcome these problems, by providing a higher level network using the internationally agreed X.25 protocol. Moreover, the network growth is based on a range of components that may be implemented in accordance with current standards, while wider system interconnection, as proposed by the OSI reference model is inherent to the proposed architecture.

The design and implementation of the X.25 Packet Switching Exchange has demonstrated that an inexpensive modular design based on microprocessor technology can be used for this function. The performance figures show that a single Link Access Unit can handle a number of full-duplex links which are probably sufficient to meet the requirements of a small network. The use of a number of Link Access Units interconnected by an Interchannel Unit of the type described increases proportionally the exchange throughput, which may be further improved by the use

of more sophisticated and currently available X.25 link level controllers.

The implementation of the Exchange Management Unit provides good diagnostic facilities, either to monitor the exchange and assess performance figures or to readily identify and replace faulty components.

Finally, the usefulness of the proposed architecture as well as other network solutions cannot be viewed independently from the continuing development in local area network technologies (broadband, optical fibres, etc) and the current standardization efforts that must be directed to achieve wider network compatibilty and system interconnection.

REFERENCES

[1] Roberts, L.G., and Wessler, B.D., "Computer network developments to achieve resource sharing", AFIPS SJCC, Vol. 36, pp. 543-549, 1970.

[2] Kleinrock, L. , "Queueing Systems - Volume II : Computer Applications", Wiley 1976.

[3] Kermani, P., and Kleinrock, L. , "A trade-off study of switching systems", Proc. ICC, Boston, MA, pp. 20.4.1-20.4.8, June 1979.

[4] Roberts, L.G., "Data by the packet", IEEE Spectrum, pp. 46-51, Feb. 1974.

[5] Heart, F.E., Kahn, R.E., Ornstein, S.M., Crowther, W.R., and Walden, D.C., "The Interface Message Processor for the ARPA computer network", AFIPS SJCC, Vol. 36, pp. 551-567, 1970.

[6] Pouzin, L., "Presentation and major design aspects of the CYCLADES computer network", Proc. of the NATO Advanced Study Institute on Computer Communication Networks", Sussex, U.K., pp. 415-434, Sept. 1973.

[7] Barber, D.L.A., "A European Informatics Network: achievments and prospects", Proc. 3rd ICCC, Toronto, Canada, pp. 44-50, Aug. 1976.

[8] Danet, A., Despres, R., Le Rest, A., Pichon, G., and Ritzenthaler, S., "The French public packet switching service: the Transpac network", Proc. 3rd ICCC, Toronto, Canada, pp. 251-259, Aug. 1976.

[9] Twyver, D.A., and Rybczynski, A., "Datapac subscriber interfaces", Proc. 3rd ICCC, Toronto, Canada, pp. 143-149, Aug. 1976.

[10] Bright, R.D., "Experimental Packet Switching project of the UK Post Office", Proc. of the NATO Advanced Study Institute on Computer Communication Networks", Sussex, U.K., pp. 435-444, Sept. 1973.

[11] Tymes, L., "TYMENET a terminal oriented communications network", AFIPS SJCC, Vol. 38, pp. 211-216, June 1971.

[12] Gray, J.P., "Network services in Systems Network Architecture", IEEE Trans. on Communications, Vol. COM-25, No. 1, pp. 104-116, Jan. 1977

[13] Wecker, S., "DNA: the Digital Network Architecture", IEEE Trans. on Communications, Vol. COM-28, No. 4, pp. 510-526, April 1980.

[14] Kleinrock, L., "Principles and lessons in packet communications", Proc. IEEE, Vol. 66, No. 11, pp. 1320-1329, Nov. 1978.

[15] Green, P.E., "An introduction to network architectures and protocols", IEEE Trans. on Communications, Vol. COM-28, pp. 413-423, April 1980.

[16] Pouzin, L., "A tutorial on protocols", Proc. IEEE, Vol. 66, No. 11, pp. 1346-1370, Nov.1978.

[17] CCITT, "Draft Recommendation X.25", CCITT AP VI-No.55-E, Geneva, Switzerland, pp. 80-144, 1976.

[18] Pouzin, L., "Virtual circuits vs. datagrams - technical and political issues", AFIPS NCC, Vol. 45, pp. 483-494, June 1976.

[19] Zimmerman, H., "OSI reference model - the ISO model of architecture for Open Systems Interconnection", IEEE Trans. on Communications, Vol. COM-28, No. 4, pp. 425-432, April 1980.

[20] Folts, H., "X.25 transaction oriented features - datagram and fast select", IEEE Trans. on Communications, Vol. COM-28, No. 4, pp. 496-500, April 1980.

[21] Cerf, V., McKenzie, A., Scantelbury, R., and Zimmerman, H., "Proposal for an international end-to-end protocol", Computer Communications Review, Vol. 6, No. 1, pp. 63-89, Jan. 1976.

[22] CCITT, "Provisional Recommendations X.3, X.25, X.28, X.29 on packet-switched data transmission services", Geneva, Switzerland, pp. 1-8/ 50-84, 1978.

[23] Clark, D.C., Pogran, K.T., and Reed, D.P., "An introduction to local area networks", Proc. IEEE, Vol. 66, No. 11, pp. 1497-1517, Nov. 1978.

[24] McQillan, J.M., "Local network technology and the lessons of history", Proc. Local Area Communications Network Symposium, Boston, MA, pp. 191-197, May 1979.

[25] IEEE 802 - Local Network Standard, Draft B, Oct. 1981.

[26] Penney, B.K., and Baghdadi, A.A., "Survey of computer communication loop networks: part 1", Computer Communications, Vol. 2, No. 4, pp. 165-180, Aug. 1979.

[27] Wilkes, M.V., and Wheeler, D.J., "The Cambridge digital communication ring", Proc. Local Area Communications Network Symposium", Boston, MA, pp. 47-61, May 1979.

[28] Liu, M.T., and Reames, C.C., "Message communication protocol and operating system design for the Distributed Loop Computer Network (DLCN)", 4th Annual Symposium on Computer Architectures, Vol. 5, No. 7, pp. 193-200, March 1977.

[29] Tobagi, F.A., "Multiaccess protocols in packet communication systems", IEEE Trans. on Communications, Vol. COM-28, No. 4, pp. 468-487, April 1980.

[30] Abramson, N., "The Aloha system - another alternative for computer communications", AFIPS FJCC, Vol. 37, pp. 281-285, Nov. 1970.

[31] Metcalfe, R.M., and Boggs, D.R., "Ethernet: distributed packet switching for local computer networks", Comm. ACM, Vol. 19, No. 7, pp. 395-404, July 1976.

[32] "Ethernet: a local area network - data link layer and physical layer specifications", Digital, Intel and Xerox Corporations, Version 1.0, Sept. 1980.

[33] The Yankee Group, "The report on Local Communications", pp. 167-172, 1981.

[34] Heart, F.E., Ornstein, S.M., Crowther, W.R., and Barker, W.B., "A new minicomputer/multiprocessor for the ARPA network", Proc. of the NATO Advanced Study Institute on Computer Communication Networks", Sussex, U.K., pp. 159-180, Sept. 1973.

[35] CCITT, "Provisional Recommendations X.3, X.25, X.28, X.29", Geneva, Switzerland, pp. 50-80, 1978.

[36] CCITT, "Data Communication Networks, services and facilities, terminal equipment and interfaces", Yellow Book, Vol. VIII, fascicle VIII.2, Geneva, Switzerland, pp. 100-190, 1981.

[37] Hess, M.L., Brethes, M., and Saito, A., "A comparison of four X.25 public network interfaces", Proc. ICC, Boston, MA, pp. 38.6.1-38.6.8, June 1979.

[38] Rybczynski, A., and Palfarman, J.D., "A common X.25 interface to public data networks", Computer Networks, Vol. 4, No. 3, pp. 97-110, June 1980.

[39] CCITT, "Recommendation X.21", Orange Book, Vol. VIII.2, Geneva, Switzerland, pp. 38-63, 1976.

[40] International Organization for Standardization (ISO), "HDLC, frame structure" - IS-3309, "HDLC, elements of procedure" - IS-4335.

[41] Selga, J., and Rivera, J., "HDLC reliability and the FRBS method to improve it", Proc. 7th Data Communications Symposium, Mexico, pp. 260-267, Oct. 1981.

[42] Funk, G., "Message error detecting properties of HDLC protocols", IEEE Trans. on Communications, Vol. COM-30, No. 1, pp. 252-257, Jan. 1972.

[43] Davies, D.W., Barber, D.L.A., Price, W.L., and Solomonides, C.M., "Computer networks and their protocols", Wiley 1979.

[44] Belsnes, D., and Lynning, L., "Some problems with the X.25 packet level protocol", INWG protocol note 55, 1976.

[45] Bochman, G.V., "Notes on the X.25 procedures for virtual call establishment and clearing", Computer Communications Review, Vol. 7, No. 4, pp. 53-59, Oct. 1977.

[46] Bochman, G.V., "Finite state description of communication protocols", Computer Networks, Vol. 2, pp. 361-372, 1978.

[47] Rybczynski, A., "X.25 interface and end-to-end virtual circuit service characteristics", IEEE Trans. on Communications, Vol COM-28, No. 4, pp. 500-510, April 1980.

[48] Konnig, W., "Design and simulation of an end-to-end transport protocol on top of X.25", Proc. ICC, Boston, MA, pp. 50.3.1-50.3.7, June 1979.

[49] Eschenauer, E.C., and Obozinski, V.O., "The Network Communication Manager: a Transport Station for the SGB network", Computer Networks, Vol. 2, pp. 236-249, 1978.

[50] Rinde, J., and Caisse, A., "Passive flow control techniques for distributed networks", Proc. Int. Symp on Flow Control in Computer Networks", Versailles, France, pp. 155-160, Feb. 1979.

[51] Zilog, "Z80-CPU/ Z80 A-CPU technical manual", pp. 1-75, 1977.

[52] Zilog, "Z80 SIO technical manual", pp. 1-47, Aug. 1978.

[53] "LSI ready to make a mark on packet-switching networks" (part 1), "LSI circuit simplifies packet-network connection" (part 2), pp. 89-99, Dec. 20, 1979.

[54] Pouzin, L., "Methods, tools and observations on flow control in packet switched data networks", IEEE Trans. on Communications, Vol. COM-29, No. 4, pp. 413-426, April 1981.

[55] Giessler, A. , Hanle, J., Konig, A., and Pade, E., "Free buffer allocation - an investigation by simulation", Computer Networks, Vol. 2, pp. 191-208, 1978.

[56] Gerla, M., and Kleinrock, L., "Flow control: a comparative survey", IEEE Trans. on Communications, Vol. COM-28, No. 4, pp. 553-574, April 1980.

[57] Chou, W., and Gerla, M., "A unified flow control and congestion control model for packet networks", Proc. 3rd ICCC, Toronto, Canada, pp. 475-482, Aug. 1976.

[58] McQillan, J.M., "Interaction between routing and congestion control in computer networks", Proc. Int. Symp. on Flow Control in Computer Networks, Versailles, France, pp. 63-75, Feb. 1979

[59] Lemieux, C., "Theory of Flow Control in shared networks and its application in the Canadian telephone network", IEEE Trans. on Communications, Vol. COM-29, No. 4, pp. 399-413, April 1981.

[60] Rudin, H., and Mueller, H., "Dynamic routing and flow control", IEEE Trans. on Communications, Vol. COM-28, No. 7, pp. 1030-1039, July 1980.

[61] Raubold, E., and Haenle, J., "A method of deadlock-free resource allocation and flow control in packet networks", Proc. 3rd ICCC, Toronto, Canada, pp. 483-487, Aug. 1976.

[62] Lam, S.S., and Reiser, M., "Congestion control of store-and-forward networks by Input Buffer Limits", IEEE Trans. on Communications, Vol. COM-27, No. 1, pp. 127-133, Jan. 1979.

[63] Schwartz, M., and Saad, S., "Analysis of congestion control techniques in computer communication networks", Proc. Int. Symp. on Flow Control in Computer Networks, Versailles, France, pp. 113-130, Feb. 1979.

[64] Kamoun, F., "A drop and throttle flow control (DTFC) policy for computer networks", 9th International Teletraffic Congress, Spain, pp. 8, Oct. 1979.

[65] Sproule, D.E., and Mellor, F., "Routing, flow and congestion control in the Datapac network", IEEE Trans. on Communications, Vol. COM-29, No. 4, pp. 386-391, April 1981.

[66] Magoon, R., and Twyver, D., "Flow and congestion control in SL-10 networks", Proc. Int. Symp. on Flow Control in Computer Networks, Versailles, France, pp. 45-61, Feb. 1979.

[67] Simon, J.M., and Danet, A., "Controle des resources et principes du routage dans le reseau Transpac", Proc. Int. Symp. on Flow Control in Computer Networks, Versailles, France, pp. 33-43, Feb. 1979.

[68] Uninett, "Implementation of the CCITT X.25 link level procedures", Uninett report No. 3, 1977-04-01.

[69] Belsnes, D., "X.25 implemented in Simula", Proc. Eurocomp 78, England, pp. 387-401, 1978.

[70] Patel, A., and Purser, M., "System programming for data communications on minicomputers", Software - Practice and Experience, Vol. 10, pp. 283-305, 1980.

[71] Bochman, G.V., and Joachin, T., "Development and structure of an X.25 implementation", IEEE Trans. on Software Enginneering, Vol. SE-5, pp. 429-439, Sept. 1979.

[72] "A network independent Transport Service", Sudy Group 3 of the Post Office PSS User Forum, Yellow Book, pp. 1-89, Feb. 1980.

[73] Mostek, "DDT-80 Operating System", version 1.3, 1977.

[74] Bux, W., "Local area subnetworks: a performance comparison", IEEE Trans. on Communications, Vol. COM-29, No. 10, pp. 1465-1473, Oct. 1981.

[75] Penney, B.K., and Baghdadi, A.A., "Survey of computer communication loop networks: part 2", Computer Communications, Vol. 2, No. 5, pp. 224-241, Oct. 1979.

[76] Gelenbe, E., Labetoule, J, and Pujolle, G., "Performance evaluation of the protocol HDLC", Proc. of the Symp. on Computer Network protocols", Liege, Belgium, pp. G.3.1-G.3.5, Feb. 1978.

[77] Labetoule, J., and Pujolle, G., "Modelling and performance evaluation of the protocol HDLC", Proc. Int. Symp. on Flow Control in Computer Networks, Versailles, France, pp. 307-320, Feb. 1979.

[78] Traynham, K.C., and Steen, R.F., "SDLC and BSC on satellite links: a performance comparison", Computer Communications Review, Vol. 7, No. 4, pp. 3-14, Oct. 1977.

[79] Bux, W., and Truong, H.L., "A queueing model for HDLC controlled data links", Proc. Int. Symp. on Flow Control in Computer Networks, Versailles, France, pp. 287-306, Feb. 1979.

[80] Bux, W., Kummerle, K., and Truong, H.L., "Balanced HDLC procedures: a performance analysis", IEEE Trans. on Communications, Vol. COM-28, No. 11, pp. 1889-1898, Nov. 1978.

[81] Kleinrock, L., "Queueing Systems - Volume I: Theory", Wiley 1975.

[82] Rubin, I., "Communication Networks: message path delays", IEEE Trans. on Information Theory, Vol. IT-20, No. 6, pp. 738-745, Dec. 1974.

[83] Barber, D.L.A., "Terminals, tasks and trends in Computer Networks", Euro IFIP, London, pp. 395-401, Sept. 1979.

[84] Day, J.D., "Resource sharing protocols", Computer, pp. 47-56, Sept. 1979.

[85] Cerf, V.G., and Kahn, R.E., "A protocol for packet network interconnection", IEEE Trans. on Communications, Vol. COM-22, No. 5, pp. 637-648, May 1974.

[86] Postel, J.B., "Internetwork protocol approaches", IEEE Trans. on Communications, Vol. COM-28, No. 4, pp. 604-611, April 1980.

[87] Cerf, V.G., and Kirstein, P.T, "Issues in packet network interconnection", Proc. IEEE, Vol. 66, No. 11, pp. 1386-1408, Nov. 1978.

[88] Sunshine, C.A., "Interconnection of Computer Networks", Computer Networks and Simulation, North Holland, pp. 39-68, 1978.

[89] Dallas, I.N., "Transport Service Byte Stream Protocol", Revision 3, UKC Computing Laboratory, Report No. 1, pp. 1-39, Aug. 1981.

[90] Gostl, J., "A problem with the X.25 link access procedures", Computer Communications Review, Vol. 7, No. 4, pp. 37-40, Oct. 1977.

APPENDIX A

ADDITIONAL ASPECTS OF THE LINK ACCESS PROCEDURES

As has been discussed in Section 5.4 the link level procedures are intended mainly to support the packet layer functions and therefore set-up and disconnection of the link must be viewed in a global context.

We shall assume here, however, that the use of DISC has the purpose of terminating any information transfer that is or was in progress and therefore we consider that no Information frames should remain outstanding from a previous logical connection. This means that a level 2 disconnection must be followed by related actions at higher layers: remote call clearance, at the packet layer, initiated by the subnetwork, for example.

Problems inherent to the Link Access Procedures, related to the link set-up,, link failure and Primary blocking are now analysed and solutions are proposed.

A.1 Link set-up

An important question concerning the completion of link set-up relates to the beginning of Information frame transfer: should the link be completely set-up in both directions (LAP) prior to any Information frame transfer or is it enough to simply set-up one direction to enable the corresponding Information transfer?

Let us assume in the first instance that completion of link set-up in both directions is required, by means of an exchange of SARM and UA frames between stations A and B. Assuming that UA (A to B) is lost, station B will probably retransmit SARM following a time-out, but in the meantime Information frame(s) may be transmitted from A to B. Naturally, station A is not violating the protocol; moreover, it can only generally consider the reception of further SARM frames from B as link resets.

On the other hand, when receiving the Information frame(s), the Secondary in B is already set-up and the question is whether to accept or not the Information frame(s). Station B could discard the Information frame(s) and then either:

(a) send a DISC, thus leading to a premature loss of Information frames;

(b) send a CMDR, requesting a link reset from A to B;

(c) simply default station A to a time-out recovery.

None of the solutions entirely solves the problem and they do not avoid eventual looping conditions.

Station A could eventually have delayed the transmission of Information frames, allowing for the eventual retransmission(s) of SARM from B; however, station A would never be sure about the state of station B at least until the reception of an Information frame and so this does not solve the problem either.

Similar considerations would apply if a SARM (from B to A) had been lost and in both cases the situation is not qualitatively different from a link reset, since then Information frames are also temporarily suspended in one direction only.

It may be inferred, therefore, that it is advantageous to accept that information transfer may begin independently for each direction, provided that this is set-up, on the assumption that, according to the protocol, the other direction is either connected or at least is being connected. However, it is necessary to check this solution against more persistent failures of the link. These are now discussed.

A.2 Link failures

We shall consider two situations that may induce loss of synchronism between two stations: a temporary power down and a succession of logical failures (time-out) during set-up or disconnection. In either case the station that fails must be completely reinitialized and evolve to a state where synchronization between stations must be possible when a link set-up is later required.

Assuming that station B temporarily fails and later attempts to set-up the link (SARM), station A may interpret this as a reset and it was considered that this could lead to a dealock situation [90]. Several actions can be taken to avoid the deadlock although not all equally effective. For example, station B could use a CMDR to force a link reset from A; this would resynchronize both stations at the link level, but not at higher levels (the station that fails completely reinitializes, while the other only resets the link layer). This solution would therefore require further actions at higher layers (for example, station B could use a Restart packet to clear all calls still

outstanding in station A).

A different solution is proposed, however. After a failure and before a new set-up is locally required or the remote station is sensed to be active (command received), a link disconnection is first performed, thus creating the conditions to achieve context synchronization at all levels. The same procedure should also apply when a station is powered on for the first time; it would not be required, however, if the link had gracefully evolved to the disconnected state (exchange of DISC and UA frames).

This creates the need to distinguish at each station between a "real" DISC (to terminate an information transfer) and a "synchronizing" DISC. We shall assume that a real disconnection is taking place when a DISC is received by a station with its Primary and/or Secondary in the connected state (set-up). The solution here described is, therefore, compatible with the link set-up model proposed in A.1.

A.3 Primary blocking

Normally, the reception of a valid N(R) does not create any particular problems. If it produces a real acknowledgment, the acknowledged Information frames are released, the timer restarted (if Information frames are still outstanding) and the window updated. There are, however, two cases that may lead to possible temporary or permanent blocking of the Primary, or at least some performance degradation. They are now discussed.

Time-out recovery

The protocol clearly states that, once initiated by means of an Information frame retransmission (P=1), time-out recovery only ends when a supervisory frame with F=1 is received (or a reset takes place). Neverthless, it is possible that, before the end of recovery, some acknowlegments are received (with F=Ø). In the latter case, if used to release any outstanding Information frames, contention may arise if the Information frame being retransmitted (P=1) is still in the Link Output Queue.

Another situation may also develop: while in recovery all outstanding Information frames are acknowledged (F=Ø) and released, a new time-out occurs (F=1 not received) and no new Information frames are available for transmission. Hence, the Primary will be in a situation where it is in recovery but recovery cannot complete, unless the link is reset.

Since during time-out recovery the window is reduced to one (hence this must be as transient as possible) an alternative offering some advantages is to consider all the received acknowledgments (F=Ø) for N(R) validation purposes only (without any Information frame release) in order that further retransmissions are possible. This keeps the recovery mechanism in operation and only in case of repeated failures is a reset initiated. Then only at the end of recovery (F=1) are the correctly acknowledged Information frames released.

Although only one Information frame is required for this mechanism to work, by preventing release of Information frames during time-out recovery, the above described contention problem is also avoided.

Remote Busy

When a RNR is received (remote busy) transmission of Information frames must be suspended and will only resume after the condition is cleared (RR/REJ received or reset performed).

It is, however, possible that some previously outstanding Information frames are accepted by the DTE (and even acknowledged by means of piggybacking) before clearing the busy condition or that the DTE clears the busy condition but the corresponding RR/REJ is lost.

Although no time-out directly protects the clearance of the busy condition, blocking of the Primary is avoided by allowing it to (re)transmit the Information pointed to by the received RNR and this is now protected by a time-out.

For this mechanism to work, however, and because no other Information frames are allowed to be transmitted by the DCE, this Information frame must be kept for eventual retransmission, even if acknowledged (by piggybacking), until the condition is cleared. Otherwise eventual loss of RR/REJ will block the Primary (with reset as a possible default).

APPENDIX B

EXCHANGE PROGRAM LISTINGS - X.25 CONTROL SOFTWARE

```
                        0001 ;       **********************
                        0002 ;       * PROCESS SCHEDULER *
                        0003 ;       **********************
                        0004 ;
                        0005 ;
C000  C303C0            0006         JP      BEGIN           ;power on jump
C003  F3                0007 BEGIN   DI                      ;disable Interrupts
C004  3100FF            0008         LD      SP,STACK        ;init Stack pointer
C007  CD1CC0            0009         CALL    INIT
C00A  FB                0010         EI                      ;enable Interrupts
C00B  CD5AC3            0011 RUN     CALL    TIMNG           ;schedule processes
C00E  CDAFC3            0012         CALL    BFMNG
C011  CD67C4            0013         CALL    LINK
C014  CD74CC            0014         CALL    PACK
C017  CD47DC            0015         CALL    BUS
C01A  18EF              0016         JR      RUN             ;repeat
                        0017 ;
                        0018 ;
                        0019 ;       ***********************
                        0020 ;       * LAU INITIALIZATION *
                        0021 ;       ***********************
                        0022 ;
                        0023 ;
C01C  ED5E              0024 INIT    IM      2               ;set interrupt mode
C01E  3EDF              0025         LD      A,IVEC
C020  ED47              0026         LD      I,A
C022  011E00            0027 INPOOL  LD      BC,2*NBUF       ;init pool (buf addr)
C025  215CDF            0028         LD      HL,BUFTAB
C028  1100F8            0029         LD      DE,TPOOL
C02B  EDB0              0030         LDIR
C02D  AF                0031         XOR     A
C02E  2160F9            0032 INQ     LD      HL,QR           ;init queue variab
C031  061B              0033         LD      B,1BH
C033  77                0034 INQ1    LD      (HL),A
C034  23                0035         INC     HL
C035  10FC              0036         DJNZ    INQ1
C037  2163F9            0037         LD      HL,NOFB         ;assign buffers
C03A  360B              0038         LD      (HL),NBUF-NGB0  ;NOFB=NBUF-NGB0
C03C  2164F9            0039         LD      HL,NOGB
C03F  3604              0040         LD      (HL),NGB0       ;NOGB=NGB0
C041  2165F9            0041         LD      HL,DPOOL
C044  360F              0042         LD      (HL),NBUF
C046  210001            0043 INLL    LD      HL,INLST        ;init lev2 state
C049  2230F9            0044         LD      (LSTATE),HL
C04C  2132F9            0045         LD      HL,MAXNS        ;init lev2 variab
C04F  0606              0046         LD      B,06
C051  77                0047 INLL1   LD      (HL),A
C052  23                0048         INC     HL
C053  10FC              0049         DJNZ    INLL1
C055  3240F9            0050         LD      (BUFREG),A      ;reset Busy
C058  3247F9            0051         LD      (IRLP),A
C05B  2141F9            0052 INPL    LD      HL,GENST        ;init lev3 state
C05E  0606              0053         LD      B,06
C060  77                0054 INPL1   LD      (HL),A
C061  23                0055         INC     HL
C062  10FC              0056         DJNZ    INPL1
C064  2100DF            0057 INADD   LD      HL,ADTAB        ;get LAU address
C067  3E01              0058         LD      A,ADDSX         ;from address table
C069  B5                0059         OR      L
C06A  6F                0060         LD      L,A
```

```
C06B  7E        0061         LD     A,(HL)
C06C  324CF9    0062         LD     (LOCADD),A      ;save in LOCADD
C06F  AF        0063 INADD0  XOR    A
C070  060E      0064         LD     B,0FH-ADDSX
C072  0E0F      0065         LD     C,0FH
C074  2110DF    0066         LD     HL,ADTAB+10H    ;get sign digits
C077  2B        0067 INADD1  DEC    HL
C078  0D        0068         DEC    C
C079  BE        0069         CP     (HL)
C07A  2803      0070         JR     Z,INADD2
C07C  0601      0071         LD     B,01
C07E  0C        0072         INC    C
C07F  10F6      0073 INADD2  DJNZ   INADD1
C081  79        0074         LD     A,C
C082  324DF9    0075         LD     (SIGDIG),A      ;save in SIGDIG
C085  2100F9    0076 INCH    LD     HL,CSI          ;init chan tables
C088  3E00      0077         LD     A,P1            ;state=p1
C08A  CD73C1    0078         CALL   INCH0
C08D  2108F9    0079         LD     HL,RDCE
C090  3A4CF9    0080         LD     A,(LOCADD)      ;rem LAU=local
C093  CD73C1    0081         CALL   INCH0
C096  AF        0082         XOR    A               ;rem chan=0
C097  2110F9    0083         LD     HL,RCI
C09A  CD73C1    0084         CALL   INCH0
C09D  21B0F9    0085 INRCD   LD     HL,TRCDP        ;init records
C0A0  CD7AC     0086         CALL   INRCD0
C0A3  2140FA    0087         LD     HL,CRCDP
C0A6  CD7AC1    0088         CALL   INRCD0
C0A9  CDCDC0    0089         CALL   INDCE           ;init LAU tables
C0AC  3A41F9    0090         LD     A,(GENST)
C0AF  4F        0091         LD     C,A
C0B0  CD9CD4    0092         CALL   INRST           ;init LAU synchr
C0B3  79        0093         LD     A,C
C0B4  3241F9    0094         LD     (GENST),A
C0B7  CD20D9    0095         CALL   GETST           ;init EMU functions
C0BA  CD99D9    0096         CALL   CHMAP
C0BD  CD5ADA    0097         CALL   INITRC
C0C0  CDF5C0    0098         CALL   PERRES          ;reset peripherals
C0C3  CD0AC1    0099         CALL   PINIT           ;init PIO
C0C6  CD20C1    0100         CALL   TINIT           ;init CTC
C0C9  CD54C1    0101         CALL   SINIT           ;init SIO
C0CC  C9        0102         RET
                0103 ;
                0104 ;       ****************************
                0105 ;       * INITIALIZATION ROUTINES *
                0106 ;       ****************************
                0107 ;
C0CD  AF        0108 INDCE   XOR    A               ;init LAU tables
C0CE  21C0F8    0109         LD     HL,RDCENC
C0D1  CD82C1    0110         CALL   INDCE0          ;nbr calls
C0D4  21D0F8    0111         LD     HL,RDCEHC
C0D7  CD82C1    0112         CALL   INDCE0          ;nbr mapping calls
C0DA  21E0F8    0113         LD     HL,RDCEST
C0DD  3E00      0114         LD     A,INOFFI
C0DF  CD82C1    0115         CALL   INDCE0          ;state
C0E2  21F0F8    0116         LD     HL,RDCERS
C0E5  3E11      0117         LD     A,11H
C0E7  CD82C1    0118         CALL   INDCE0          ;Rst/Clear reason
C0EA  21E0F8    0119         LD     HL,RDCEST
C0ED  3A4CF9    0120         LD     A,(LOCADD)
```

```
C0F0  85          0121          ADD     L
C0F1  6F          0122          LD      L,A
C0F2  3601        0123          LD      (HL),INONI
C0F4  C9          0124          RET
C0F5  DB7D        0125 PERRES   IN      A,(SIOCA)       ;periph reset
C0F7  DB7F        0126          IN      A,(SIOCB)
C0F9  3E18        0127          LD      A,18H
C0FB  D37D        0128          OUT     (SIOCA),A
C0FD  D37F        0129          OUT     (SIOCB),A
C0FF  3E03        0130          LD      A,03H
C101  D3DC        0131          OUT     (CTC0),A
C103  D3DD        0132          OUT     (CTC1),A
C105  D3DE        0133          OUT     (CTC2),A
C107  D3DF        0134          OUT     (CTC3),A
C109  C9          0135          RET
C10A  2198DF      0136 PINIT    LD      HL,PIOTAB       ;PIO init (PIOTAB)
C10D  0EBD        0137          LD      C,PIOCA
C10F  EDA3        0138          OUTI
C111  EDA3        0139          OUTI
C113  0EBF        0140          LD      C,PIOCB
C115  EDA3        0141          OUTI
C117  EDA3        0142          OUTI
C119  EDA3        0143          OUTI
C11B  0EBE        0144          LD      C,PIODB
C11D  EDA3        0145          OUTI
C11F  C9          0146          RET
C120  219EDF      0147 TINIT    LD      HL,TIMTAB       ;CTC init (TIMTAB)
C123  0EDC        0148          LD      C,CTC0
C125  EDA3        0149          OUTI
C127  EDA3        0150          OUTI
C129  EDA3        0151          OUTI
C12B  0EDD        0152          LD      C,CTC1
C12D  EDA3        0153          OUTI
C12F  0EDE        0154          LD      C,CTC2
C131  EDA3        0155          OUTI
C133  EDA3        0156          OUTI
C135  0EDF        0157          LD      C,CTC3
C137  EDA3        0158          OUTI
C139  EDA3        0159          OUTI
C13B  2152F9      0160          LD      HL,TIMRG3       ;init lev3 timers
C13E  360A        0161          LD      (HL),TIME3
C140  2153F9      0162          LD      HL,TMRG3A
C143  360F        0163          LD      (HL),TIME3A
C145  2154F9      0164          LD      HL,TMRG3B
C148  3601        0165          LD      (HL),TIME3B
C14A  2155F9      0166          LD      HL,TMRG3C
C14D  361F        0167          LD      (HL),TIME3C
C14F  AF          0168          XOR     A
C150  3250F9      0169          LD      (INTREG),A
C153  C9          0170          RET
C154  21A6DF      0171 SINIT    LD      HL,SIOTAB       ;init SIO (SIOTAB)
C157  060A        0172          LD      B,0AH
C159  0E7D        0173          LD      C,SIOCA
C15B  EDB3        0174          OTIR
C15D  060A        0175          LD      B,0AH
C15F  0E7F        0176          LD      C,SIOCB
C161  EDB3        0177          OTIR
C163  3E01        0178          LD      A,01H           ;init rcv variab
C165  3222F9      0179          LD      (RBYTE),A
C168  3E10        0180          LD      A,INRCA
```

```
C16A  3220F9      0181          LD      (RCREGA),A
C16D  3E40        0182          LD      A,INTRB         ;init transm variab
C16F  3221F9      0183          LD      (TRREGB),A
C172  C9          0184          RET
C173  0608        0185 INCH0    LD      B,NBCH          ;load chan table
C175  77          0186 INCH1    LD      (HL),A
C176  23          0187          INC     HL
C177  10FC        0188          DJNZ    INCH1
C179  C9          0189          RET
C17A  0638        0190 INRCD0   LD      B,2*NRECD       ;reset records
C17C  AF          0191          XOR     A
C17D  77          0192 INRCD1   LD      (HL),A
C17E  23          0193          INC     HL
C17F  10FC        0194          DJNZ    INRCD1
C181  C9          0195          RET
C182  060B        0196 INDCE0   LD      B,NDCE          ;load LAU table
C184  77          0197 INDCE1   LD      (HL),A
C185  23          0198          INC     HL
C186  10FC        0199          DJNZ    INDCE1
C188  C9          0200          RET
                  0201 ;
                  0202 ;
                  0203 ;        ********************************
                  0204 ;        * INTERRUPT SERVICE ROUTINES *
                  0205 ;        ********************************
                  0206 ;
                  0207 ;
C189  08          0208 TRB      EX      AF,AF'          ;transm interrupt
C18A  D9          0209          EXX                     ;channel B
C18B  2123F9      0210          LD      HL,TBYTE
C18E  35          0211          DEC     (HL)            ;nbr bytes to transm
C18F  280D        0212          JR      Z,TRB1
C191  0E7E        0213          LD      C,SIODB
C193  2A26F9      0214          LD      HL,(TPOINT)     ;read transm point
C196  EDA3        0215          OUTI                    ;write byte (SIO)
C198  2226F9      0216          LD      (TPOINT),HL     ;save pointer
C19B  C355C3      0217          JP      NOINT
C19E  3E28        0218 TRB1     LD      A,28H           ;end of frame
C1A0  D37F        0219          OUT     (SIOCB),A       ;reset Tx Int pending
C1A2  3601        0220          LD      (HL),01
C1A4  2121F9      0221          LD      HL,TRREGB
C1A7  CBCE        0222          SET     EOM,(HL)        ;EOM=1
C1A9  CB76        0223          BIT     UND,(HL)        ;test UND
C1AB  CA55C3      0224          JP      Z,NOINT
C1AE  3640        0225          LD      (HL),INTRB      ;end of transm (CRC)
C1B0  3EA8        0226          LD      A,0A8H          ;reset CRC gen/Tx Int
C1B2  D37F        0227          OUT     (SIOCB),A
C1B4  3A66F9      0228          LD      A,(NIF5)        ;update LOQ counters
C1B7  326AF9      0229          LD      (NIF4),A
C1BA  CD20C3      0230          CALL    TX              ;new transm
C1BD  C355C3      0231          JP      NOINT
                  0232 ;
C1C0  08          0233 EXTB     EX      AF,AF'          ;ext/stat interrupt
C1C1  D9          0234          EXX                     ;channel B
C1C2  DB7F        0235          IN      A,(SIOCB)       ;read SIO state reg
C1C4  4F          0236          LD      C,A
C1C5  3E10        0237          LD      A,10H
C1C7  D37F        0238          OUT     (SIOCB),A       ;reset ext/stat Int
C1C9  79          0239          LD      A,C
C1CA  E644        0240          AND     44H             ;mask UND/TXBUF (SIO)
```

```
C1CC  2121F9      0241          LD      HL,TRREGB
C1CF  AE          0242          XOR     (HL)          ;compare TRREGB
C1D0  CB77        0243          BIT     UND,A         ;test und interrupt
C1D2  CA55C3      0244          JP      Z,NOINT
C1D5  CBF6        0245          SET     UND,(HL)      ;und int served
C1D7  FE43        0246          CP      43H           ;test end of transm
C1D9  CA55C3      0247          JP      Z,NOINT
C1DC  CB4F        0248          BIT     EOM,A         ;test val end transm
C1DE  2812        0249          JR      Z,UND0
C1E0  3640        0250          LD      (HL),INTRB    ;valid end of transm
C1E2  3EA8        0251          LD      A,0A8H
C1E4  D37F        0252          OUT     (SIOCB),A     ;reset CRC gen/Tx Int
C1E6  3A66F9      0253          LD      A,(NIF5)      ;update LOQ counters
C1E9  326AF9      0254          LD      (NIF4),A
C1EC  CD20C3      0255          CALL    TX            ;new transm
C1EF  C355C3      0256          JP      NOINT
C1F2  3E08        0257 UND0     LD      A,08H         ;underrun/send abort
C1F4  D37F        0258          OUT     (SIOCB),A
C1F6  3603        0259          LD      (HL),TRABT    ;prepare retransm
C1F8  3E01        0260          LD      A,01
C1FA  3223F9      0261          LD      (TBYTE),A
C1FD  2166F9      0262          LD      HL,NIF5
C200  34          0263          INC     (HL)          ;inc NIF5
C201  216BF9      0264          LD      HL,DIMQ6
C204  35          0265          DEC     (HL)          ;dec transm point
C205  3EE8        0266          LD      A,0E8H
C207  D37F        0267          OUT     (SIOCB),A     ;reset Und/Tx Int
C209  C355C3      0268          JP      NOINT
                  0269 ;
C20C  00          0270 RECB     NOP                   ;rcv/sp rcv Int
C20D  08          0271 SRB      EX      AF,AF'        ;channel B (unused)
C20E  D9          0272          EXX
C20F  3E33        0273          LD      A,33H         ;disable rcv
C211  D37F        0274          OUT     (SIOCB),A
C213  DB7E        0275          IN      A,(SIODB)
C215  C355C3      0276          JP      NOINT
                  0277 ;
C218  08          0278 TRA      EX      AF,AF'        ;transm Int
C219  D9          0279          EXX                   ;channel A (unused)
C21A  3E2D        0280          LD      A,2DH         ;disable transm
C21C  D37D        0281          OUT     (SIOCA),A
C21E  3E00        0282          LD      A,00H
C220  D37D        0283          OUT     (SIOCA),A
C222  C355C3      0284          JP      NOINT
                  0285 ;
C225  08          0286 EXTA     EX      AF,AF'        ;ext/stat interrupt
C226  D9          0287          EXX                   ;channel A
C227  DB7D        0288          IN      A,(SIOCA)     ;read SIO state reg
C229  2120F9      0289          LD      HL,RCREGA
C22C  46          0290          LD      B,(HL)
C22D  A8          0291          XOR     B             ;compare RCREGA
C22E  4F          0292          LD      C,A
C22F  3E10        0293          LD      A,10H
C231  D37D        0294          OUT     (SIOCA),A
C233  CB61        0295          BIT     HT,C          ;test hunt interrupt
C235  280A        0296          JR      Z,EXTA1
C237  CB60        0297 HUNT0    BIT     HT,B
C239  2804        0298          JR      Z,HUNT1
C23B  CBA6        0299          RES     HT,(HL)
C23D  1802        0300          JR      EXTA1
```

```
C23F CBE6        0301 HUNT1  SET   HT,(HL)
C241 CB79        0302 EXTA1  BIT   AB,C          ;test abort interrupt
C243 CA55C3      0303        JP    Z,NOINT
C246 CB78        0304 ABORT0 BIT   AB,B          ;test begin/end abort
C248 2805        0305        JR    Z,ABORT1
C24A CBBE        0306        RES   AB,(HL)       ;end abort
C24C C355C3      0307        JP    NOINT
C24F CBFE        0308 ABORT1 SET   AB,(HL)       ;begin abort detection
C251 CB46        0309        BIT   RC,(HL)       ;test reception
C253 CA55C3      0310        JP    Z,NOINT
C256 CB86        0311        RES   RC,(HL)       ;cancel reception
C258 3E01        0312        LD    A,01
C25A 3222F9      0313        LD    (RBYTE),A
C25D 2162F9      0314        LD    HL,DLQ1
C260 35          0315        DEC   (HL)
C261 2163F9      0316        LD    HL,NOFB
C264 34          0317        INC   (HL)
C265 C355C3      0318        JP    NOINT
                 0319 ;
C268 08          0320 RECA   EX    AF,AF'        ;receive interrupt
C269 D9          0321        EXX                 ;channel A
C26A 2122F9      0322        LD    HL,RBYTE      ;test buffer space
C26D 35          0323        DEC   (HL)
C26E 280D        0324        JR    Z,RECA1
C270 0E7C        0325        LD    C,SIODA
C272 2A24F9      0326        LD    HL,(RPOINT)   ;read rcv point
C275 EDA2        0327        INI                 ;read byte (SIO)
C277 2224F9      0328        LD    (RPOINT),HL   ;save pointer
C27A C355C3      0329        JP    NOINT
C27D 3601        0330 RECA1  LD    (HL),01       ;no space
C27F 2120F9      0331        LD    HL,RCREGA
C282 CB46        0332        BIT   RC,(HL)       ;test reception
C284 2805        0333        JR    Z,RECA2
C286 DB7C        0334        IN    A,(SIODA)     ;overflow
C288 C355C3      0335        JP    NOINT
C28B 3A63F9      0336 RECA2  LD    A,(NOFB)      ;begin/get buffer
C28E B7          0337        OR    A
C28F 201B        0338        JR    NZ,RECA3
C291 3E03        0339        LD    A,03          ;no buf/cancel rcv
C293 D37D        0340        OUT   (SIOCA),A
C295 3ED0        0341        LD    A,0D0H
C297 D37D        0342        OUT   (SIOCA),A
C299 DB7C        0343        IN    A,(SIODA)     ;clear rcv FIFO
C29B DB7C        0344        IN    A,(SIODA)
C29D DB7C        0345        IN    A,(SIODA)
C29F 3E73        0346        LD    A,73H
C2A1 D37D        0347        OUT   (SIOCA),A
C2A3 3ED9        0348        LD    A,0D9H        ;enable rcv/hunt
C2A5 D37D        0349        OUT   (SIOCA),A
C2A7 CBE6        0350        SET   HT,(HL)
C2A9 C355C3      0351        JP    NOINT
C2AC 3D          0352 RECA3  DEC   A             ;buf available
C2AD 3263F9      0353        LD    (NOFB),A
C2B0 CBC6        0354        SET   RC,(HL)       ;begin reception
C2B2 2162F9      0355        LD    HL,DLQ1       ;get buffer
C2B5 7E          0356        LD    A,(HL)
C2B6 34          0357        INC   (HL)
C2B7 E60F        0358        AND   QMASK
C2B9 07          0359        RLCA
C2BA 2100F8      0360        LD    HL,TPQ1
```

```
C2BD B5          0361         OR    L
C2BE 6F          0362         LD    L,A
C2BF 7E          0363         LD    A,(HL)
C2C0 2C          0364         INC   L
C2C1 66          0365         LD    H,(HL)
C2C2 6F          0366         LD    L,A
C2C3 2228F9      0367         LD    (RBUF),HL      ;save buf address
C2C6 23          0368         INC   HL             ;reserve length byte
C2C7 0E7C        0369         LD    C,SIODA
C2C9 EDA2        0370         INI                  ;read byte
C2CB 2224F9      0371         LD    (RPOINT),HL    ;save rcv point
C2CE 3E87        0372         LD    A,BUFL-1
C2D0 3222F9      0373         LD    (RBYTE),A      ;current buf space
C2D3 C355C3      0374         JP    NOINT
                 0375 ;
C2D6 08          0376 SRA     EX    AF,AF'         ;spec rcv interrupt
C2D7 D9          0377         EXX                  ;channel A
C2D8 3E01        0378         LD    A,01H
C2DA D37D        0379         OUT   (SIOCA),A
C2DC DB7D        0380         IN    A,(SIOCA)      ;read SIO state reg
C2DE E6E0        0381         AND   0E0H           ;mask rcv flags
C2E0 2120F9      0382         LD    HL,RCREGA
C2E3 FE80        0383         CP    A,80H          ;test val end rcv
C2E5 2821        0384         JR    Z,SRA2
C2E7 CB7F        0385         BIT   7,A            ;inv/test end rcv
C2E9 2804        0386         JR    Z,SRA1
C2EB 3E30        0387         LD    A,30H          ;inv end reception
C2ED D37D        0388         OUT   (SIOCA),A      ;error reset (SIO)
C2EF DB7C        0389 SRA1    IN    A,(SIODA)      ;read byte
C2F1 CB46        0390         BIT   RC,(HL)        ;test reception
C2F3 CA55C3      0391         JP    Z,NOINT
C2F6 CB86        0392         RES   RC,(HL)        ;cancel normal rcv
C2F8 3E01        0393         LD    A,01           ;rcv stay enabled
C2FA 3222F9      0394         LD    (RBYTE),A
C2FD 2162F9      0395         LD    HL,DLQ1        ;return buffer
C300 35          0396         DEC   (HL)
C301 2163F9      0397         LD    HL,NOFB
C304 34          0398         INC   (HL)
C305 C355C3      0399         JP    NOINT
C308 DB7C        0400 SRA2    IN    A,(SIODA)      ;valid end reception
C30A CB86        0401         RES   RC,(HL)
C30C 2160F9      0402         LD    HL,CTQ1        ;transf to LOQ
C30F 34          0403         INC   (HL)
C310 2122F9      0404         LD    HL,RBYTE
C313 46          0405         LD    B,(HL)         ;available length
C314 3601        0406         LD    (HL),01        ;init RBYTE
C316 3E88        0407         LD    A,BUFL
C318 90          0408         SUB   B              ;frame length
C319 2A28F9      0409         LD    HL,(RBUF)      ;insert length byte
C31C 77          0410         LD    (HL),A
C31D C355C3      0411         JP    NOINT
                 0412 ;
C320 B7          0413 TX      OR    A              ;test current transm
C321 C8          0414         RET   Z
C322 3D          0415         DEC   A              ;begin new transm
C323 3266F9      0416         LD    (NIF5),A
C326 216BF9      0417         LD    HL,DIMQ6       ;get frame (LOQ)
C329 7E          0418         LD    A,(HL)
C32A 34          0419         INC   (HL)
C32B E60F        0420         AND   QMASK
```

```
C32D  07        0421          RLCA
C32E  21A0F8    0422          LD      HL,TPQ6
C331  B5        0423          OR      L
C332  6F        0424          LD      L,A
C333  7E        0425          LD      A,(HL)
C334  2C        0426          INC     L
C335  66        0427          LD      H,(HL)
C336  6F        0428          LD      L,A
C337  7E        0429          LD      A,(HL)          ;read length
C338  23        0430          INC     HL              ;point next byte
C339  3D        0431          DEC     A
C33A  3223F9    0432          LD      (TBYTE),A       ;init TBYTE
C33D  0E7E      0433          LD      C,SIODB
C33F  EDA3      0434          OUTI                    ;write first byte
C341  2226F9    0435          LD      (TPOINT),HL     ;save pointer
C344  2121F9    0436          LD      HL,TRREGB
C347  3601      0437          LD      (HL),TRFRM
C349  3EC0      0438          LD      A,0C0H          ;reset UND (SIO)
C34B  D37F      0439          OUT     (SIOCB),A
C34D  C9        0440          RET
                0441 ;
C34E  08        0442 TIMER    EX      AF,AF'          ;timer interrupt
C34F  D9        0443          EXX
C350  2150F9    0444          LD      HL,INTREG
C353  CBFE      0445          SET     MTIM,(HL)       ;set master timer
C355  D9        0446 NOINT    EXX
C356  08        0447          EX      AF,AF'
C357  FB        0448 ENDINT   EI
C358  ED4D      0449          RETI
                0450 ;
                0451 ;
                0452 ;        *****************
                0453 ;        * TIMER MANAGER *
                0454 ;        *****************
                0455 ;
                0456 ;
C35A  3A50F9    0457 TIMNG    LD      A,(INTREG)      ;timer manager
C35D  CB7F      0458          BIT     MTIM,A          ;test master timer
C35F  C8        0459          RET     Z
C360  CBBF      0460          RES     MTIM,A
C362  CB4F      0461          BIT     T2ON,A          ;test lev2 timer
C364  2808      0462          JR      Z,TIMER1
C366  2151F9    0463          LD      HL,TIMRG2       ;lev2 timer ON/update
C369  35        0464          DEC     (HL)
C36A  2002      0465          JR      NZ,TIMER1
C36C  CBC7      0466          SET     T2OUT,A         ;lev2 time-out
C36E  2152F9    0467 TIMER1   LD      HL,TIMRG3       ;test lev3 master tim
C371  35        0468          DEC     (HL)
C372  2020      0469          JR      NZ,TIMER4
C374  360A      0470          LD      (HL),TIME3      ;ON/update
C376  2153F9    0471          LD      HL,TMRG3A
C379  35        0472          DEC     (HL)            ;update X25 Int tim
C37A  2004      0473          JR      NZ,TIMER2
C37C  360F      0474          LD      (HL),TIME3A     ;time-out (X.25 interf)
C37E  CBD7      0475          SET     T3OUT,A
C380  2154F9    0476 TIMER2   LD      HL,TMRG3B
C383  35        0477          DEC     (HL)            ;update LAU comm tim
C384  2004      0478          JR      NZ,TIMER3
C386  3601      0479          LD      (HL),TIME3B     ;time-out (LAU command)
C388  CBDF      0480          SET     TBOUT,A
```

```
C38A  2155F9      0481 TIMER3  LD     HL,TMRG3C
C38D  35          0482         DEC    (HL)          ;update LAU recov tim
C38E  2004        0483         JR     NZ,TIMER4
C390  361F        0484         LD     (HL),TIME3C   ;time-out (LAU recov)
C392  CBE7        0485         SET    TBOUT1,A
C394  CB6F        0486 TIMER4  BIT    TRACE,A       ;monitor trace timer
C396  2810        0487         JR     Z,TIMER5
C398  219AF9      0488         LD     HL,ACTC1      ;update trace timer
C39B  35          0489         DEC    (HL)
C39C  200A        0490         JR     NZ,TIMER5
C39E  360A        0491         LD     (HL),ACTIM1
C3A0  219BF9      0492         LD     HL,ACTC2
C3A3  35          0493         DEC    (HL)
C3A4  2002        0494         JR     NZ,TIMER5
C3A6  CBF7        0495         SET    ENDTRC,A      ;end of trace interval
C3A8  3250F9      0496 TIMER5  LD     (INTREG),A
C3AB  CD2FDC      0497         CALL   MONTM         ;monitor timer
C3AE  C9          0498         RET
                  0499 ;
                  0500 ;
                  0501 ;       ******************
                  0502 ;       * BUFFER MANAGER *
                  0503 ;       ******************
                  0504 ;
                  0505 ;
C3AF  2140F9      0506 BFMNG   LD     HL,BUFREG     ;buf manager
C3B2  CB46        0507         BIT    BY,(HL)       ;test busy
C3B4  2010        0508         JR     NZ,BFMNG1
C3B6  3A63F9      0509         LD     A,(NOFB)      ;not busy
C3B9  D602        0510         SUB    NFB0          ;NOFB-NFB0
C3BB  3033        0511         JR     NC,BFMNG3     ;test busy limit
C3BD  CBC6        0512         SET    BY,(HL)       ;enter busy
C3BF  0E04        0513         LD     C,NGB0
C3C1  CDF1C3      0514         CALL   BFMNGA        ;assign buf (busy)
C3C4  182A        0515         JR     BFMNG3
C3C6  3A63F9      0516 BFMNG1  LD     A,(NOFB)      ;busy
C3C9  47          0517         LD     B,A           ;NOFB
C3CA  3A64F9      0518         LD     A,(NOGB)
C3CD  4F          0519         LD     C,A           ;NOGB
C3CE  80          0520         ADD    B             ;NOFB+NOGB
C3CF  FE08        0521         CP     N1            ;compare with N1
C3D1  3813        0522         JR     C,BFMNG2
C3D3  CB86        0523         RES    BY,(HL)       ;clear busy
C3D5  79          0524         LD     A,C           ;reassign buffers
C3D6  D604        0525         SUB    NGB0          ;NOGB-NGB0
C3D8  2163F9      0526         LD     HL,NOFB       ;NOFB
C3DB  F3          0527         DI
C3DC  86          0528         ADD    (HL)          ;add NOGB-NGB0
C3DD  77          0529         LD     (HL),A
C3DE  FB          0530         EI
C3DF  3E04        0531         LD     A,NGB0
C3E1  3264F9      0532         LD     (NOGB),A      ;NOGB=NGB0
C3E4  180A        0533         JR     BFMNG3
C3E6  FE03        0534 BFMNG2  CP     NFB0+1        ;stay busy
C3E8  3806        0535         JR     C,BFMNG3
C3EA  78          0536         LD     A,B           ;NOFB
C3EB  D602        0537         SUB    NFB0          ;NOFB-NFB0
C3ED  CDF1C3      0538         CALL   BFMNGA        ;assign buf (busy)
C3F0  C9          0539 BFMNG3  RET
                  0540 ;
```

```
C3F1  47        0541 BFMNGA  LD    B,A            ;NOFB-NFB0(<=0)
C3F2  ED44      0542         NEG                  ;NFB0-NOFB
C3F4  2163F9    0543         LD    HL,NOFB        ;NOFB
C3F7  F3        0544         DI
C3F8  86        0545         ADD   (HL)           ;add NFB0-NOFB
C3F9  77        0546         LD    (HL),A
C3FA  FB        0547         EI
C3FB  79        0548         LD    A,C            ;NOGB
C3FC  80        0549         ADD   B              ;add NOFB-NFB0
C3FD  3264F9    0550         LD    (NOGB),A
C400  C9        0551         RET
                0552 ;
                0553 ;       ****************************
                0554 ;       * BUFFER TRANSFER ROUTINES *
                0555 ;       ****************************
                0556 ;
C401  E60F      0557 GETBUF  AND   QMASK          ;get buffer
C403  07        0558         RLCA                 ;entry offset
C404  B5        0559         OR    L
C405  6F        0560         LD    L,A            ;point to entry
C406  7E        0561         LD    A,(HL)         ;address LSB
C407  2C        0562         INC   L
C408  66        0563         LD    H,(HL)         ;address MSB
C409  6F        0564         LD    L,A
C40A  2256F9    0565         LD    (CURBUF),HL    ;save in CURBUF
C40D  C9        0566         RET
C40E  E60F      0567 PUTBUF  AND   QMASK          ;put buffer
C410  07        0568         RLCA                 ;entry offset
C411  B5        0569         OR    L
C412  6F        0570         LD    L,A            ;point to entry
C413  3A56F9    0571         LD    A,(CURBUF)     ;address LSB
C416  77        0572         LD    (HL),A
C417  2C        0573         INC   L
C418  3A57F9    0574         LD    A,(CURBUF+1)   ;address MSB
C41B  77        0575         LD    (HL),A
C41C  C9        0576         RET
C41D  2163F9    0577 GTPOOL  LD    HL,NOFB        ;get buf (pool)
C420  3E02      0578         LD    A,NFB0
C422  F3        0579         DI
C423  BE        0580         CP    (HL)           ;test Busy limit
C424  3004      0581         JR    NC,GTP1
C426  35        0582         DEC   (HL)           ;not Busy (dec NOFB)
C427  FB        0583         EI
C428  180A      0584         JR    GTP2
C42A  FB        0585 GTP1    EI                   ;set Busy (dec NOGB)
C42B  2164F9    0586         LD    HL,NOGB
C42E  35        0587         DEC   (HL)
C42F  2140F9    0588         LD    HL,BUFREG
C432  CBC6      0589         SET   BY,(HL)
C434  2165F9    0590 GTP2    LD    HL,DPOOL       ;get buffer
C437  35        0591         DEC   (HL)
C438  7E        0592         LD    A,(HL)
C439  2100F8    0593         LD    HL,TPOOL
C43C  CD01C4    0594         CALL  GETBUF
C43F  C9        0595         RET
C440  2165F9    0596 RELBUF  LD    HL,DPOOL       ;release buf
C443  7E        0597         LD    A,(HL)
C444  34        0598         INC   (HL)
C445  2100F8    0599         LD    HL,TPOOL
C448  CD0EC4    0600         CALL  PUTBUF
```

```
C49C  2805      0661          JR    Z,INTLP0
C49E  CB96      0662          RES   PD,(HL)        ;init discn
C4A0  CDF7CA    0663          CALL  DISCN0
C4A3  C9        0664 INTLP0   RET
                0665 ;
                0666 ;        **********************************
                0667 ;        * REPROCESS TRANSMITTED FRAMES *
                0668 ;        **********************************
                0669 ;
C4A4  3A3AF9    0670 REPFR    LD    A,(NTF)       ;repr trm frames
C4A7  B7        0671          OR    A
C4A8  CA10C5    0672          JP    Z,REPFR8
C4AB  57        0673          LD    D,A           ;nbr fr to reproc
C4AC  2167F9    0674 REPFR0   LD    HL,DFQ6       ;get frame (LOQ)
C4AF  7E        0675          LD    A,(HL)
C4B0  34        0676          INC   (HL)
C4B1  21A0F8    0677          LD    HL,TPQ6
C4B4  CD01C4    0678          CALL  GETBUF
C4B7  23        0679          INC   HL            ;point addr field
C4B8  7E        0680          LD    A,(HL)        ;read addr field
C4B9  23        0681          INC   HL            ;point contr field
C4BA  FE01      0682          CP    TRESAD        ;test resp frame
C4BC  200B      0683          JR    NZ,REPFR1
C4BE  66        0684          LD    H,(HL)        ;response frame
C4BF  2E4C      0685          LD    L,M76
C4C1  CDDDDB    0686          CALL  LDMES
C4C4  CD40C4    0687          CALL  RELBUF        ;release buff
C4C7  1844      0688          JR    REPFR7
C4C9  7E        0689 REPFR1   LD    A,(HL)        ;not resp/read contr
C4CA  CB47      0690          BIT   0,A           ;test comm type
C4CC  280A      0691          JR    Z,REPFR2
C4CE  2E4A      0692          LD    L,M74         ;SARM/DISC
C4D0  67        0693          LD    H,A
C4D1  CDDDDB    0694          CALL  LDMES
C4D4  CB99      0695          RES   WAIT,C        ;clear WAIT cond
C4D6  1835      0696          JR    REPFR7
C4D8  2E4B      0697 REPFR2   LD    L,M75         ;INFORMATION
C4DA  67        0698          LD    H,A
C4DB  CDDDDB    0699          CALL  LDMES
C4DE  2173F9    0700          LD    HL,NIF2
C4E1  35        0701          DEC   (HL)          ;dec nbr I in LOQ
C4E2  CB59      0702          BIT   WAIT,C        ;repr INF/test WAIT
C4E4  2823      0703          JR    Z,REPFR6
C4E6  CB61      0704          BIT   REC,C         ;test recovery
C4E8  2011      0705          JR    NZ,REPFR4
C4EA  CD4CCC    0706          CALL  STP           ;not recov but WAIT
C4ED  CB41      0707          BIT   COM,C         ;test SARM/DISC outs
C4EF  2805      0708          JR    Z,REPFR3
C4F1  CD54CC    0709          CALL  START         ;SAR/DIS out/start tim
C4F4  1817      0710          JR    REPFR7
C4F6  CD9BCB    0711 REPFR3   CALL  RETR          ;no SAR/DIS test NIF2
C4F9  1812      0712          JR    REPFR7
C4FB  CB67      0713 REPFR4   BIT   4,A           ;recov/test P sent
C4FD  2804      0714          JR    Z,REPFR5
C4FF  CBE9      0715          SET   PS,C          ;P bit sent (set PS)
C501  180A      0716          JR    REPFR7
C503  CD4CCC    0717 REPFR5   CALL  STP           ;P not sent
C506  CD9BCB    0718          CALL  RETR
C509  2132F9    0719 REPFR6   LD    HL,MAXNS
C50C  34        0720          INC   (HL)          ;inc MAXNS outs
```

```
C50D 15          0721 REPFR7  DEC   D
C50E 209C        0722         JR    NZ,REPFR0      ;repeat transm fram
C510 C9          0723 REPFR8  RET
                 0724 ;
                 0725 ;       **********************************
                 0726 ;       * PROCESS RECEIVED FRAME (LIQ) *
                 0727 ;       **********************************
                 0728 ;
C511 3A3BF9      0729 LIQ     LD    A,(NRF)        ;read nbr rcv frames
C514 B7          0730         OR    A
C515 CA66C8      0731         JP    Z,LIQ1
C518 DD2140FA    0732         LD    IX,FRMA        ;process one frame
C51C CDE3D6      0733         CALL  INCRCD
C51F 2160F9      0734         LD    HL,CTQ1        ;dec LIQ counter
C522 35          0735         DEC   (HL)
C523 2161F9      0736         LD    HL,DFQ1        ;get frame
C526 7E          0737         LD    A,(HL)
C527 34          0738         INC   (HL)
C528 2100F8      0739         LD    HL,TPQ1
C52B CD01C4      0740         CALL  GETBUF
C52E 7E          0741         LD    A,(HL)         ;read length
C52F D603        0742         SUB   FLMIN          ;test short frame
C531 300D        0743         JR    NC,LIQ0
C533 2E4D        0744         LD    L,M77          ;inv (short) frame
C535 2600        0745         LD    H,00
C537 CDDDDB      0746         CALL  LDMES
C53A CD40C4      0747         CALL  RELBUF
C53D C381C8      0748         JP    ENDLIQ
C540 57          0749 LIQ0    LD    D,A            ;nbr Inf bytes
C541 23          0750         INC   HL             ;point addr field
C542 7E          0751         LD    A,(HL)         ;read addr field
C543 23          0752         INC   HL             ;point contr field
C544 5E          0753         LD    E,(HL)         ;read contr field
C545 FE03        0754 DECADD  CP    RRESAD         ;test addr field
C547 2826        0755         JR    Z,DCRES
C549 FE01        0756         CP    RCOMAD
C54B 280D        0757         JR    Z,DCCOM
C54D 2E4D        0758         LD    L,M77          ;invalid address
C54F 2601        0759         LD    H,01
C551 CDDDDB      0760         CALL  LDMES
C554 CD40C4      0761         CALL  RELBUF
C557 C381C8      0762         JP    ENDLIQ
C55A 7B          0763 DCCOM   LD    A,E            ;command/decode
C55B CB47        0764         BIT   0,A            ;test INF
C55D CA36C6      0765         JP    Z,INF
C560 E6EF        0766         AND   0EFH           ;not INF/mask P bit
C562 FE0F        0767         CP    0FH            ;test SARM
C564 CA92C5      0768         JP    Z,SARM
C567 FE43        0769         CP    43H            ;test DISC
C569 CADAC5      0770         JP    Z,DISC
C56C C31DC6      0771         JP    INVCOM         ;invalid command
C56F 7B          0772 DCRES   LD    A,E            ;response/decode
C570 CB47        0773         BIT   0,A
C572 CA5BC8      0774         JP    Z,INVRES
C575 CB4F        0775         BIT   1,A            ;test resp type
C577 280F        0776         JR    Z,DCRES1
C579 E6EF        0777         AND   0EFH           ;not SUP/mask F bit
C57B FE63        0778         CP    63H            ;test UA
C57D CA17C7      0779         JP    Z,UA
C580 FE87        0780         CP    87H            ;test CMDR
```

```
C582  CA28C8      0781          JP    Z,CMDR
C585  C35BC8      0782          JP    INVRES
C588  E60F        0783 DCRES1   AND   0FH          ;SUP/mask N(R)
C58A  FE0D        0784          CP    0DH          ;test valid SUP
C58C  CA5BC8      0785          JP    Z,INVRES
C58F  C386C7      0786          JP    SUP
                  0787 ;
C592  AF          0788 SARM     XOR   A            ;SARM/test length
C593  BA          0789          CP    D
C594  280A        0790          JR    Z,SARM1
C596  3E03        0791          LD    A,03         ;invalid length
C598  3238F9      0792          LD    (REJRSN),A   ;reject SARM
C59B  CD28C6      0793          CALL  REJCM
C59E  1837        0794          JR    SARM4
C5A0  2E08        0795 SARM1    LD    L,M08        ;valid SARM
C5A2  2602        0796          LD    H,M02
C5A4  CDDDDB      0797          CALL  LDMES
C5A7  DD2148FA    0798          LD    IX,COMA
C5AB  CDE3D6      0799          CALL  INCRCD
C5AE  CB40        0800          BIT   LF,B         ;test link state
C5B0  2808        0801          JR    Z,SARM2
C5B2  CD5CC4      0802          CALL  MREL         ;failure state
C5B5  CDCBCA      0803          CALL  SDIS         ;resynchr DISC
C5B8  181D        0804          JR    SARM4
C5BA  CB50        0805 SARM2    BIT   DIS,B        ;test discn
C5BC  2805        0806          JR    Z,SARM3
C5BE  CD5CC4      0807          CALL  MREL         ;in disconnection
C5C1  1814        0808          JR    SARM4
C5C3  CDC6C9      0809 SARM3    CALL  TUA          ;accept/ack SARM
C5C6  CBD8        0810          SET   SEC,B        ;update state
C5C8  3A35F9      0811          LD    A,(SEQNBR)   ;V(R)=0
C5CB  E60E        0812          AND   0EH
C5CD  3235F9      0813          LD    (SEQNBR),A
C5D0  CB48        0814          BIT   PRIM,B       ;test prim state
C5D2  2003        0815          JR    NZ,SARM4
C5D4  CDB5CA      0816          CALL  SSAR         ;set-up primary
C5D7  C366C8      0817 SARM4    JP    LIQ1
                  0818 ;
C5DA  AF          0819 DISC     XOR   A            ;DISC/test length
C5DB  BA          0820          CP    D
C5DC  280A        0821          JR    Z,DISC1
C5DE  3E03        0822          LD    A,03         ;invalid length
C5E0  3238F9      0823          LD    (REJRSN),A   ;reject DISC
C5E3  CD28C6      0824          CALL  REJCM
C5E6  1832        0825          JR    DISC5
C5E8  2E09        0826 DISC1    LD    L,M09        ;valid DISC
C5EA  2602        0827          LD    H,M02
C5EC  CDDDDB      0828          CALL  LDMES
C5EF  DD2148FA    0829          LD    IX,COMA
C5F3  CDE3D6      0830          CALL  INCRCD
C5F6  CDC6C9      0831          CALL  TUA          ;accept/ack DISC
C5F9  CB48        0832          BIT   PRIM,B       ;test prim state
C5FB  2004        0833          JR    NZ,DISC2
C5FD  CB58        0834          BIT   SEC,B        ;test sec state
C5FF  2807        0835          JR    Z,DISC3
C601  CB98        0836 DISC2    RES   SEC,B        ;real discn
C603  CDEFCA      0837          CALL  DISCN        ;init prim discn
C606  1812        0838          JR    DISC5
C608  CB40        0839 DISC3    BIT   LF,B         ;disconected/test fail
C60A  2805        0840          JR    Z,DISC4
```

```
C60C CDCBCA      0841          CALL   SDIS          ;fail/resynch DISC
C60F 1809        0842          JR     DISC5
C611 CB41        0843 DISC4    BIT    COM,C         ;clear set-up delay
C613 2005        0844          JR .   NZ,DISC5
C615 CB99        0845          RES    WAIT,C
C617 CD4CCC      0846          CALL   STP
C61A C366C8      0847 DISC5    JP     LIQ1
                 0848 ;
C61D 3E01        0849 INVCOM   LD     A,01          ;invalid command
C61F 3238F9      0850          LD     (REJRSN),A    ;reject command
C622 CD28C6      0851          CALL   REJCM
C625 C366C8      0852          JP     LIQ1
C628 7B          0853 REJCM    LD     A,E           ;answer inv comm
C629 3239F9      0854          LD     (REJCTR),A
C62C 2E4F        0855          LD     L,M79
C62E 63          0856          LD     H,E
C62F CDDDDB      0857          CALL   LDMES
C632 CD33CA      0858          CALL   TCMDR
C635 C9          0859          RET
                 0860 ;
C636 CB58        0861 INF      BIT    SEC,B         ;process INF
C638 201A        0862          JR     NZ,INF2
C63A 2E19        0863          LD     L,M25         ;sec discn (error)
C63C 2602        0864          LD     H,M02
C63E CDDDDB      0865          CALL   LDMES
C641 CD40C4      0866          CALL   RELBUF
C644 CB40        0867          BIT    LF,B          ;test fail state
C646 2806        0868          JR     Z,INF1
C648 CDCBCA      0869          CALL   SDIS          ;resynchr DISC
C64B C302C7      0870          JP     ENDINF
C64E CDEFCA      0871 INF1     CALL   DISCN         ;init discn (error)
C651 C302C7      0872          JP     ENDINF
C654 CB50        0873 INF2     BIT    DIS,B         ;sec connect/test discn
C656 2806        0874          JR     Z,INF3
C658 CD5CC4      0875          CALL   MREL          ;in disconnection
C65B C302C7      0876          JP     ENDINF
C65E FD2A56F9    0877 INF3     LD     IY,(CURBUF)
C662 CD22CB      0878          CALL   NRI           ;process N(R)
C665 FD2256F9    0879          LD     (CURBUF),IY
C669 CB60        0880          BIT    SEC0,B        ;test CMDR outst
C66B 2806        0881          JR     Z,INF4
C66D CD33CA      0882          CALL   TCMDR         ;repeat CMDR
C670 C302C7      0883          JP     ENDINF
C673 3E83        0884 INF4     LD     A,ILMAX       ;test max length
C675 92          0885          SUB    D
C676 3008        0886          JR     NC,INF5
C678 3E04        0887          LD     A,04          ;inv length(>max)
C67A 3238F9      0888          LD     (REJRSN),A    ;reject INF
C67D CD28C6      0889          CALL   REJCM
C680 C302C7      0890          JP     ENDINF
C683 2E42        0891 INF5     LD     L,M66
C685 63          0892          LD     H,E
C686 CDDDDB      0893          CALL   LDMES
C689 DD2144FA    0894          LD     IX,INFA
C68D CDE3D6      0895          CALL   INCRCD
C690 3A35F9      0896          LD     A,(SEQNBR)    ;get current V(R)
C693 07          0897          RLCA
C694 07          0898          RLCA
C695 07          0899          RLCA
C696 07          0900          RLCA
```

```
C697 93          0901        SUB   E
C698 E60F        0902        AND   0FH          ;compare with N(S)
C69A 203D        0903        JR    NZ,INF8
C69C CBB0        0904        RES   SEC2,B       ;N(S)=V(R)/clear REJ
C69E 3A40F9      0905        LD    A,(BUFREG)
C6A1 CB47        0906        BIT   BY,A         ;test BUSY
C6A3 2806        0907        JR    Z,INF6
C6A5 CD05C7      0908        CALL  IBUSY        ;BUSY condition
C6A8 C3C2C7      0909        JP    ENDINF
C6AB 3A35F9      0910 INF6   LD    A,(SEQNBR)   ;not BUSY/accept
C6AE C620        0911        ADD   20H          ;inc V(R)
C6B0 3235F9      0912        LD    (SEQNBR),A
C6B3 216EF9      0913        LD    HL,DLQ2      ;transf INF to PIQ
C6B6 7E          0914        LD    A,(HL)
C6B7 34          0915        INC   (HL)
C6B8 2120F8      0916        LD    HL,TPQ2
C6BB CD0EC4      0917        CALL  PUTBUF
C6BE 216CF9      0918        LD    HL,CTQ2      ;inc PIQ count
C6C1 34          0919        INC   (HL)
C6C2 CB63        0920        BIT   4,E          ;test P bit
C6C4 2009        0921        JR    NZ,INF7
C6C6 CB68        0922        BIT   SEC1,B       ;P=0/test RNR sent
C6C8 2005        0923        JR    NZ,INF7
C6CA CBF9        0924        SET   PGB,C        ;P=0;RNR=0/pigg ack
C6CC C3C2C7      0925        JP    ENDINF
C6CF CD1DC4      0926 INF7   CALL  GTPOOL       ;send RR/get buff
C6D2 CBA8        0927        RES   SEC1,B       ;clear RNR sent
C6D4 CDE8C9      0928        CALL  TRR
C6D7 1829        0929        JR    ENDINF
C6D9 3A40F9      0930 INF8   LD    A,(BUFREG)   ;inv N(S)
C6DC CB47        0931        BIT   BY,A         ;test BUSY
C6DE 2805        0932        JR    Z,INF9
C6E0 CD05C7      0933        CALL  IBUSY        ;BUSY condition
C6E3 181D        0934        JR    ENDINF
C6E5 CB63        0935 INF9   BIT   4,E          ;not BUSY/test P bit
C6E7 2809        0936        JR    Z,INF10
C6E9 CBA8        0937        RES   SEC1,B       ;P=1/clear RNR sent
C6EB CBB0        0938        RES   SEC2,B       ;clear REJ sent
C6ED CDE8C9      0939        CALL  TRR          ;send RR (P=1)
C6F0 1810        0940        JR    ENDINF
C6F2 CB70        0941 INF10  BIT   SEC2,B       ;P=0/test REJ sent
C6F4 2805        0942        JR    Z,INF11
C6F6 CD5CC4      0943        CALL  MREL         ;REJ outs
C6F9 1807        0944        JR    ENDINF
C6FB CBF0        0945 INF11  SET   SEC2,B       ;send REJ (set REJ)
C6FD CBA8        0946        RES   SEC1,B       ;clear RNR sent
C6FF CD06CA      0947        CALL  TREJ
C702 C366C8      0948 ENDINF JP    LIQ1
C705 CB63        0949 IBUSY  BIT   4,E          ;test P bit
C707 2008        0950        JR    NZ,IBUSY1
C709 CB68        0951        BIT   SEC1,B       ;test RNR sent
C70B 2804        0952        JR    Z,IBUSY1
C70D CD5CC4      0953        CALL  MREL         ;P=0 and RNR sent
C710 C9          0954        RET
C711 CBE8        0955 IBUSY1 SET   SEC1,B       ;send RNR (set RNR)
C713 CDF7C9      0956        CALL  TRNR
C716 C9          0957        RET
                 0958 ;
C717 CD40C4      0959 UA     CALL  RELBUF       ;UA received
C71A AF          0960        XOR   A
```

```
C71B BA          0961        CP    D          ;test UA length
C71C 2807        0962        JR    Z,UA1
C71E 3E01        0963        LD    A,01       ;invalid length
C720 CD82C8      0964        CALL  RESET
C723 185E        0965        JR    ENDUA
C725 CB63        0966 UA1    BIT   4,E        ;test F bit
C727 2807        0967        JR    Z,UA2
C729 3E03        0968        LD    A,03       ;invalid F bit
C72B CD82C8      0969        CALL  RESET
C72E 1853        0970        JR    ENDUA
C730 CB41        0971 UA2    BIT   COM,C      ;test SARM/DISC outs
C732 2007        0972        JR    NZ,UA3
C734 3E04        0973        LD    A,04       ;no SARM/DISC (error)
C736 CD82C8      0974        CALL  RESET
C739 1848        0975        JR    ENDUA
C73B CB59        0976 UA3    BIT   WAIT,C     ;test SARM/DISC in LOQ
C73D 2044        0977        JR    NZ,ENDUA
C73F 2E0A        0978        LD    L,M10      ;valid UA
C741 2602        0979        LD    H,M02
C743 CDDDDB      0980        CALL  LDMES
C746 CD4CCC      0981        CALL  STP        ;update state variab
C749 AF          0982        XOR   A
C74A 3236F9      0983        LD    (RETCNT),A
C74D CB81        0984        RES   COM,C
C74F CBB1        0985        RES   RBY,C
C751 CB49        0986        BIT   SD,C       ;test SARM or DISC
C753 2021        0987        JR    NZ,UA5
C755 CBC8        0988        SET   PRIM,B     ;SARM outs
C757 CB50        0989        BIT   DIS,B      ;test discn
C759 2016        0990        JR    NZ,UA4
C75B 3232F9      0991        LD    (MAXNS),A  ;no discn/MAXNS=0
C75E 3233F9      0992        LD    (MINNS),A  ;MINNS=0
C761 3234F9      0993        LD    (LASTNR),A ;last recv N(R)=0
C764 3A35F9      0994        LD    A,(SEQNBR) ;V(S)=0
C767 E6E0        0995        AND   0E0H
C769 3235F9      0996        LD    (SEQNBR),A
C76C CDBCCB      0997        CALL  RETR2      ;reenable I transf
C76F 1812        0998        JR    ENDUA
C771 CDD2CA      0999 UA4    CALL  TDIS       ;init discn (delayed)
C774 180D        1000        JR    ENDUA
C776 CB98        1001 UA5    RES   SEC,B      ;DISC outs
C778 CB90        1002        RES   DIS,B
C77A CB88        1003        RES   PRIM,B
C77C CB80        1004        RES   LF,B
C77E CBD9        1005        SET   WAIT,C     ;delay next set-up
C780 CD54CC      1006        CALL  START
C783 C366C8      1007 ENDUA  JP    LIQ1
                 1008 ;
C786 CD40C4      1009 SUP    CALL  RELBUF     ;SUP received
C789 CB48        1010        BIT   PRIM,B     ;test prim state
C78B CA25C8      1011        JP    Z,ENDSUP
C78E AF          1012        XOR   A          ;prim connected
C78F BA          1013        CP    D          ;test SUP length
C790 2808        1014        JR    Z,SUP1
C792 3E01        1015        LD    A,01       ;invalid length
C794 CD82C8      1016        CALL  RESET
C797 C325C8      1017        JP    ENDSUP
C79A CB63        1018 SUP1   BIT   4,E        ;test F bit
C79C 280E        1019        JR    Z,SUP3
C79E CB69        1020        BIT   PS,C       ;test P outs
```

```
C7A0  2008      1021          JR    NZ,SUP2
C7A2  3E03      1022          LD    A,03          ;invalid F bit
C7A4  CD82C8    1023          CALL  RESET
C7A7  C325C8    1024          JP    ENDSUP
C7AA  CBA1      1025 SUP2     RES   REC,C         ;valid F=1/end recov
C7AC  CD00CB    1026 SUP3     CALL  NR            ;process N(R)
C7AF  CB41      1027          BIT   COM,C         ;test SARM/DISC outs
C7B1  C225C8    1028          JP    NZ,ENDSUP
C7B4  CD41CB    1029          CALL  ACK           ;accept SUP/ack
C7B7  CD32CB    1030          CALL  ENDREC        ;update end recov
C7BA  CB53      1031          BIT   2,E           ;decode SUP
C7BC  2025      1032          JR    NZ,RNR
C7BE  CB5B      1033          BIT   3,E
C7C0  2043      1034          JR    NZ,REJ
C7C2  2E43      1035 RR       LD    L,M67         ;RR received
C7C4  63        1036          LD    H,E
C7C5  CDDDDB    1037          CALL  LDMES
C7C8  DD214CFA  1038          LD    IX,RRA
C7CC  CDE3D6    1039          CALL  INCRCD
C7CF  CBB1      1040          RES   RBY,C         ;clear remote busy
C7D1  CB59      1041          BIT   WAIT,C        ;test window
C7D3  C225C8    1042          JP    NZ,ENDSUP
C7D6  3A6FF9    1043          LD    A,(NIF1)
C7D9  FE07      1044          CP    WINDOW
C7DB  CA25C8    1045          JP    Z,ENDSUP
C7DE  CBD1      1046          SET   IAL,C         ;(re)enable I transf
C7E0  C325C8    1047          JP    ENDSUP
C7E3  2E44      1048 RNR      LD    L,M68         ;RNR received
C7E5  63        1049          LD    H,E
C7E6  CDDDDB    1050          CALL  LDMES
C7E9  DD2150FA  1051          LD    IX,RNRA
C7ED  CDE3D6    1052          CALL  INCRCD
C7F0  CBF1      1053          SET   RBY,C         ;set remote busy
C7F2  AF        1054          XOR   A             ;reset retr counters
C7F3  3236F9    1055          LD    (RETCNT),A
C7F6  3237F9    1056          LD    (RICNT),A
C7F9  216FF9    1057          LD    HL,NIF1       ;test nbr INF outs
C7FC  BE        1058          CP    (HL)
C7FD  CA25C8    1059          JP    Z,ENDSUP
C800  CB91      1060          RES   IAL,C         ;>0/disable I transf
C802  C325C8    1061          JP    ENDSUP
C805  2E45      1062 REJ      LD    L,M69         ;REJ received
C807  63        1063          LD    H,E
C808  CDDDDB    1064          CALL  LDMES
C80B  DD2154FA  1065          LD    IX,REJA
C80F  CDE3D6    1066          CALL  INCRCD
C812  CBB1      1067          RES   RBY,C         ;clear remote busy
C814  CB59      1068          BIT   WAIT,C        ;prepare retransm
C816  C225C8    1069          JP    NZ,ENDSUP
C819  3A33F9    1070          LD    A,(MINNS)
C81C  3232F9    1071          LD    (MAXNS),A     ;MAXNS=MINNS
C81F  CD4CCC    1072          CALL  STP
C822  CD9BCB    1073          CALL  RETR
C825  C366C8    1074 ENDSUP   JP    LIQ1
                1075 ;
C828  3E03      1076 CMDR     LD    A,03          ;CMDR received
C82A  BA        1077          CP    D             ;test CMDR length
C82B  280A      1078          JR    Z,CMDR1
C82D  CD40C4    1079          CALL  RELBUF        ;invalid length
C830  3E01      1080          LD    A,01
```

```
C832  CD82C8      1081          CALL    RESET
C835  1821        1082          JR      CMDR2
C837  DD2A56F9    1083 CMDR1    LD      IX,(CURBUF)     ;valid CMDR
C83B  2E40        1084          LD      L,M64
C83D  DD6603      1085          LD      H,(IX+3)        ;read rej contr
C840  CDDDDB      1086          CALL    LDMES
C843  2E61        1087          LD      L,M97
C845  DD6605      1088          LD      H,(IX+5)        ;read rej reason
C848  CDDDDB      1089          CALL    LDMES
C84B  CD40C4      1090          CALL    RELBUF
C84E  DD2158FA    1091          LD      IX,CMRA
C852  CDE3D6      1092          CALL    INCRCD
C855  CD95C8      1093          CALL    RESET1
C858  C366C8      1094 CMDR2    JP      LIQ1
                  1095 ;
C85B  CD40C4      1096 INVRES   CALL    RELBUF          ;invalid response
C85E  3E00        1097          LD      A,00
C860  CD82C8      1098          CALL    RESET
C863  C366C8      1099          JP      LIQ1
                  1100 ;
C866  CB68        1101 LIQ1     BIT     SEC1,B          ;test RNR sent
C868  2817        1102          JR      Z,ENDLIQ
C86A  CB70        1103          BIT     SEC2,B          ;test REJ sent
C86C  2013        1104          JR      NZ,ENDLIQ
C86E  3A40F9      1105          LD      A,(BUFREG)      ;RNR/no REJ sent
C871  CB47        1106          BIT     BY,A            ;test BUSY
C873  200C        1107          JR      NZ,ENDLIQ
C875  CD1DC4      1108          CALL    GTPOOL          ;clear BUSY
C878  CBF0        1109          SET     SEC2,B          ;send REJ (set REJ)
C87A  CBA8        1110          RES     SEC1,B          ;clear RNR sent
C87C  1E00        1111          LD      E,00
C87E  CD06CA      1112          CALL    TREJ            ;send REJ
C881  C9          1113 ENDLIQ   RET
C882  2E4E        1114 RESET    LD      L,M78           ;recov prim error
C884  63          1115          LD      H,E
C885  CDDDDB      1116          CALL    LDMES
C888  2E61        1117          LD      L,M97
C88A  67          1118          LD      H,A
C88B  CDDDDB      1119          CALL    LDMES
C88E  DD216CFA    1120          LD      IX,IVR
C892  CDE3D6      1121          CALL    INCRCD
C895  CB48        1122 RESET1   BIT     PRIM,B          ;test prim state
C897  C8          1123          RET     Z
C898  CDB5CA      1124          CALL    SSAR            ;init link reset
C89B  C9          1125          RET
                  1126 ;
                  1127 ;        ************
                  1128 ;        * TIME-OUT *
                  1129 ;        ************
                  1130 ;
C89C  2150F9      1131 TOUT     LD      HL,INTREG       ;time-out detection
C89F  CB46        1132          BIT     T2OUT,(HL)
C8A1  C8          1133          RET     Z
C8A2  CB8E        1134          RES     T2ON,(HL)       ;time-out/stop tim
C8A4  CB86        1135          RES     T2OUT,(HL)      ;clear time-out
C8A6  DD216EFA    1136          LD      IX,TM2
C8AA  CDE3D6      1137          CALL    INCRCD
C8AD  3A36F9      1138          LD      A,(RETCNT)
C8B0  FE03        1139          CP      RETMAX          ;test retr count
C8B2  285D        1140          JR      Z,TOUT5
```

```
C8B4  3C        1141          INC   A              ;not max/inc count
C8B5  3236F9    1142          LD    (RETCNT),A
C8B8  CB59      1143          BIT   WAIT,C         ;test state
C8BA  281C      1144          JR    Z,TOUT1
C8BC  CB48      1145          BIT   PRIM,B         ;WAIT/test prim
C8BE  200C      1146          JR    NZ,TOUT0
C8C0  CB41      1147          BIT   COM,C          ;WAIT/prim discn
C8C2  2008      1148          JR    NZ,TOUT0
C8C4  CB99      1149          RES   WAIT,C         ;clear set-up delay
C8C6  AF        1150          XOR   A
C8C7  3236F9    1151          LD    (RETCNT),A
C8CA  186F      1152          JR    TOUT8
C8CC  CD54CC    1153 TOUT0    CALL  START          ;WAIT/rest timer
C8CF  2E12      1154          LD    L,M18
C8D1  2602      1155          LD    H,M02
C8D3  CDDDDB    1156          CALL  LDMES
C8D6  1863      1157          JR    TOUT8
C8D8  CB41      1158 TOUT1    BIT   COM,C          ;test frame type
C8DA  2010      1159          JR    NZ,TOUT2
C8DC  2E15      1160          LD    L,M21          ;INF outs
C8DE  2602      1161          LD    H,M02
C8E0  CDDDDB    1162          CALL  LDMES
C8E3  CBE1      1163          SET   REC,C          ;retransm (recov)
C8E5  CBA9      1164          RES   PS,C
C8E7  CD9BCB    1165          CALL  RETR
C8EA  184F      1166          JR    TOUT8
C8EC  CB49      1167 TOUT2    BIT   SD,C           ;test SARM/DISC outs
C8EE  280C      1168          JR    Z,TOUT3
C8F0  2E14      1169          LD    L,M20          ;DISC outs
C8F2  2602      1170          LD    H,M02
C8F4  CDDDDB    1171          CALL  LDMES
C8F7  CDD2CA    1172          CALL  TDIS           ;retransm DISC
C8FA  183F      1173          JR    TOUT8
C8FC  CB50      1174 TOUT3    BIT   DIS,B          ;SARM outs/test discn
C8FE  200C      1175          JR    NZ,TOUT4
C900  2E13      1176          LD    L,M19          ;no discn/retr SARM
C902  2602      1177          LD    H,M02
C904  CDDDDB    1178          CALL  LDMES
C907  CDBCCA    1179          CALL  TSAR
C90A  182F      1180          JR    TOUT8
C90C  CDCFCA    1181 TOUT4    CALL  INITD          ;init discn
C90F  182A      1182          JR    TOUT8
C911  CB41      1183 TOUT5    BIT   COM,C          ;max retr/frame type
C913  280C      1184          JR    Z,TOUT6
C915  2E17      1185          LD    L,M23          ;not INF outs
C917  2602      1186          LD    H,M02
C919  CDDDDB    1187          CALL  LDMES
C91C  CDD1CB    1188          CALL  LFAIL          ;link failure
C91F  181A      1189          JR    TOUT8
C921  2E16      1190 TOUT6    LD    L,M22          ;INF failure
C923  2602      1191          LD    H,M02
C925  CDDDDB    1192          CALL  LDMES
C928  3E03      1193          LD    A,RIMAX
C92A  2137F9    1194          LD    HL,RICNT       ;test INF recov count
C92D  BE        1195          CP    (HL)
C92E  2806      1196          JR    Z,TOUT7
C930  34        1197          INC   (HL)           ;reset link
C931  CDB9CA    1198          CALL  INITS
C934  1805      1199          JR    TOUT8
C936  3600      1200 TOUT7    LD    (HL),00        ;INF recov fail
```

```
C938  CDF7CA      1201          CALL    DISCNC          ;discn link
C93B  C9          1202 TOUT8    RET
                  1203 ;
                  1204 ;        ************************
                  1205 ;        * INFORMATION MANAGER *
                  1206 ;        ************************
                  1207 ;
C93C  2172F9      1208 IMNG     LD      HL,NIFC         ;INF frames to process
C93F  AF          1209          XOR     A
C940  BE          1210          CP      (HL)
C941  286F        1211          JR      Z,IMNG2
C943  CB51        1212          BIT     IAL,C           ;test I transf condit
C945  2016        1213          JR      NZ,TINF
C947  CB48        1214          BIT     PRIM,B          ;I disabled/test prim
C949  2067        1215          JR      NZ,IMNG2
C94B  CB40        1216          BIT     LF,B            ;test failure
C94D  2009        1217          JR      NZ,IMNG1
C94F  CB59        1218          BIT     WAIT,C          ;test set-up delay
C951  205F        1219          JR      NZ,IMNG2
C953  CDB5CA      1220          CALL    SSAR            ;link set-up
C956  185A        1221          JR      IMNG2
C958  CDCBCA      1222 IMNG1    CALL    SDIS            ;discn before set-up
C95B  1855        1223          JR      IMNG2
C95D  35          1224 TINF     DEC     (HL)            ;I transf enabled
C95E  2174F9      1225          LD      HL,DTFQ5        ;get I frame (POQ)
C961  7E          1226          LD      A,(HL)
C962  34          1227          INC     (HL)
C963  2180F8      1228          LD      HL,TPQ5
C966  CD01C4      1229          CALL    GETBUF
C969  23          1230          INC     HL
C96A  3603        1231          LD      (HL),TCOMAD     ;format header (addr)
C96C  23          1232          INC     HL
C96D  3A35F9      1233          LD      A,(SEQNBR)      ;get V(R)/V(S)
C970  CB61        1234          BIT     REC,C           ;test recov
C972  2804        1235          JR      Z,TINF1
C974  F610        1236          OR      10H             ;set P bit (recov)
C976  CBD9        1237          SET     WAIT,C
C978  77          1238 TINF1    LD      (HL),A          ;insert contr field
C979  2E46        1239          LD      L,M70
C97B  67          1240          LD      H,A
C97C  CDDDDB      1241          CALL    LDMES
C97F  DD2146FA    1242          LD      IX,INFB
C983  CDE3D6      1243          CALL    INCRCD
C986  E6EE        1244          AND     0EEH            ;inc V(S)
C988  C602        1245          ADD     02H
C98A  E6EE        1246          AND     0EEH
C98C  3235F9      1247          LD      (SEQNBR),A      ;save V(S)
C98F  CD91CA      1248          CALL    TRANSF          ;transf to LOQ
C992  2173F9      1249          LD      HL,NIF2         ;incr INF in LOQ
C995  34          1250          INC     (HL)
C996  CD54CC      1251          CALL    START
C999  CBB9        1252          RES     PGB,C           ;piggyback done
C99B  CB91        1253          RES     IAL,C           ;disable I transf
C99D  216FF9      1254          LD      HL,NIF1         ;inc unack I frames
C9A0  34          1255          INC     (HL)
C9A1  3E07        1256          LD      A,WINDOW
C9A3  BE          1257          CP      (HL)            ;test window
C9A4  280C        1258          JR      Z,IMNC2
C9A6  CB61        1259          BIT     REC,C           ;test recovery
C9A8  2008        1260          JR      NZ,IMNG2
```

```
C9AA  CB71        1261          BIT   RBY,C           ;test remote busy
C9AC  2004        1262          JR    NZ,IMNG2
C9AE  CBD1        1263          SET   IAL,C           ;reenable I transf
C9B0  180E        1264          JR    IMNG3
C9B2  CB79        1265 IMNG2    BIT   PGB,C           ;test piggyback
C9B4  280A        1266          JR    Z,IMNG3
C9B6  CBB9        1267          RES   PGB,C           ;pigg impossible
C9B8  CD1DC4      1268          CALL  GTPOOL          ;send RR/get buff
C9BB  1E00        1269          LD    E,00
C9BD  CDE8C9      1270          CALL  TRR
C9C0  C9          1271 IMNG3    RET
                  1272 ;
C9C1  ED4330F9    1273 LINK2    LD    (LSTATE),BC     ;save link state
C9C5  C9          1274          RET
                  1275 ;
                  1276 ;        ************************
                  1277 ;        * LINK LEVEL ROUTINES *
                  1278 ;        ************************
                  1279 ;
C9C6  2E0D        1280 TUA      LD    L,M13           ;transm UA
C9C8  2602        1281          LD    H,M02
C9CA  CDDDDB      1282          CALL  LDMES
C9CD  CBA0        1283          RES   SEC0,B          ;reset sec substate
C9CF  CBA8        1284          RES   SEC1,B
C9D1  CBB0        1285          RES   SEC2,B
C9D3  2A56F9      1286          LD    HL,(CURBUF)     ;get buffer
C9D6  3603        1287          LD    (HL),FLMIN      ;insert length
C9D8  23          1288          INC   HL
C9D9  3601        1289          LD    (HL),TRESAD     ;insert addr field
C9DB  23          1290          INC   HL
C9DC  3663        1291          LD    (HL),63H        ;insert contr field
C9DE  CB63        1292          BIT   4,E             ;test recv P bit
C9E0  2802        1293          JR    Z,TUA1
C9E2  3673        1294          LD    (HL),73H        ;set F=1
C9E4  CD91CA      1295 TUA1     CALL  TRANSF          ;transf to LOQ
C9E7  C9          1296          RET
C9E8  DD214EFA    1297 TRR      LD    IX,RRB          ;transm RR
C9EC  CDE3D6      1298          CALL  INCRCD
C9EF  1601        1299          LD    D,01            ;RR code
C9F1  2E47        1300          LD    L,M71
C9F3  CD15CA      1301          CALL  TSUP            ;format SUP
C9F6  C9          1302          RET
C9F7  DD2152FA    1303 TRNR     LD    IX,RNRB         ;transm RNR
C9FB  CDE3D6      1304          CALL  INCRCD
C9FE  1605        1305          LD    D,05            ;RNR code
CA00  2E48        1306          LD    L,M72
CA02  CD15CA      1307          CALL  TSUP            ;format SUP
CA05  C9          1308          RET
CA06  DD2156FA    1309 TREJ     LD    IX,REJB         ;transm REJ
CA0A  CDE3D6      1310          CALL  INCRCD
CA0D  1609        1311          LD    D,09            ;REJ code
CA0F  2E49        1312          LD    L,M73
CA11  CD15CA      1313          CALL  TSUP
CA14  C9          1314          RET
CA15  3A35F9      1315 TSUP     LD    A,(SEQNBR)      ;form contr/get V(R)
CA18  E6E0        1316          AND   0E0H
CA1A  B2          1317          OR    D               ;RR/RNR/REJ code
CA1B  CB63        1318          BIT   4,E             ;test recv P bit
CA1D  2802        1319          JR    Z,TSUP1
CA1F  F610        1320          OR    10H             ;set F=1
```

```
CA21  67          1321 TSUP1   LD    H,A             ;control field
CA22  CDDDDB      1322         CALL  LDMES
CA25  2A56F9      1323         LD    HL,(CURBUF)     ;get buffer
CA28  3603        1324         LD    (HL),FLMIN      ;insert length
CA2A  23          1325         INC   HL
CA2B  3601        1326         LD    (HL),TRESAD     ;insert addr field
CA2D  23          1327         INC   HL
CA2E  77          1328         LD    (HL),A          ;insert contr field
CA2F  CD91CA      1329         CALL  TRANSF          ;transf to LOQ
CA32  C9          1330         RET
CA33  3A39F9      1331 TCMDR   LD    A,(REJCTR)      ;transm CMDR
CA36  2E41        1332         LD    L,M65
CA38  67          1333         LD    H,A
CA39  CDDDDB      1334         CALL  LDMES
CA3C  CBEC        1335         SET   SEC0,B          ;set CMDR sent
CA3E  CBA8        1336         RES   SEC1,B
CA40  CBB0        1337         RES   SEC2,B
CA42  2A56F9      1338         LD    HL,(CURBUF)     ;get buffer
CA45  3606        1339         LD    (HL),FLMIN+3    ;insert length
CA47  23          1340         INC   HL
CA48  3601        1341         LD    (HL),TRESAD     ;insert addr field
CA4A  23          1342         INC   HL
CA4B  3687        1343         LD    (HL),87H        ;insert contr field
CA4D  CB63        1344         BIT   4,E             ;test recv P bit
CA4F  2802        1345         JR    Z,TCMDR1
CA51  3697        1346         LD    (HL),97H        ;set F=1
CA53  23          1347 TCMDR1  INC   HL
CA54  77          1348         LD    (HL),A          ;insert rej contr
CA55  23          1349         INC   HL
CA56  3A35F9      1350         LD    A,(SEQNBR)
CA59  E6E0        1351         AND   0E0H
CA5B  77          1352         LD    (HL),A          ;insert V(R)
CA5C  23          1353         INC   HL
CA5D  3A38F9      1354         LD    A,(REJRSN)
CA60  77          1355         LD    (HL),A          ;insert rej reason
CA61  2E61        1356         LD    L,M97
CA63  67          1357         LD    H,A
CA64  CDDDDB      1358         CALL  LDMES
CA67  DD215AFA    1359         LD    IX,CMRB
CA6B  CDE3D6      1360         CALL  INCRCD
CA6E  CD91CA      1361         CALL  TRANSF          ;transf to LOQ
CA71  C9          1362         RET
CA72  CBC1        1363 TSD     SET   COM,C           ;transm SARM/DISC
CA74  CBD9        1364         SET   WAIT,C
CA76  21F8F7      1365         LD    HL,SDBUF        ;get buffer
CA79  2256F9      1366         LD    (CURBUF),HL
CA7C  3603        1367         LD    (HL),FLMIN      ;insert length
CA7E  23          1368         INC   HL
CA7F  3603        1369         LD    (HL),TCOMAD     ;insert addr field
CA81  23          1370         INC   HL
CA82  72          1371         LD    (HL),D          ;insert contr field
CA83  DD214AFA    1372         LD    IX,COMB
CA87  CDE3D6      1373         CALL  INCRCD
CA8A  CD91CA      1374         CALL  TRANSF          ;transf to LOQ
CA8D  CD54CC      1375         CALL  START
CA90  C9          1376         RET
                  1377 ;
CA91  DD2142FA    1378 TRANSF  LD    IX,FRMB         ;transf frame to LOQ
CA95  CDE3D6      1379         CALL  INCRCD
CA98  2168F9      1380         LD    HL,DLQ6
```

```
CA9B 7E          1381          LD      A,(HL)
CA9C 34          1382          INC     (HL)
CA9D 21A0F8      1383          LD      HL,TPQ6
CAA0 CD0EC4      1384          CALL    PUTBUF
CAA3 2169F9      1385          LD      HL,NIF3         ;inc LOQ counters
CAA6 34          1386          INC     (HL)
CAA7 216AF9      1387          LD      HL,NIF4
CAAA F3          1388          DI
CAAB 34          1389          INC     (HL)
CAAC 2166F9      1390          LD      HL,NIF5
CAAF 34          1391          INC     (HL)
CAB0 FB          1392          EI
CAB1 CD63CC      1393          CALL    TRANSM          ;init transmission
CAB4 C9          1394          RET
                 1395 ;
CAB5 CB41        1396 SSAR     BIT     COM,C           ;snd SARM/test com outs
CAB7 2011        1397          JR      NZ,SSAR0
CAB9 CDE1CA      1398 INITS    CALL    INITSD          ;first SARM
CABC 16CF        1399 TSAR     LD      D,0FH           ;SARM code
CABE CB89        1400          RES     SD,C            ;SARM outs
CAC0 2E0B        1401          LD      L,M11
CAC2 2602        1402          LD      H,M02
CAC4 CDDDDB      1403          CALL    LDMES
CAC7 CD72CA      1404          CALL    TSD             ;transm SARM
CACA C9          1405 SSAR0    RET
CACB CB41        1406 SDIS     BIT     COM,C           ;snd DISC/test com outs
CACD 2011        1407          JR      NZ,SDIS0
CACF CDE1CA      1408 INITD    CALL    INITSD          ;first DISC
CAD2 1643        1409 TDIS     LD      D,43H           ;DISC code
CAD4 CBC9        1410          SET     SD,C            ;DISC outs
CAD6 2E0C        1411          LD      L,M12
CAD8 2602        1412          LD      H,M02
CADA CDDDDB      1413          CALL    LDMES
CADD CD72CA      1414          CALL    TSD             ;transm DISC
CAE0 C9          1415 SDIS0    RET
CAE1 CD4CCC      1416 INITSD   CALL    STP             ;first SARM/DISC
CAE4 AF          1417          XOR     A
CAE5 3236F9      1418          LD      (RETCNT),A      ;reset ret count
CAE8 CBA1        1419          RES     REC,C           ;clear INF recov
CAEA CBA9        1420          RES     PS,C
CAEC CB91        1421          RES     IAL,C           ;disable INF transf
CAEE C9          1422          RET
CAEF CB50        1423 DISCN    BIT     DIS,B           ;test discn
CAF1 C8          1424          RET     Z
CAF2 2147F9      1425          LD      HL,IRLP         ;signal discn (lev3)
CAF5 CBCE        1426          SET     LD,(HL)
CAF7 CBD0        1427 DISCN0   SET     DIS,B           ;set discn
CAF9 CDCBCA      1428          CALL    SDIS            ;send DISC
CAFC CD7DCB      1429          CALL    CLINF           ;clear I frames (POQ)
CAFF C9          1430          RET
                 1431 ;
CB00 CB41        1432 NR       BIT     COM,C           ;process N(R)
CB02 C0          1433          RET     NZ
CB03 7B          1434          LD      A,E             ;get N(R)
CB04 07          1435          RLCA
CB05 07          1436          RLCA
CB06 07          1437          RLCA
CB07 2134F9      1438          LD      HL,LASTNR       ;get last recv N(R)
CB0A 66          1439          LD      H,(HL)
CB0B 94          1440          SUB     H
```

```
CB0C  E607          1441          AND     07H             ;compare
CB0E  C8            1442          RET     Z
CB0F  6F            1443          LD      L,A             ;nbr ack frames
CB10  84            1444          ADD     H
CB11  3234F9        1445          LD      (LASTNR),A
CB14  3A32F9        1446          LD      A,(MAXNS)       ;get MAXNS
CB17  94            1447          SUB     H               ;nbr outs frames
CB18  E607          1448          AND     07H
CB1A  BD            1449          CP      L               ;compare ack/outs
CB1B  D0            1450          RET     NC
CB1C  3E02          1451          LD      A,02            ;invalid N(R)
CB1E  CD82C8        1452          CALL    RESET
CB21  C9            1453          RET
CB22  CB48          1454 NRI      BIT     PRIM,B          ;proc N(R) (INF)
CB24  C8            1455          RET     Z
CB25  CD00CB        1456          CALL    NR
CB28  CB41          1457          BIT     COM,C           ;test SARM/DISC outs
CB2A  C0            1458          RET     NZ
CB2B  CB71          1459          BIT     RBY,C           ;test remote busy
CB2D  C0            1460          RET     NZ
CB2E  CD41CB        1461          CALL    ACK             ;accept N(R) ack
CB31  C9            1462          RET
CB32  CB63          1463 ENDREC   BIT     4,E             ;update end recov
CB34  C8            1464          RET     Z               ;not end recov
CB35  CBA9          1465          RES     PS,C            ;end recov/clear PS
CB37  3A33F9        1466          LD      A,(MINNS)
CB3A  3232F9        1467          LD      (MAXNS),A       ;MAXNS=MINNS
CB3D  CDAACB        1468          CALL    RETR1           ;prepare next I transm
CB40  C9            1469          RET
CB41  CB61          1470 ACK      BIT     REC,C           ;release ack I frames
CB43  C0            1471          RET     NZ              ;no release (recov)
CB44  3A34F9        1472          LD      A,(LASTNR)      ;get nbr to release
CB47  2133F9        1473          LD      HL,MINNS
CB4A  96            1474          SUB     (HL)
CB4B  E607          1475          AND     07H
CB4D  C8            1476          RET     Z               ;nbr=0
CB4E  CBD1          1477          SET     IAL,C           ;>0/(re)enable I transf
CB50  C5            1478          PUSH    BC
CB51  47            1479          LD      B,A
CB52  86            1480          ADD     (HL)
CB53  77            1481          LD      (HL),A          ;MINNS=LASTNR
CB54  AF            1482          XOR     A               ;reset retr counts
CB55  3236F9        1483          LD      (RETCNT),A
CB58  3237F9        1484          LD      (RICNT),A
CB5B  CD4CCC        1485          CALL    STP
CB5E  216FF9        1486          LD      HL,NIF1         ;update unack count
CB61  7E            1487          LD      A,(HL)
CB62  90            1488          SUB     B
CB63  77            1489          LD      (HL),A
CB64  C454CC        1490          CALL    NZ,START        ;restart timer
CB67  CD6CCB        1491          CALL    RELINF          ;release ack I frames
CB6A  C1            1492          POP     BC
CB6B  C9            1493 ENDACK   RET
                    1494 ;
CB6C  2170F9        1495 RELINF   LD      HL,DFQ5         ;release I frames (POQ)
CB6F  7E            1496          LD      A,(HL)
CB70  34            1497          INC     (HL)
CB71  2180F8        1498          LD      HL,TPQ5
CB74  CD01C4        1499          CALL    GETBUF
CB77  CD40C4        1500          CALL    RELBUF
```

```
CB7A 10FC       1501        DJNZ   RELINF      ;repeat all INF (POQ)
CB7C C9         1502        RET
CB7D 3A6FF9     1503 CLINF  LD     A,(NIF1)    ;clear POQ
CB80 2172F9     1504        LD     HL,NIF0
CB83 86         1505        ADD    (HL)
CB84 3600       1506        LD     (HL),00
CB86 326FF9     1507        LD     (NIF1),A    ;nbr I frames in POQ
CB89 C8         1508        RET    Z
CB8A C5         1509        PUSH   BC
CB8B 47         1510        LD     B,A
CB8C CD6CCB     1511        CALL   RELINF
CB8F C1         1512        POP    BC
CB90 AF         1513        XOR    A
CB91 326FF9     1514        LD     (NIF1),A    ;POQ cleared (count)
CB94 3A70F9     1515        LD     A,(DFQ5)    ;update POQ pointers
CB97 3274F9     1516        LD     (DTFQ5),A
CB9A C9         1517        RET
                1518 ;
CB9B 3A73F9     1519 RETR   LD     A,(NIF2)    ;prepare INF retransm
CB9E B7         1520        OR     A           ;test nbr INF in LOQ
CB9F 2809       1521        JR     Z,RETR1
CBA1 CBD9       1522        SET    WAIT,C      ;nbr>0
CBA3 CB91       1523        RES    IAL,C       ;disable I frame transf
CBA5 CD54CC     1524        CALL   START
CBA8 1826       1525        JR     RETR2
CBAA CB99       1526 RETR1  RES    WAIT,C      ;nbr=0
CBAC 3A33F9     1527        LD     A,(MINNS)   ;V(S)=MINNS
CBAF E607       1528        AND    07H
CBB1 07         1529        RLCA
CBB2 6F         1530        LD     L,A
CBB3 3A35F9     1531        LD     A,(SEQNBR)
CBB6 E6E0       1532        AND    0E0H
CBB8 B5         1533        OR     L
CBB9 3235F9     1534        LD     (SEQNBR),A
CBBC CBD1       1535 RETR2  SET    IAL,C       ;enable I frame transf
CBBE 3A70F9     1536        LD     A,(DFQ5)    ;update POQ pointers
CBC1 3274F9     1537        LD     (DTFQ5),A
CBC4 3A72F9     1538        LD     A,(NIF0)    ;update POQ counters
CBC7 216FF9     1539        LD     HL,NIF1
CBCA 86         1540        ADD    (HL)        ;NIF0=NIF1+NIF0
CBCB 3600       1541        LD     (HL),00     ;NIF1=0
CBCD 3272F9     1542        LD     (NIF0),A
CBD0 C9         1543 RETR3  RET
                1544 ;
CBD1 CD43CC     1545 LFAIL  CALL   DISIO       ;link failure/reset SIO
CBD4 2147F9     1546        LD     HL,IRLP
CBD7 CBCE       1547        SET    LD,(HL)     ;signal fail (lev3)
CBD9 2160F9     1548        LD     HL,CTQ1     ;clear LIQ
CBDC 7E         1549        LD     A,(HL)
CBDD 3600       1550        LD     (HL),00
CBDF 2120F9     1551        LD     HL,RCREGA   ;test current reception
CBE2 CB46       1552        BIT    RC,(HL)
CBE4 2801       1553        JR     Z,LFAIL0
CBE6 3C         1554        INC    A           ;reception (one buf)
CBE7 2140F9     1555 LFAIL0 LD     HL,BUFREG   ;return buffers (pool)
CBEA CB46       1556        BIT    BY,(HL)     ;test Busy
CBEC 2163F9     1557        LD     HL,NOFB
CBEF 2803       1558        JR     Z,LFAIL1
CBF1 2164F9     1559        LD     HL,NOGB
CBF4 86         1560 LFAIL1 ADD    (HL)
```

```
CBF5  77        1561          LD      (HL),A
CBF6  3A61F9    1562          LD      A,(DFQ1)        ;update LIQ pointers
CBF9  3262F9    1563          LD      (DLQ1),A
CBFC  AF        1564          XOR     A               ;clear LOQ
CBFD  326AF9    1565          LD      (NIF4),A
CC00  3266F9    1566          LD      (NIF5),A
CC03  3A68F9    1567          LD      A,(DLQ6)        ;update LOQ pointers
CC06  326BF9    1568          LD      (DTMQ6),A
CC09  3A69F9    1569          LD      A,(NIF3)        ;reproc frames in LOQ
CC0C  B7        1570          OR      A
CC0D  2817      1571          JR      Z,LFAIL4
CC0F  47        1572 LFAIL2   LD      B,A
CC10  2167F9    1573          LD      HL,DFQ6         ;get frame
CC13  7E        1574          LD      A,(HL)
CC14  34        1575          INC     (HL)
CC15  21A0F8    1576          LD      HL,TPQ6
CC18  CD01C4    1577          CALL    GETBUF
CC1B  23        1578          INC     HL
CC1C  7E        1579          LD      A,(HL)          ;read addr field
CC1D  FE01      1580          CP      TRESAD
CC1F  2003      1581          JR      NZ,LFAIL3
CC21  CD40C4    1582          CALL    RELBUF          ;response/release
CC24  10E9      1583 LFAIL3   DJNZ    LFAIL2          ;repeat until LOQ clear
CC26  AF        1584 LFAIL4   XOR     A               ;LOQ cleared
CC27  3269F9    1585          LD      (NIF3),A
CC2A  3273F9    1586          LD      (NIF2),A
CC2D  CD7DCB    1587          CALL    CLINF           ;clear POQ
CC30  F3        1588          DI
CC31  CD54C1    1589          CALL    SINIT           ;reinit SIO
CC34  FB        1590          EI
CC35  AF        1591          XOR     A
CC36  2132F9    1592          LD      HL,MAXNS        ;reinit link variab
CC39  0606      1593          LD      B,06
CC3B  77        1594 LFAIL5   LD      (HL),A
CC3C  23        1595          INC     HL
CC3D  10FC      1596          DJNZ    LFAIL5
CC3F  010001    1597          LD      BC,INLST        ;reinit link state
CC42  C9        1598          RET
                1599 ;
CC43  F3        1600 DISIO    DI                      ;reset SIO
CC44  3E18      1601          LD      A,18H
CC46  D37D      1602          OUT     (SIOCA),A
CC48  D37F      1603          OUT     (SIOCB),A
CC4A  FB        1604          EI
CC4B  C9        1605          RET
CC4C  2150F9    1606 STP      LD      HL,INTREG       ;stop timer
CC4F  CB8E      1607          RES     T2ON,(HL)
CC51  CB86      1608          RES     T2OUT,(HL)      ;clear time-out
CC53  C9        1609          RET
CC54  2150F9    1610 START    LD      HL,INTREG       ;start timer
CC57  CB4E      1611          BIT     T2ON,(HL)       ;test timer ON
CC59  2007      1612          JR      NZ,START0
CC5B  3E0A      1613          LD      A,TIME2         ;tim OFF/load count
CC5D  3251F9    1614          LD      (TIMRG2),A
CC60  CBCE      1615          SET     T2ON,(HL)       ;timer started
CC62  C9        1616 START0   RET
CC63  2121F9    1617 TRANSM   LD      HL,TRREGB       ;frame transmission
CC66  CB46      1618          BIT     T,(HL)          ;test current transm
CC68  C0        1619          RET     NZ
CC69  F3        1620          DI                      ;no transm
```

```
CC6A  3A66F9      1621          LD      A,(NIF5)        ;nbr fr to transm
CC6D  D9          1622          EXX
CC6E  CD20C3      1623          CALL    TX              ;init transm
CC71  D9          1624          EXX
CC72  FB          1625          EI
CC73  C9          1626          RET
                  1627 ;
                  1628 ;
                  1629 ;        ************************
                  1630 ;        * PACKET LEVEL PROCESS *
                  1631 ;        ************************
                  1632 ;
                  1633 ;
                  1634 ;register C is generally used to store, test and
                  1635 ;modify the packet level state
                  1636 ;register E is used to store the current logical
                  1637 ;channel across the X.25 interface
                  1638 ;
                  1639 ;
                  1640 ;        ********************************
                  1641 ;        * PACKET LEVEL MAIN PROCEDURES *
                  1642 ;        ********************************
                  1643 ;
CC74  CD8ACC      1644 PACK     CALL    PACK1           ;pack lev scheduler
CC77  CD8FCC      1645          CALL    INTPL
CC7A  CDA8CC      1646          CALL    PIQ
CC7D  CD9AD0      1647          CALL    BIQ
CC80  CD84D3      1648          CALL    TIM
CC83  CDC1D4      1649          CALL    SCAN
CC86  CD8ED5      1650          CALL    PACK2
CC89  C9          1651          RET
                  1652 ;
CC8A  2141F9      1653 PACK1    LD      HL,GENST        ;read pack state
CC8D  4E          1654          LD      C,(HL)
CC8E  C9          1655          RET
                  1656 ;
                  1657 ;        ********************************
                  1658 ;        * INTERFACE WITH THE LINK LEVEL *
                  1659 ;        ********************************
                  1660 ;
CC8F  2147F9      1661 INTPL    LD      HL,IRLP         ;read 12/13 state
CC92  CB4E        1662          BIT     LD,(HL)         ;test lev2 signal
CC94  2811        1663          JR      Z,INTPL0
CC96  CB8E        1664          RES     LD,(HL)         ;remote call clear
CC98  CD1FD7      1665          CALL    CLPAK
CC9B  CB89        1666          RES     DCERST,C
CC9D  CB81        1667          RES     TONRST,C
CC9F  2149F9      1668          LD      HL,CLRST        ;clear reason
CCA2  3609        1669          LD      (HL),09
CCA4  CD3CD7      1670          CALL    RST
CCA7  C9          1671 INTPL0   RET
                  1672 ;
                  1673 ;        ***********************************************
                  1674 ;        * PROCESS PACKET RECEIVED FROM THE DTE (PIQ) *
                  1675 ;        ***********************************************
                  1676 ;
CCA8  216CF9      1677 PIQ      LD      HL,CTQ2         ;read nbr pack (PIQ)
CCAB  AF          1678          XOR     A
CCAC  BE          1679          CP      (HL)
CCAD  CA99D0      1680          JP      Z,ENDPIQ
```

```
CCB0  35        1681        DEC   (HL)        ;dec PIQ counter
CCB1  DD215CFA  1682        LD    IX,PAKA
CCB5  CDE3D6    1683        CALL  INCRCD
CCB8  216DF9    1684        LD    HL,DFQ2     ;get pack
CCBB  7E        1685        LD    A,(HL)
CCBC  34        1686        INC   (HL)
CCBD  2120F8    1687        LD    HL,TPQ2
CCC0  CD01C4    1688        CALL  GETBUF
CCC3  7E        1689 PCTR   LD    A,(HL)      ;read length
CCC4  23        1690        INC   HL
CCC5  23        1691        INC   HL
CCC6  23        1692        INC   HL          ;point ID/chan group
CCC7  46        1693        LD    B,(HL)      ;read ID field
CCC8  23        1694        INC   HL          ;point chan number
CCC9  5E        1695        LD    E,(HL)      ;read chan number
CCCA  D606      1696 SPRJ   SUB   PLMIN       ;test short packet
CCCC  DA87D0    1697        JP    C,REJPAK
CCCF  57        1698        LD    D,A         ;nbr Data bytes
CCD0  78        1699 INVCH  LD    A,B         ;validate group nbr
CCD1  E60F      1700        AND   0FH         ;get ch group nbr
CCD3  C28AD0    1701        JP    NZ,REJPK1
CCD6  3E07      1702        LD    A,MXCH      ;validate ch range
CCD8  93        1703        SUB   E
CCD9  DA8AD0    1704        JP    C,REJPK1
CCDC  23        1705        INC   HL          ;point pack type
CCDD  AF        1706        XOR   A
CCDE  BB        1707        CP    E           ;test for chan=0
CCDF  78        1708        LD    A,B
CCE0  CA5ECD    1709        JP    Z,DCPK6
CCE3  CB46      1710 DCPK   BIT   0,(HL)      ;ch>0/test Data
CCE5  200C      1711        JR    NZ,DCPK1
CCE7  0640      1712        LD    B,40H       ;Data (int code)
CCE9  E630      1713        AND   IDMASK      ;get ID
CCEB  FE10      1714        CP    10H         ;validate ID
CCED  CAF2CD    1715        JP    Z,DCST
CCF0  C359CD    1716        JP    INVPAK
CCF3  E6F0      1717 DCPK1  AND   0F0H        ;non Data/get ID
CCF5  FE10      1718        CP    10H         ;validate ID
CCF7  C259CD    1719        JP    NZ,INVPAK
CCFA  7E        1720        LD    A,(HL)      ;read pack type
CCFB  15        1721        DEC   D           ;validate length
CCFC  15        1722        DEC   D
CCFD  2817      1723        JR    Z,DCPK2
CCFF  14        1724        INC   D
CD00  281E      1725        JR    Z,DCPK3
CD02  14        1726        INC   D
CD03  282C      1727        JR    Z,DCPK4
CD05  FE0B      1728        CP    0BH         ;test Call Req
CD07  C259CD    1729        JP    NZ,INVPAK
CD0A  225CF9    1730        LD    (CRADD),HL  ;Call Req/save point
CD0D  2148F9    1731        LD    HL,CRDL     ;save data length
CD10  72        1732        LD    (HL),D
CD11  0600      1733        LD    B,00        ;Call Req int code
CD13  C3F2CD    1734        JP    DCST
CD16  0640      1735 DCPK2  LD    B,40H       ;Data type int code
CD18  FE1B      1736        CP    1BH         ;test Reset Req
CD1A  CAF2CD    1737        JP    Z,DCST
CD1D  C359CD    1738        JP    INVPAK
CD20  0620      1739 DCPK3  LD    B,20H       ;Clear Req int code
CD22  FE13      1740        CP    13H         ;test Clear Req
```

```
CD24 CAF2CD      1741          JP    Z,DCST
CD27 0640        1742          LD    B,40H       ;Data type int code
CD29 FE23        1743          CP    23H         ;test Interrupt pack
CD2B CAF2CD      1744          JP    Z,DCST
CD2E C359CD      1745          JP    INVPAK
CD31 CB4F        1746 DCPK4    BIT   1,A         ;test Sup pack
CD33 281D        1747          JR    Z,DCPK5
CD35 0610        1748          LD    B,10H       ;non Sup/Call Acc code
CD37 FE0F        1749          CP    0FH         ;test Call Acc
CD39 CAF2CD      1750          JP    Z,DCST
CD3C 0630        1751          LD    B,30H       ;Clear Conf int code
CD3E FE17        1752          CP    17H         ;test Clear Conf
CD40 CAF2CD      1753          JP    Z,DCST
CD43 0640        1754          LD    B,40H       ;Data type int code
CD45 FE1F        1755          CP    1FH         ;test Reset Conf
CD47 CAF2CD      1756          JP    Z,DCST
CD4A FE27        1757          CP    27H         ;test Int Conf
CD4C CAF2CD      1758          JP    Z,DCST
CD4F C359CD      1759          JP    INVPAK
CD52 0640        1760 DCPK5    LD    B,40H       ;Sup/Data type code
CD54 E618        1761          AND   18H         ;mask Sup code
CD56 CAF2CD      1762          JP    Z,DCST
CD59 0650        1763 INVPAK   LD    B,50H       ;inv pack
CD5B C3F2CD      1764          JP    DCST
CD5E E6F0        1765 DCPK6    AND   0F0H        ;chan=0/get ID
CD60 FE10        1766          CP    10H         ;validate ID
CD62 C27DCD      1767          JP    NZ,INVRST
CD65 7E          1768          LD    A,(HL)
CD66 15          1769          DEC   D           ;validate length
CD67 2008        1770          JR    NZ,DCPK7
CD69 FEFB        1771          CP    0FBH        ;test Restart Req
CD6B CA93CD      1772          JP    Z,RSTRQ
CD6E C37DCD      1773          JP    INVRST
CD71 14          1774 DCPK7    INC   D
CD72 C27DCD      1775          JP    NZ,INVRST
CD75 FEFF        1776          CP    0FFH        ;test Restart Conf
CD77 CAC3CD      1777          JP    Z,RSTCF
CD7A C37DCD      1778          JP    INVRST
                 1779 ;
CD7D 2E5B        1780 INVRST   LD    L,M91       ;inv pack (ch=0)
CD7F 63          1781          LD    H,E
CD80 CDDDDB      1782          CALL  LDMES
CD83 CB49        1783          BIT   DCERST,C    ;test Rst sent to DTE
CD85 2006        1784          JR    NZ,RS1
CD87 CDDACD      1785          CALL  RSTERR      ;error (no Rst)
CD8A C399D0      1786          JP    ENDPIQ
CD8D CD5CC4      1787 RS1      CALL  MREL        ;restarting
CD90 C399D0      1788          JP    ENDPIQ
CD93 2E0E        1789 RSTRQ    LD    L,M14       ;Restart Req
CD95 2602        1790          LD    H,M02
CD97 CDDDDB      1791          CALL  LDMES
CD9A DD2168FA    1792          LD    IX,DTERS
CD9E CDE3D6      1793          CALL  INCRCD
CDA1 CB49        1794          BIT   DCERST,C    ;test Rst sent to DTE
CDA3 2018        1795          JR    NZ,RS2
CDA5 2149F9      1796          LD    HL,CLRST    ;accept/process
CDA8 3600        1797          LD    (HL),00
CDAA CD3CD7      1798          CALL  RST
CDAD CD69D6      1799          CALL  RSTCNF      ;send Rest Conf
CDB0 CDB6D5      1800          CALL  INT
```

```
CDB3  2E11        1801          LD      L,M17
CDB5  2602        1802          LD      H,M02
CDB7  CDDDDB      1803          CALL    LDMES
CDBA  C399D0      1804          JP      ENDPIQ
CDBD  CDEACD      1805 RS2      CALL    RSTEND          ;Restart complete
CDC0  C399D0      1806          JP      ENDPIQ
CDC3  2E10        1807 RSTCF    LD      L,M16           ;Restart Conf
CDC5  2602        1808          LD      H,M02
CDC7  CDDDDB      1809          CALL    LDMES
CDCA  CB49        1810          BIT     DCERST,C        ;test Rst sent to DTE
CDCC  2006        1811          JR      NZ,RS3
CDCE  CDDACD      1812          CALL    RSTERR          ;error (no Rst)
CDD1  C399D0      1813          JP      ENDPIQ
CDD4  CDEACD      1814 RS3      CALL    RSTEND          ;Restart complete
CDD7  C399D0      1815          JP      ENDPIQ
CDDA  2149F9      1816 RSTERR   LD      HL,CLRST        ;Restart error
CDDD  3611        1817          LD      (HL),11H
CDDF  CD3CD7      1818          CALL    RST
CDE2  CBC9        1819          SET     DCERST,C        ;send Rest Ind
CDE4  1601        1820          LD      D,01H
CDE6  CDC4D6      1821          CALL    RSTINT
CDE9  C9          1822          RET
CDEA  CB81        1823 RSTEND   RES     TONRST,C        ;clear Rst condition
CDEC  CB89        1824          RES     DCERST,C
CDEE  CD5CC4      1825          CALL    MREL
CDF1  C9          1826          RET
                  1827 ;
CDF2  CB49        1828 DCST     BIT     DCERST,C        ;non Rst pack/test Rst
CDF4  C28ED0      1829          JP      NZ,REJPK2
CDF7  78          1830          LD      A,B
CDF8  0F          1831          RRCA
CDF9  0F          1832          RRCA
CDFA  0F          1833          RRCA
CDFB  C651        1834          ADD     M81
CDFD  6F          1835          LD      L,A
CDFE  63          1836          LD      H,E
CDFF  CDDDDB      1837          CALL    LDMES
CE02  2100DD      1838          LD      HL,PKTAB        ;point event/stat tab
CE05  1600        1839          LD      D,00
CE07  FD2100F9    1840          LD      IY,CSI          ;point chan stat tab
CE0B  FD19        1841          ADD     IY,DE           ;point chan state
CE0D  FD7E00      1842          LD      A,(IY)          ;read chan state
CE10  E60F        1843          AND     STAMSK          ;mask state code
CE12  B0          1844          OR      B               ;or int pack code
CE13  07          1845          RLCA                    ;table offset
CE14  85          1846          ADD     L
CE15  6F          1847          LD      L,A
CE16  7E          1848          LD      A,(HL)          ;LSB rout address
CE17  2C          1849          INC     L
CE18  66          1850          LD      H,(HL)          ;MSB rout address
CE19  6F          1851          LD      L,A
CE1A  E9          1852          JP      (HL)            ;jump event/stat rout
                  1853 ;
                  1854 ;in the following Ri routines IY points to the
                  1855 ;channel state; each routine updates the channel
                  1856 ;state and counters, formats a new packet if required
                  1857 ;and transfers the packet to the POQ or the BOQ
                  1858 ;
CE1B  2A5CF9      1859 R1       LD      HL,(CRADD)      ;Call Req/get point
CE1E  23          1860          INC     HL              ;point addr length
```

```
CE1F C5          1861         PUSH   BC
CE20 D5          1862         PUSH   DE
CE21 E5          1863         PUSH   HL
CE22 7E          1864         LD     A,(HL)          ;read addr length
CE23 47          1865         LD     B,A
CE24 CB38        1866         SRL    B
CE26 CB38        1867         SRL    B
CE28 CB38        1868         SRL    B
CE2A CB38        1869         SRL    B
CE2C E60F        1870         AND    0FH             ;called addr length
CE2E 4F          1871         LD     C,A             ;calling addr length
CE2F 80          1872         ADD    B
CE30 3C          1873         INC    A
CE31 CB3F        1874         SRL    A               ;nbr address bytes
CE33 5F          1875         LD     E,A
CE34 1C          1876         INC    E
CE35 AF          1877         XOR    A
CE36 57          1878         LD     D,A
CE37 19          1879         ADD    HL,DE           ;point fac length
CE38 BE          1880         CP     (HL)            ;validate fac length
CE39 E1          1881         POP    HL
CE3A C2E9CE      1882         JP     NZ,INVCR
CE3D 1C          1883         INC    E
CE3E 3A48F9      1884         LD     A,(CRDL)        ;User Data length
CE41 93          1885         SUB    E               ;validate
CE42 DAE9CE      1886         JP     C,INVCR
CE45 D611        1887         SUB    CUD+1
CE47 D2E9CE      1888         JP     NC,INVCR
CE4A AF          1889         XOR    A               ;test addr length (>0)
CE4B B8          1890         CP     B
CE4C CAE9CE      1891         JP     Z,INVCR
CE4F B9          1892         CP     C
CE50 CAE9CE      1893         JP     Z,INVCR
CE53 EB          1894 CDADD   EX     DE,HL           ;called addr (DE)
CE54 2100DF      1895         LD     HL,ADTAB        ;addrr table (HL)
CE57 3A4DF9      1896         LD     A,(SIGDIG)      ;sig dig (loc addr)
CE5A 47          1897         LD     B,A
CE5B 79          1898         LD     A,C             ;called addr length
CE5C B8          1899         CP     B               ;compare lengths
CE5D 300D        1900         JR     NC,CDADD2
CE5F 3E01        1901         LD     A,ADDSX         ;called<local length
CE61 B8          1902         CP     B
CE62 2804        1903         JR     Z,CDADD1
CE64 3E0A        1904         LD     A,EXTDCE        ;to gateway (LAU=11)
CE66 182A        1905         JR     CDADD8
CE68 3E00        1906 CDADD1  LD     A,00            ;to LAU=0
CE6A 1826        1907         JR     CDADD8
CE6C B5          1908 CDADD2  OR     L               ;called>=local length
CE6D 6F          1909         LD     L,A             ;point high dig (tab)
CE6E 79          1910         LD     A,C
CE6F D600        1911         SUB    ADDSX-1
CE71 47          1912         LD     B,A             ;nbr dig to compare
CE72 0EAA        1913         LD     C,0AAH          ;get right/left dig
CE74 CB09        1914 CDADD3  RRC    C
CE76 3005        1915         JR     NC,CDADD4
CE78 1A          1916         LD     A,(DE)
CE79 E60F        1917         AND    0FH             ;right digit
CE7B 1808        1918         JR     CDADD5
CE7D 13          1919 CDADD4  INC    DE              ;point next byte
CE7E 1A          1920         LD     A,(DE)
```

```
CE7F E6F0        1921         AND   0F0H
CE81 0F          1922         RRCA
CE82 0F          1923         RRCA
CE83 0F          1924         RRCA
CE84 0F          1925         RRCA                    ;left digit
CE85 BE          1926 CDADD5  CP    (HL)              ;compare
CE86 2807        1927         JR    Z,CDADD7
CE88 05          1928         DEC   B                 ;different
CE89 2802        1929         JR    Z,CDADD6          ;test if last dig
CE8B 3E0A        1930         LD    A,EXTDCE          ;to gateway (LAU=11)
CE8D 0601        1931 CDADD6  LD    B,01H             ;ends comparison
CE8F 2B          1932 CDADD7  DEC   HL                ;point next dig (tab)
CE90 10E2        1933         DJNZ  CDADD3            ;repeat
CE92 FE0A        1934 CDADD8  CP    EXTDCE            ;validate routing
CE94 2006        1935         JR    NZ,VALCR
CE96 214CF9      1936         LD    HL,LOCADD
CE99 BE          1937         CP    (HL)
CE9A 284D        1938         JR    Z,INVCR
CE9C D1          1939 VALCR   POP   DE                ;valid Call Req
CE9D C1          1940         POP   BC
CE9E 47          1941         LD    B,A               ;save rem LAU
CE9F 21E0F8      1942         LD    HL,RDCEST         ;point LAU stat tab
CEA2 85          1943         ADD   L
CEA3 6F          1944         LD    L,A               ;point LAUi state
CEA4 CB46        1945         BIT   ONI,(HL)          ;test LAUi On
CEA6 2008        1946         JR    NZ,VALCR1
CEA8 16D0        1947         LD    D,0DH             ;LAUi Off
CEAA CDF3CE      1948         CALL  REJCR             ;reject Call
CEAD C399D0      1949         JP    ENDPIQ
CEB0 CB56        1950 VALCR1  BIT   RSTI,(HL)         ;test LAUi Restart
CEB2 2808        1951         JR    Z,VALCR2
CEB4 1601        1952         LD    D,01H             ;Restart in progress
CEB6 CDF3CE      1953         CALL  REJCR             ;reject Call
CEB9 C399D0      1954         JP    ENDPIQ
CEBC CBDE        1955 VALCR2  SET   COMI,(HL)         ;accept/resp expect
CEBE FD3600 2A   1956         LD    (IY),P10          ;new state
CEC2 CDACD5      1957         CALL  INCCH             ;inc active chan
CEC5 CDA2D5      1958         CALL  INCTM2
CEC8 78          1959         LD    A,B
CEC9 21C0F8      1960         LD    HL,RDCENC         ;inc LAUi calls
CECC 85          1961         ADD   L
CECD 6F          1962         LD    L,A
CECE 34          1963         INC   (HL)
CECF 78          1964         LD    A,B
CED0 21D0F8      1965         LD    HL,RDCEHC         ;inc LAUi set-up calls
CED3 85          1966         ADD   L
CED4 6F          1967         LD    L,A
CED5 34          1968         INC   (HL)
CED6 78          1969         LD    A,B
CED7 2108F9      1970         LD    HL,RDCE           ;map remote LAU
CEDA 1600        1971         LD    D,00
CEDC 19          1972         ADD   HL,DE
CEDD 77          1973         LD    (HL),A
CEDE 2110F9      1974         LD    HL,RCI            ;map rem chan(=0)
CEE1 19          1975         ADD   HL,DE
CEE2 72          1976         LD    (HL),D
CEE3 CDDCD5      1977         CALL  EXT1              ;transf to BOQ
CEE6 C399D0      1978         JP    ENDPIQ
CEE9 D1          1979 INVCR   POP   DE                ;inv Call Req
CEEA C1          1980         POP   BC
```

```
CEEB  1603      1981        LD    D,03
CEED  CDF3CE    1982        CALL  REJCR       ;reject Call
CEF0  C399D0    1983        JP    ENDPIQ
CEF3  FD360017  1984 REJCR  LD    (IY),P7     ;local Call clear
CEF7  CD9DD5    1985        CALL  INCTM1
CEFA  CDACD5    1986        CALL  INCCH       ;inc active channels
CEFD  CD78D6    1987        CALL  CLINT
CF00  C9        1988        RET
CF01  FD360017  1989 R2     LD    (IY),P7
CF05  CD9DD5    1990        CALL  INCTM1
CF08  CDACD5    1991        CALL  INCCH
CF0B  1613      1992        LD    D,13H
CF0D  CD78D6    1993        CALL  CLINT
CF10  C399D0    1994        JP    ENDPIQ
CF13  CD96D6    1995 R3     CALL  CCINT
CF16  C399D0    1996        JP    ENDPIQ
CF19  FD360000  1997 R4     LD    (IY),P1
CF1D  CDB1D5    1998        CALL  DECCH
CF20  1632      1999        LD    D,32H
CF22  CD36D6    2000        CALL  DIAGN
CF25  CD7ED5    2001        CALL  DECSCN
CF28  C399D0    2002        JP    ENDPIQ
CF2B  FD360000  2003 R5     LD    (IY),P1
CF2F  CDB1D5    2004        CALL  DECCH
CF32  CD5CC4    2005        CALL  MREL
CF35  CD7ED5    2006        CALL  DECSCN
CF38  C399D0    2007        JP    ENDPIQ
CF3B  CD93D5    2008 R6     CALL  STOP
CF3E  CDA2D5    2009        CALL  INCTM2
CF41  FD360025  2010        LD    (IY),P5
CF45  1605      2011        LD    D,05H
CF47  CDA3D6    2012        CALL  CLEXT
CF4A  C399D0    2013        JP    ENDPIQ
CF4D  CD93D5    2014 R7     CALL  STOP
CF50  CDA2D5    2015        CALL  INCTM2
CF53  FD360024  2016        LD    (IY),P4
CF57  CDCDD5    2017        CALL  EXT
CF5A  DD2162FA  2018        LD    IX,CALB
CF5E  CDE3D6    2019        CALL  INCRCD
CF61  C399D0    2020        JP    ENDPIQ
CF64  CD93D5    2021 R8     CALL  STOP
CF67  CDA2D5    2022        CALL  INCTM2
CF6A  FD360026  2023        LD    (IY),P6
CF6E  CDCDD5    2024        CALL  EXT
CF71  C399D0    2025        JP    ENDPIQ
CF74  CD93D5    2026 R9     CALL  STOP
CF77  CDA2D5    2027        CALL  INCTM2
CF7A  FD360028  2028        LD    (IY),P8
CF7E  1611      2029        LD    D,11H
CF80  CDA3D6    2030        CALL  CLEXT
CF83  C399D0    2031        JP    ENDPIQ
CF86  FD360028  2032 R10    LD    (IY),P8
CF8A  1611      2033        LD    D,11H
CF8C  CDA3D6    2034        CALL  CLEXT
CF8F  C399D0    2035        JP    ENDPIQ
CF92  FD360026  2036 R11    LD    (IY),P6
CF96  CDCDD5    2037        CALL  EXT
CF99  C399D0    2038        JP    ENDPIQ
CF9C  FD360024  2039 R12    LD    (IY),P4
CFA0  CDCDD5    2040        CALL  EXT
```

```
CFA3  DD2164FA     2041          LD    IX,DTEDAT
CFA7  CDE3D6       2042          CALL  INCRCD
CFAA  C399D0       2043          JP    ENDPIQ
CFAD  FD7E00       2044 R13      LD    A,(IY)
CFB0  E6FC         2045          AND   STMSK1
CFB2  F628         2046          OR    P8
CFB4  FD7700       2047          LD    (IY),A
CFB7  CD5CC4       2048          CALL  MREL
CFBA  C399D0       2049          JP    ENDPIQ
CFBD  FD7E00       2050 R14      LD    A,(IY)
CFC0  E6F0         2051          AND   STMSK1
CFC2  F626         2052          OR    P6
CFC4  FD7700       2053          LD    (IY),A
CFC7  CD5CC4       2054          CALL  MREL
CFCA  C399D0       2055          JP    ENDPIQ
CFCD  CD93D5       2056 R15      CALL  STOP
CFD0  FD360000     2057          LD    (IY),P1
CFD4  CDB1D5       2058          CALL  DECCH
CFD7  CD5CC4       2059          CALL  MREL
CFDA  C399D0       2060          JP    ENDPIQ
CFDD  FD360017     2061 R16      LD    (IY),P7
CFE1  CDB7D6       2062          CALL  CCEXT
CFE4  CD15D7       2063          CALL  DECNC
CFE7  C399D0       2064          JP    ENDPIQ
CFEA  CD93D5       2065 R17      CALL  STOP
CFED  FD360000     2066          LD    (IY),P1
CFF1  CDB1D5       2067          CALL  DECCH
CFF4  CDB7D6       2068          CALL  CCEXT
CFF7  CD15D7       2069          CALL  DECNC
CFFA  C399D0       2070          JP    ENDPIQ
CFFD  CD93D5       2071 R18      CALL  STOP
D000  FD360000     2072          LD    (IY),P1
D004  CDB1D5       2073          CALL  DECCH
D007  CDCDD5       2074          CALL  EXT
D00A  CD15D7       2075          CALL  DECNC
D00D  C399D0       2076          JP    ENDPIQ
D010  FD360017     2077 R19      LD    (IY),P7
D014  CD9DD5       2078          CALL  INCTM1
D017  1613         2079          LD    D,13H
D019  CD78D6       2080          CALL  CLINT
D01C  CD7ED5       2081          CALL  DECSCN
D01F  C399D0       2082          JP    ENDPIQ
D022  FD360000     2083 R20      LD    (IY),P1
D026  CDB1D5       2084          CALL  DECCH
D029  CD96D6       2085          CALL  CCINT
D02C  CD7ED5       2086          CALL  DECSCN
D02F  C399D0       2087          JP    ENDPIQ
D032  FD360028     2088 R21      LD    (IY),P8
D036  CDA2D5       2089          CALL  INCTM2
D039  1611         2090          LD    D,11H
D03B  CDA3D6       2091          CALL  CLEXT
D03E  CD7ED5       2092          CALL  DECSCN
D041  C399D0       2093          JP    ENDPIQ
D044  FD360026     2094 R22      LD    (IY),P6
D048  CDA2D5       2095          CALL  INCTM2
D04B  CDCDD5       2096          CALL  EXT
D04E  CD7ED5       2097          CALL  DECSCN
D051  C399D0       2098          JP    ENDPIQ
D054  FD36008B     2099 R23      LD    (IY),P11
D058  CDB7D6       2100          CALL  CCEXT
```

```
D05B  CD15D7      2101          CALL    DECNC
D05E  C399D0      2102          JP      ENDPIQ
D061  FD360000    2103 R24      LD      (IY),P1
D065  CDB1D5      2104          CALL    DECCH
D068  CDB7D6      2105          CALL    CCEXT
D06B  CD15D7      2106          CALL    DECNC
D06E  C399D0      2107          JP      ENDPIQ
D071  FD360000    2108 R25      LD      (IY),P1
D075  CDB1D5      2109          CALL    DECCH
D078  CDCDD5      2110          CALL    EXT
D07B  CD15D7      2111          CALL    DECNC
D07E  C399D0      2112          JP      ENDPIQ
D081  CD5CC4      2113 R26      CALL    MREL
D084  C399D0      2114          JP      ENDPIQ
D087  AF          2115 REJPAK   XOR     A               ;reject pack
D088  1806        2116          JR      REJPK3
D08A  3E01        2117 REJPK1   LD      A,01
D08C  1802        2118          JR      REJPK3
D08E  3E02        2119 REJPK2   LD      A,02
D090  2E60        2120 REJPK3   LD      L,M96
D092  67          2121          LD      H,A
D093  CDDDDB      2122          CALL    LDMES
D096  CD40C4      2123          CALL    RELBUF
D099  C9          2124 ENDPIQ   RET
                  2125 ;
                  2126 ;        ***********************************************
                  2127 ;        * PROCESS PACKET RECEIVED FROM THE LAU (BIQ) *
                  2128 ;        ***********************************************
                  2129 ;
D09A  2175F9      2130 BIQ      LD      HL,CTQ4         ;read nbr pack (BIQ)
D09D  AF          2131          XOR     A
D09E  BE          2132          CP      (HL)
D09F  CA83D3      2133          JP      Z,ENDBIQ
D0A2  35          2134          DEC     (HL)            ;dec BIQ counter
D0A3  2176F9      2135          LD      HL,DFQ4         ;get packet
D0A6  7E          2136          LD      A,(HL)
D0A7  34          2137          INC     (HL)
D0A8  2160F8      2138          LD      HL,TPQ4
D0AB  CD01C4      2139          CALL    GETBUF
D0AE  23          2140 PCTR1    INC     HL              ;point rem LAU field
D0AF  56          2141          LD      D,(HL)          ;read rem LAU
D0B0  23          2142          INC     HL              ;point rem chan field
D0B1  5E          2143          LD      E,(HL)          ;read rem channel
D0B2  ED534AF9    2144          LD      (REMOTE),DE     ;save rem LAU/channel
D0B6  23          2145          INC     HL
D0B7  23          2146          INC     HL              ;point local chan
D0B8  5E          2147          LD      E,(HL)          ;read local channel
D0B9  23          2148          INC     HL              ;point pack field
D0BA  7E          2149          LD      A,(HL)          ;read pack
D0BB  FEF3        2150          CP      MNREQ           ;test if EMU command
D0BD  C2C6D0      2151          JP      NZ,DECPK
D0C0  CD74D8      2152          CALL    MNRQ            ;process EMU command
D0C3  C383D3      2153          JP      ENDBIQ
D0C6  47          2154 DECPK    LD      B,A
D0C7  7A          2155          LD      A,D
D0C8  21E0F8      2156          LD      HL,RDCEST       ;point LAU state tab
D0CB  85          2157          ADD     L
D0CC  6F          2158          LD      L,A             ;point LAUi state
D0CD  78          2159          LD      A,B             ;packet type
D0CE  FEFB        2160          CP      0FBH            ;test LAU Restart
```

```
D0D0  CA1DD1        2161          JP      Z,BRSRQ
D0D3  FEFF          2162          CP      0FFH            ;test LAU Rst Conf
D0D5  CA3ED1        2163          JP      Z,BRSCF
D0D8  CB56          2164          BIT     RSTI,(HL)       ;no Rst pack/test Rst
D0DA  2808          2165          JR      Z,DECPK1
D0DC  CBC6          2166          SET     ONI,(HL)        ;Restarting (LAUi)
D0DE  CD40C4        2167          CALL    RELBUF
D0E1  C383D3        2168          JP      ENDBIQ
D0E4  CB46          2169 DECPK1   BIT     ONI,(HL)        ;test if LAUi ON
D0E6  2008          2170          JR      NZ,DECPK2
D0E8  360D          2171          LD      (HL),RSCMI      ;OFF
D0EA  CD60D1        2172          CALL    BUSERR          ;Bus Rst error
D0ED  C383D3        2173          JP      ENDBIQ
D0F0  FE0B          2174 DECPK2   CP      0BH             ;ON/test Inc Call
D0F2  CA72D1        2175          JP      Z,ICALL
D0F5  0600          2176          LD      B,00            ;Call Conn int code
D0F7  FE0F          2177          CP      0FH             ;test Call Conn
D0F9  CAFBD1        2178          JP      Z,DECST
D0FC  0610          2179          LD      B,10H           ;Clear Ind int code
D0FE  FE13          2180          CP      13H             ;test Clear Indication
D100  CAFBD1        2181          JP      Z,DECST
D103  0620          2182          LD      B,20H           ;Clear Conf int code
D105  FE17          2183          CP      17H             ;test Clear Conf
D107  CAFBD1        2184          JP      Z,DECST
D10A  0640          2185          LD      B,40H           ;HDCF int code
D10C  FE87          2186          CP      HDCF            ;test HDCF
D10E  CA07D2        2187          JP      Z,DECST1
D111  0650          2188          LD      B,50H           ;HDRJ int code
D113  FE83          2189          CP      HDRJ            ;test HDRJ
D115  CA07D2        2190          JP      Z,DECST1
D118  0630          2191          LD      B,30H           ;Data type int code
D11A  C3FBD1        2192          JP      DECST
                    2193 ;
D11D  CBC6          2194 BRSRQ    SET     ONI,(HL)        ;LAU Rst rcv/process
D11F  DD2A56F9      2195          LD      IX,(CURBUF)     ;get Rst reason
D123  DD7E06        2196          LD      A,(IX+6)
D126  3249F9        2197          LD      (CLRST),A
D129  CD95D7        2198          CALL    BUSRST          ;clear LAUi calls
D12C  CD69D6        2199          CALL    RSTCNF          ;send Rst Conf (LAUi)
D12F  3A4BF9        2200          LD      A,(REMCTE+1)
D132  110000        2201          LD      DE,00
D135  CDDCD5        2202          CALL    EXT1
D138  CD22D8        2203          CALL    BOQOV           ;test BOQ overflow
D13B  C383D3        2204          JP      ENDBIQ
D13E  CBC6          2205 BRSCF    SET     ONI,(HL)        ;LAU Rst Conf rcv/proc
D140  CB56          2206          BIT     RSTI,(HL)       ;test Rst in progress
D142  2010          2207          JR      NZ,BRSCF1
D144  360D          2208          LD      (HL),RSCMI
D146  CD60D1        2209          CALL    BUSERR          ;Bus Rst error
D149  3E13          2210          LD      A,13H
D14B  3249F9        2211          LD      (CLRST),A
D14E  CD95D7        2212          CALL    BUSRST          ;clear LAUi calls
D151  C383D3        2213          JP      ENDBIQ
D154  CB5E          2214 BRSCF1   BIT     COMI,(HL)       ;test Rst sent (LAUi)
D156  2802          2215          JR      Z,BRSCF2
D158  3601          2216          LD      (HL),INONI      ;valid Conf/end Rst
D15A  CD40C4        2217 BRSCF2   CALL    RELBUF
D15D  C383D3        2218          JP      ENDBIQ
D160  1611          2219 BUSERR   LD      D,11H           ;Bus Rst error
D162  CD58D6        2220          CALL    RSTIND          ;send Rst (LAUi)
```

```
D165  3A4BF9          2221          LD      A,(REMOTE+1)
D168  110000          2222          LD      DE,00
D16B  CDDCD5          2223          CALL    EXT1
D16E  CD22D8          2224          CALL    BOQOV
D171  C9              2225          RET
                      2226 ;
D172  3A40F9          2227 ICALL    LD      A,(BUFREC)      ;proc Inc Call
D175  CB47            2228          BIT     BY,A            ;test Busy
D177  C2D8D1          2229          JP      NZ,REJIC
D17A  CB49            2230          BIT     DCERST,C        ;test Rst (X25 Interf)
D17C  C2D8D1          2231          JP      NZ,REJIC
D17F  3E07            2232          LD      A,MXCH          ;test chan available
D181  2142F9          2233          LD      HL,NRCH
D184  BE              2234          CP      (HL)
D185  CAD8D1          2235          JP      Z,REJIC
D188  34              2236          INC     (HL)            ;avail/inc active ch
D189  2100F9          2237          LD      HL,CSI          ;search lowest ch (p1)
D18C  3E00            2238          LD      A,P1
D18E  23              2239 ICAL0    INC     HL
D18F  BE              2240          CP      (HL)
D190  20FC            2241          JR      NZ,ICAL0
D192  7D              2242          LD      A,L
D193  E60F            2243          AND     CHMASK
D195  5F              2244          LD      E,A             ;channel number
D196  3613            2245          LD      (HL),P3
D198  CD9DD5          2246          CALL    INCTM1
D19B  DD2A56F9        2247          LD      IX,(CURBUF)
D19F  DD7304          2248          LD      (IX+4),E        ;insert ch nbr (header)
D1A2  CDB6D5          2249          CALL    INT             ;transf to POQ
D1A5  2E52            2250          LD      L,M82
D1A7  63              2251          LD      H,E
D1A8  CDDDDB          2252          CALL    LDMES
D1AB  CD1DC4          2253          CALL    GTPOOL          ;get buff (pool)
D1AE  CD27D6          2254          CALL    HDCNF           ;format HDCF
D1B1  1600            2255          LD      D,00
D1B3  2110F9          2256          LD      HL,RCI          ;map rem channel
D1B6  19              2257          ADD     HL,DE
D1B7  3A4AF9          2258          LD      A,(REMOTE)      ;rem channel
D1BA  77              2259          LD      (HL),A
D1BB  2108F9          2260          LD      HL,RDCE         ;map rem LAU
D1BE  19              2261          ADD     HL,DE
D1BF  57              2262          LD      D,A
D1C0  3A4BF9          2263          LD      A,(REMOTE+1)    ;rem LAU
D1C3  77              2264          LD      (HL),A
D1C4  47              2265          LD      B,A
D1C5  21C0F8          2266          LD      HL,RDCENC       ;inc LAUi calls
D1C8  85              2267          ADD     L
D1C9  6F              2268          LD      L,A
D1CA  34              2269          INC     (HL)
D1CB  78              2270          LD      A,B
D1CC  CDDCD5          2271          CALL    EXT1            ;transf to BOQ
D1CF  2E02            2272          LD      L,M02
D1D1  65              2273          LD      H,L
D1D2  CDDDDB          2274          CALL    LDMES
D1D5  C383D3          2275          JP      ENDBIQ
D1D8  1601            2276 REJIC    LD      D,01            ;reject Inc Call
D1DA  CD16D6          2277          CALL    HDREJ           ;format HDRJ
D1DD  3A4AF9          2278          LD      A,(REMOTE)      ;get rem channel
D1E0  57              2279          LD      D,A
D1E1  3A4BF9          2280          LD      A,(REMOTE+1)    ;get rem LAU
```

```
D1E4  1E00        2281          LD      E,00        ;local chan (=0)
D1E6  CDDCD5      2282          CALL    EXTI        ;transf to BOQ
D1E9  2E52        2283          LD      L,M82
D1EB  63          2284          LD      H,E
D1EC  CDDDDB      2285          CALL    LDMES
D1EF  2E5D        2286          LD      L,M93
D1F1  63          2287          LD      H,E
D1F2  CDDDDB      2288          CALL    LDMES
D1F5  CD22D8      2289          CALL    BOQOV       ;test BOQ overflow
D1F8  C383D3      2290          JP      ENDBIQ
                  2291 ;
D1FB  78          2292 DECST    LD      A,B         ;process other pack
D1FC  0F          2293          RRCA
D1FD  0F          2294          RRCA
D1FE  0F          2295          RRCA
D1FF  00          2296          NOP
D200  C654        2297          ADD     M84
D202  6F          2298          LD      L,A
D203  63          2299          LD      H,E
D204  CDDDDB      2300          CALL    LDMES
D207  2100DE      2301 DECST1   LD      HL,PKTAB1   ;point event/stat tab
D20A  1600        2302          LD      D,00
D20C  FD2100F9    2303          LD      IY,CSI      ;point chan stat tab
D210  FD19        2304          ADD     IY,DE       ;point chan state
D212  FD7E00      2305          LD      A,(IY)      ;read chan state
D215  E60F        2306          AND     STAMSK      ;mask state code
D217  B0          2307          OR      B           ;or int pack code
D218  07          2308          RLCA                ;table offset
D219  85          2309          ADD     L
D21A  6F          2310          LD      L,A
D21B  7E          2311          LD      A,(HL)      ;LSB rout addr
D21C  2C          2312          INC     L
D21D  66          2313          LD      H,(HL)      ;MSB rout addr
D21E  6F          2314          LD      L,A
D21F  E9          2315          JP      (HL)        ;jump event/stat rout
                  2316 ;
                  2317 ;in the following RBi routines IY points to the
                  2318 ;channel state; each routine updates the channel
                  2319 ;state and counters, formats a new packet if required
                  2320 ;and transfers the packet to the POQ or the BOQ
                  2321 ;
D220  CD5CC4      2322 RB1      CALL    MREL
D223  C383D3      2323          JP      ENDBIQ
D226  FD360024    2324 RB2      LD      (IY),P4
D22A  CDB6D5      2325          CALL    INT
D22D  DD2160FA    2326          LD      IX,CALA
D231  CDE3D6      2327          CALL    INCRCD
D234  C383D3      2328          JP      ENDBIQ
D237  CD93D5      2329 RB3      CALL    STOP
D23A  FD360019    2330          LD      (IY),P9
D23E  CD9DD5      2331          CALL    INCTM1
D241  CD88D6      2332          CALL    CLINTA
D244  C383D3      2333          JP      ENDBIQ
D247  FD360019    2334 RB4      LD      (IY),P9
D24B  CD98D5      2335          CALL    STOP2
D24E  CD9DD5      2336          CALL    INCTM1
D251  CD88D6      2337          CALL    CLINTA
D254  C383D3      2338          JP      ENDBIQ
D257  FD360024    2339 RB5      LD      (IY),P4
D25B  CDB6D5      2340          CALL    INT
```

```
D25E  DD2166FA      2341          LD      IX,DCEDAT
D262  CDE3D6        2342          CALL    INCRCD
D265  C383D3        2343          JP      ENDBIQ
D268  FD360017      2344 RB6      LD      (IY),P7
D26C  CD98D5        2345          CALL    STOP2
D26F  CD9DD5        2346          CALL    INCTM1
D272  1605          2347          LD      D,05
D274  DD2A56F9      2348          LD      IX,(CURBUF)
D278  DD7206        2349          LD      (IX+6),D
D27B  CDB6D5        2350          CALL    INT
D27E  CDDCD6        2351          CALL    CLRSN
D281  CD15D7        2352          CALL    DECNC
D284  C383D3        2353          JP      ENDBIQ
D287  FD360000      2354 RB7      LD      (IY),P1
D28B  CDB1D5        2355          CALL    DECCH
D28E  CD98D5        2356          CALL    STOP2
D291  CD96D6        2357          CALL    CCINT
D294  CD15D7        2358          CALL    DECNC
D297  C383D3        2359          JP      ENDBIQ
D29A  FD360000      2360 RB8      LD      (IY),P1
D29E  CDB1D5        2361          CALL    DECCH
D2A1  CD98D5        2362          CALL    STOP2
D2A4  CDB6D5        2363          CALL    INT
D2A7  CD15D7        2364          CALL    DECNC
D2AA  C383D3        2365          JP      ENDBIQ
D2AD  FD360017      2366 RB9      LD      (IY),P7
D2B1  CD98D5        2367          CALL    STOP2
D2B4  CD9DD5        2368          CALL    INCTM1
D2B7  1613          2369          LD      D,13H
D2B9  DD2A56F9      2370          LD      IX,(CURBUF)
D2BD  DD7206        2371          LD      (IX+6),D
D2C0  CDB6D5        2372          CALL    INT
D2C3  CDDCD6        2373          CALL    CLRSN
D2C6  CD15D7        2374          CALL    DECNC
D2C9  C383D3        2375          JP      ENDBIQ
D2CC  CDF1D6        2376 RB10     CALL    MAP
D2CF  FD360022      2377          LD      (IY),P2
D2D3  CDFCD6        2378          CALL    DECHC
D2D6  CD5CC4        2379          CALL    MREL
D2D9  C383D3        2380          JP      ENDBIQ
D2DC  CDF1D6        2381 RB11     CALL    MAP
D2DF  1600          2382          LD      D,00
D2E1  CDA3D6        2383          CALL    CLEXT
D2E4  CDFCD6        2384          CALL    DECHC
D2E7  CD22D8        2385          CALL    BOQOV
D2EA  C383D3        2386          JP      ENDBIQ
D2ED  CDF1D6        2387 RB12     CALL    MAP
D2F0  1611          2388          LD      D,11H
D2F2  CDA3D6        2389          CALL    CLEXT
D2F5  CDFCD6        2390          CALL    DECHC
D2F8  CD22D8        2391          CALL    BOQOV
D2FB  C383D3        2392          JP      ENDBIQ
D2FE  FD360017      2393 RB13     LD      (IY),P7
D302  CD98D5        2394          CALL    STOP2
D305  CD9DD5        2395          CALL    INCTM1
D308  DD2A56F9      2396          LD      IX,(CURBUF)
D30C  DD5606        2397          LD      D,(IX+6)
D30F  CD78D6        2398          CALL    CLINT
D312  CDFCD6        2399          CALL    DECHC
D315  CD15D7        2400          CALL    DECNC
```

```
D318  C383D3      2401          JP      ENDBIQ
D31B  FD360000    2402 RB14     LD      (IY),P1
D31F  CD98D5      2403          CALL    STOP2
D322  CDB1D5      2404          CALL    DECCH
D325  CD96D6      2405          CALL    CCINT
D328  CDFCD6      2406          CALL    DECHC
D32B  CD15D7      2407          CALL    DECNC
D32E  C383D3      2408          JP      ENDBIQ
D331  FD360017    2409 RB15     LD      (IY),P7
D335  CD98D5      2410          CALL    STOP2
D338  CD9DD5      2411          CALL    INCTM1
D33B  1613        2412          LD      D,13H
D33D  CD78D6      2413          CALL    CLINT
D340  CDFCD6      2414          CALL    DECHC
D343  CD15D7      2415          CALL    DECNC
D346  C383D3      2416          JP      ENDBIQ
D349  FD360019    2417 RB16     LD      (IY),P9
D34D  CD9DD5      2418          CALL    INCTM1
D350  CD88D6      2419          CALL    CLINTA
D353  CD7ED5      2420          CALL    DECSCN
D356  C383D3      2421          JP      ENDBIQ
D359  FD360017    2422 RB17     LD      (IY),P7
D35D  CD98D5      2423          CALL    STOP2
D360  CD9DD5      2424          CALL    INCTM1
D363  1605        2425          LD      D,05H
D365  CD78D6      2426          CALL    CLINT
D368  CD15D7      2427          CALL    DECNC
D36B  C383D3      2428          JP      ENDBIQ
D36E  FD360017    2429 RB18     LD      (IY),P7
D372  CD98D5      2430          CALL    STOP2
D375  CD9DD5      2431          CALL    INCTM1
D378  1613        2432          LD      D,13H
D37A  CD78D6      2433          CALL    CLINT
D37D  CD15D7      2434          CALL    DECNC
D380  C383D3      2435          JP      ENDBIQ
D383  C9          2436 ENDBIQ   RET
                  2437 ;
                  2438 ;        *********
                  2439 ;        * TIMER *
                  2440 ;        *********
                  2441 ;
D384  2150F9      2442 TIM      LD      HL,INTREG       ;lev3 timers
D387  CB56        2443          BIT     T3OUT,(HL)      ;test X25 Interf tim
D389  CA0ED4      2444          JP      Z,BUSTM
D38C  CB96        2445          RES     T3OUT,(HL)      ;time-out
D38E  CB49        2446          BIT     DCERST,C        ;test Rst in progress
D390  2826        2447          JR      Z,TIM1
D392  CB41        2448          BIT     TONRST,C        ;Rst/test timer ON
D394  281D        2449          JR      Z,TIM0
D396  CB81        2450          RES     TONRST,C        ;Rst timer ON/fail
D398  CB89        2451          RES     DCERST,C
D 9A  2147F9      2452          LD      HL,IRLP
D39D  CBD6        2453          SET     PD,(HL)
D39F  CD1FD7      2454          CALL    CLPAK
D3A2  2E18        2455          LD      L,M24
D3A4  2602        2456          LD      H,M02
D3A6  CDDDDB      2457          CALL    LDMES
D3A9  DD2176FA    2458          LD      IX,PFL
D3AD  CDE3D6      2459          CALL    INCRCD
D3B0  C30ED4      2460          JP      BUSTM
```

```
D3B3  CBC1          2461 TIM0   SET   TONRST,C       ;Rst timer OFF/start
D3B5  C3ED04        2462        JP    BUSTM
D3B8  3A43F9        2463 TIM1   LD    A,(STMC)      ;no Rst/ch tim count
D3BB  B7            2464        OR    A
D3BC  CAED04        2465        JP    Z,BUSTM
D3BF  47            2466        LD    B,A           ;nbr chan to scan
D3C0  2100F9        2467        LD    HL,CSI        ;point ch stat tab
D3C3  23            2468 TIM2   INC   HL
D3C4  CB66          2469        BIT   CHTIM1,(HL)   ;test tim condit
D3C6  28FB          2470        JR    Z,TIM2
D3C8  CB76          2471        BIT   CHTON,(HL)    ;test timer ON
D3CA  2004          2472        JR    NZ,TIM3
D3CC  CBF6          2473        SET   CHTON,(HL)    ;timer OFF/start
D3CE  182D          2474        JR    TIM6
D3D0  E5            2475 TIM3   PUSH  HL            ;timer ON/time-out
D3D1  7D            2476        LD    A,L
D3D2  E60F          2477        AND   CHMASK        ;get channel nbr
D3D4  2E62          2478        LD    L,M98
D3D6  67            2479        LD    H,A
D3D7  CDDDDB        2480        CALL  LDMES
D3DA  DD2174FA      2481        LD    IX,TM3
D3DE  CDE3D6        2482        CALL  INCRCD
D3E1  CD93D5        2483        CALL  STOP          ;dec tim count
D3E4  CDA7D5        2484        CALL  INCSPC        ;inc scan count (recov)
D3E7  E1            2485        POP   HL
D3E8  7E            2486        LD    A,(HL)
D3E9  CBB6          2487        RES   CHTON,(HL)
D3EB  FE13          2488        CP    P3            ;test state (p3/p7/p9)
D3ED  2004          2489        JR    NZ,TIM4
D3EF  368C          2490        LD    (HL),P12      ;p3/new stat p12
D3F1  18CA          2491        JR    TIM6
D3F3  FE19          2492 TIM4   CP    P9
D3F5  2004          2493        JR    NZ,TIM5
D3F7  368D          2494        LD    (HL),P13      ;p9/new stat p13
D3F9  18C2          2495        JR    TIM6
D3FB  368B          2496 TIM5   LD    (HL),P11      ;p7/new stat p11
D3FD  10C4          2497 TIM6   DJNZ  TIM2          ;repeat (chan)
D3FF  3A45F9        2498        LD    A,(SPCC)      ;update ch scan variab
D402  B7            2499        OR    A
D403  CAED04        2500        JP    Z,BUSTM
D406  CBD1          2501        SET   SCN1,C
D408  2108F9        2502        LD    HL,CSI+MXCH+1
D40B  2258F9        2503        LD    (SCPT1),HL
D40E  2150F9        2504 BUSTM  LD    HL,INTREG
D411  CB5E          2505        BIT   TBOUT,(HL)    ;test LAU Interf tim
D413  CA3CD4        2506        JP    Z,BUSTM4
D416  CB9E          2507        RES   TBOUT,(HL)    ;time-out
D418  0608          2508        LD    B,NDCE        ;scan all LAU's
D41A  21DFF8        2509        LD    HL,RDCEST-1   ;point LAU stat tab
D41D  23            2510 BUSTM1 INC   HL
D41E  CB5E          2511        BIT   COMI,(HL)     ;test resp expec (LAUi)
D420  2818          2512        JR    Z,BUSTM3
D422  CB66          2513        BIT   TONI,(HL)     ;test timer ON (LAUi)
D424  2004          2514        JR    NZ,BUSTM2
D426  CBE6          2515        SET   TONI,(HL)     ;timer OFF/start (LAUi)
D428  1810          2516        JR    BUSTM3
D42A  3600          2517 BUSTM2 LD    (HL),INOFFI   ;timer ON/time-out
D42C  3E08          2518        LD    A,NDCE        ;get LAUi addr
D42E  90            2519        SUB   B
D42F  324BF9        2520        LD    (REMOTE+1),A
```

```
D432  3E11        2521          LD      A,11H
D434  3249F9      2522          LD      (CLRST),A
D437  CD95D7      2523          CALL    BUSRST          ;clear LAUi calls
D43A  10E1        2524 BUSTM3   DJNZ    BUSTM1          ;repeat all LAU's
D43C  2150F9      2525 BUSTM4   LD      HL,INTREG
D43F  CB66        2526          BIT     TBOUT1,(HL)     ;test LAU recov interv
D441  CA9BD4      2527          JP      Z,BUSTM6
D444  CBA6        2528          RES     TBOUT1,(HL)     ;time-out
D446  3A44F9      2529          LD      A,(TONC)        ;chan tim count
D449  B7          2530          OR      A
D44A  CA98D4      2531          JP      Z,BUSTM5
D44D  47          2532          LD      B,A             ;nbr chan to scan
D44E  2100F9      2533          LD      HL,CSI          ;point ch stat tab
D451  23          2534 BUSFL1   INC     HL
D452  CB6E        2535          BIT     CHTIM2,(HL)     ;test tim condit
D454  28FB        2536          JR      Z,BUSFL1
D456  CB76        2537          BIT     CHTON,(HL)      ;test timer ON
D458  2004        2538          JR      NZ,BUSFL2
D45A  CBF6        2539          SET     CHTON,(HL)      ;timer OFF/start
D45C  1816        2540          JR      BUSFL3
D45E  E5          2541 BUSFL2   PUSH    HL              ;timer ON/time-out
D45F  7D          2542          LD      A,L
D460  E60F        2543          AND     CHMASK          ;get channel nbr
D462  5F          2544          LD      E,A
D463  1600        2545          LD      D,00
D465  2108F9      2546          LD      HL,RDCE         ;get rem LAUi (map)
D468  19          2547          ADD     HL,DE
D469  7E          2548          LD      A,(HL)
D46A  21E0F8      2549          LD      HL,RDCEST       ;point LAU stat tab
D46D  85          2550          ADD     L
D46E  6F          2551          LD      L,A             ;point LAUi state
D46F  CBEE        2552          SET     BFI,(HL)        ;LAUi fail (recov)
D471  CBE1        2553          SET     BF,C            ;LAU recov
D473  E1          2554          POP     HL
D474  10DB        2555 BUSFL3   DJNZ    BUSFL1          ;repeat (chan)
D476  CB61        2556          BIT     BF,C            ;test LAU recov
D478  281E        2557          JR      Z,BUSTM5
D47A  CBA1        2558          RES     BF,C            ;LAU recov
D47C  21DFF8      2559          LD      HL,RDCEST-1     ;point LAU stat tab
D47F  0608        2560          LD      B,NDCE          ;scan all LAU's
D481  23          2561 BUSFL4   INC     HL
D482  CB6E        2562          BIT     BFI,(HL)        ;test LAUi recov
D484  2810        2563          JR      Z,BUSFL5
D486  3600        2564          LD      (HL),INOFFI     ;set LAUi recov (OFF)
D488  3E08        2565          LD      A,NDCE          ;get LAUi addr
D48A  90          2566          SUB     B
D48B  324BF9      2567          LD      (REMOTE+1),A
D48E  3E11        2568          LD      A,11H
D490  3249F9      2569          LD      (CLRST),A
D493  CD95D7      2570          CALL    BUSRST          ;clear LAUi calls
D496  10E9        2571 BUSFL5   DJNZ    BUSFL4          ;repeat all LAU's
D498  CD9CD4      2572 BUSTM5   CALL    INRST           ;init LAU Rst synch
D49B  C9          2573 BUSTM6   RET
D49C  0608        2574 INRST    LD      B,NDCE          ;LAU Rst recov
D49E  AF          2575          XOR     A
D49F  21DFF8      2576          LD      HL,RDCEST-1     ;point LAU stat tab
D4A2  23          2577 INRST1   INC     HL
D4A3  CB46        2578          BIT     ONI,(HL)        ;test LAUi ON
D4A5  2007        2579          JR      NZ,INRST2
D4A7  CB56        2580          BIT     RSTI,(HL)       ;OFF/test Rst
```

```
D4A9  2003        2581          JR      NZ,INRST2
D4AB  3606        2582          LD      (HL),SCOFFI     ;OFF/no Rst/set Rst
D4AD  3C          2583          INC     A
D4AE  10F2        2584 INRST2   DJNZ    INRST1          ;repeat all LAU's
D4B0  2146F9      2585          LD      HL,BSCC         ;upadate scan variab
D4B3  86          2586          ADD     (HL)
D4B4  77          2587          LD      (HL),A
D4B5  B7          2588          OR      A
D4B6  2808        2589          JR      Z,INRST3
D4B8  CBD9        2590          SET     SCN2,C
D4BA  21DFF8      2591          LD      HL,RDCEST-1
D4BD  225AF9      2592          LD      (SCPT2),HL
D4C0  C9          2593 INRST3   RET
                  2594 ;
                  2595 ;        ********
                  2596 ;        * SCAN *
                  2597 ;        ********
                  2598 ;
D4C1  CB59        2599 SCAN     BIT     SCN2,C          ;test LAU scan
D4C3  CAFAD4      2600          JP      Z,SCAN0
D4C6  3A40F9      2601          LD      A,(BUFREG)
D4C9  CB47        2602          BIT     BY,A            ;test Busy
D4CB  C27AD5      2603          JP      NZ,ENDSCN
D4CE  CD1DC4      2604          CALL    GTPOOL          ;not Busy/get buff
D4D1  2A5AF9      2605          LD      HL,(SCPT2)      ;read scan point
D4D4  23          2606 BUSCN1   INC     HL              ;point next LAU
D4D5  CB4E        2607          BIT     SCNI,(HL)       ;test LAUi recov
D4D7  28FB        2608          JR      Z,BUSCN1
D4D9  225AF9      2609          LD      (SCPT2),HL      ;recov/save point
D4DC  CB8E        2610          RES     SCNI,(HL)
D4DE  CBDE        2611          SET     COMI,(HL)
D4E0  7D          2612          LD      A,L
D4E1  E60F        2613          AND     DCEMSK          ;get LAUi addr
D4E3  47          2614          LD      B,A
D4E4  21F0F8      2615          LD      HL,RDCERS       ;get Rst reason
D4E7  85          2616          ADD     L
D4E8  6F          2617          LD      L,A
D4E9  56          2618          LD      D,(HL)
D4EA  CD58D6      2619          CALL    RSTIND          ;form Rst Indic
D4ED  78          2620          LD      A,B
D4EE  110000      2621          LD      DE,00
D4F1  CDDCD5      2622          CALL    EXT1            ;transf to BOQ
D4F4  CD86D5      2623          CALL    DECSC2          ;dec scan count
D4F7  C37AD5      2624 BUSCN2   JP      ENDSCN
D4FA  CB51        2625 SCAN0    BIT     SCN1,C          ;test chan scan
D4FC  CA7AD5      2626          JP      Z,ENDSCN
D4FF  3A40F9      2627          LD      A,(BUFREG)
D502  CB47        2628          BIT     BY,A            ;test Busy
D504  C27AD5      2629          JP      NZ,ENDSCN
D507  CD1DC4      2630          CALL    GTPOOL          ;not Busy/get buff
D50A  2A58F9      2631          LD      HL,(SCPT1)      ;read scan point
D50D  2B          2632 SCAN1    DEC     HL              ;point next chan
D50E  CB7E        2633          BIT     CHSCN,(HL)      ;test chan recov
D510  28FB        2634          JR      Z,SCAN1
D512  2258F9      2635          LD      (SCPT1),HL      ;recov/save point
D515  7E          2636          LD      A,(HL)          ;read chan state
D516  E60F        2637          AND     STAMSK          ;mask state code
D518  07          2638          RLCA
D519  07          2639          RLCA                    ;scan tab offset
D51A  FD21C0DE    2640          LD      IY,SCNTAB       ;point scan table
```

```
D51E 1600          2641          LD      D,00
D520 5F            2642          LD      E,A
D521 FD19          2643          ADD     IY,DE           ;point table entry
D523 7D            2644          LD      A,L
D524 E60F          2645          AND     CHMASK          ;get channel nbr
D526 5F            2646          LD      E,A             ;local channel
D527 DD2A56F9      2647          LD      IX,(CURBUF)
D52B DD7304        2648          LD      (IX+4),E        ;insert chan (header)
D52E FDE9          2649          JP      (IY)            ;jump to 'jump tab'
                   2650 ;
                   2651 ;in the following routines HL points to the
                   2652 ;channel state; each routine updates the channel
                   2653 ;state and counters, formats a new packet if required
                   2654 ;and transfers the packet to the POQ or the BOQ
                   2655 ;
D530 368B          2656 SCNA     LD      (HL),P11
D532 23            2657          INC     HL
D533 2258F9        2658          LD      (SCPT1),HL
D536 CDA7D5        2659          CALL    INCSPC
D539 CDB7D6        2660          CALL    CCEXT
D53C CD15D7        2661          CALL    DECNC
D53F C37AD5        2662          JP      ENDSCN
D542 3600          2663 SCNB     LD      (HL),P1
D544 CDB1D5        2664          CALL    DECCH
D547 CD96D6        2665          CALL    CCINT
D54A C37AD5        2666          JP      ENDSCN
D54D 3628          2667 SCNC     LD      (HL),P8
D54F 1611          2668          LD      D,11H
D551 CDA3D6        2669          CALL    CLEXT
D554 C37AD5        2670          JP      ENDSCN
D557 3617          2671 SCND     LD      (HL),P7
D559 CD9DD5        2672          CALL    INCTM1
D55C 2108F9        2673          LD      HL,RDCE         ;get LAUi addr (map)
D55F 19            2674          ADD     HL,DE
D560 7E            2675          LD      A,(HL)
D561 21F0F8        2676          LD      HL,RDCERS       ;get clear reason
D564 85            2677          ADD     L
D565 6F            2678          LD      L,A
D566 56            2679          LD      D,(HL)
D567 CD78D6        2680          CALL    CLINT
D56A 3600          2681 SCNE     LD      (HL),P1
D56C CDB1D5        2682          CALL    DECCH
D56F 1632          2683          LD      D,32H
D571 CD26D6        2684          CALL    DIAGN
D574 C37AD5        2685          JP      ENDSCN
D577 C37AD5        2686 SCNF     JP      ENDSCN
D57A CD7ED5        2687 ENDSCN   CALL    DECSCN
D57D C9            2688          RET
D57E 2145F9        2689 DECSCN   LD      HL,SPCC         ;dec ch scan count
D581 35            2690          DEC     (HL)
D582 C0            2691          RET     NZ
D583 CB91          2692          RES     SCN1,C          ;end chan scan
D585 C9            2693          RET
D586 2146F9        2694 DECSC2   LD      HL,BSCC         ;dec LAU scan count
D589 35            2695          DEC     (HL)
D58A C0            2696          RET     NZ
D58B CB99          2697          RES     SCN2,C          ;end LAU scan
D58D C9            2698          RET
                   2699 ;
D58E 2141F9        2700 PACK2    LD      HL,GENST        ;save pack state
```

```
D591  4E          2701          LD      C,(HL)
D592  C9          2702          RET
                  2703 ;
                  2704 ;        *************************
                  2705 ;        * PACKET LEVEL ROUTINES *
                  2706 ;        *************************
                  2707 ;
D593  2143F9      2708 STOP     LD      HL,STMC         ;dec STMC count
D596  35          2709          DEC     (HL)
D597  C9          2710          RET
D598  2144F9      2711 STOP2    LD      HL,TONC         ;dec TONC count
D59B  35          2712          DEC     (HL)
D59C  C9          2713          RET
D59D  2143F9      2714 INCTM1   LD      HL,STMC         ;inc STMC count
D5A0  34          2715          INC     (HL)
D5A1  C9          2716          RET
D5A2  2144F9      2717 INCTM2   LD      HL,TONC         ;inc TONC count
D5A5  34          2718          INC     (HL)
D5A6  C9          2719          RET
D5A7  2145F9      2720 INCSPC   LD      HL,SPCC         ;inc SPCC count
D5AA  34          2721          INC     (HL)
D5AB  C9          2722          RET
D5AC  2142F9      2723 INCCH    LD      HL,NRCH         ;inc NRCH count
D5AF  34          2724          INC     (HL)
D5B0  C9          2725          RET
D5B1  2142F9      2726 DECCH    LD      HL,NRCH         ;dec NRCH count
D5B4  35          2727          DEC     (HL)
D5B5  C9          2728          RET
                  2729 ;
D5B6  2171F9      2730 INT      LD      HL,DLQ5         ;transf pack to POQ
D5B9  7E          2731          LD      A,(HL)
D5BA  34          2732          INC     (HL)
D5BB  2180F8      2733          LD      HL,TPQ5
D5BE  CD0EC4      2734          CALL    PUTBUF
D5C1  2172F9      2735          LD      HL,NIF0         ;inc POQ count
D5C4  34          2736          INC     (HL)
D5C5  DD215EFA    2737          LD      IX,PAKB
D5C9  CDE3D6      2738          CALL    INCRCD
D5CC  C9          2739          RET
D5CD  1600        2740 EXT      LD      D,00            ;form/transf pack BOQ
D5CF  2108F9      2741          LD      HL,RDCE
D5D2  19          2742          ADD     HL,DE
D5D3  7E          2743          LD      A,(HL)          ;get remote LAU
D5D4  324BF9      2744          LD      (REMCTE+1),A    ;save remote LAU
D5D7  2110F9      2745          LD      HL,RCI
D5DA  19          2746          ADD     HL,DE
D5DB  56          2747          LD      D,(HL)          ;get remote channel
D5DC  2A56F9      2748 EXT1     LD      HL,(CURBUF)     ;get buffer
D5DF  23          2749          INC     HL
D5E0  77          2750          LD      (HL),A          ;insert remote LAU
D5E1  23          2751          INC     HL
D5E2  73          2752          LD      (HL),E          ;insert local channel
D5E3  23          2753          INC     HL
D5E4  23          2754          INC     HL
D5E5  72          2755          LD      (HL),D          ;insert remote channel
D5E6  217AF9      2756          LD      HL,DLQ3         ;transf pack to BOQ
D5E9  7E          2757          LD      A,(HL)
D5EA  34          2758          INC     (HL)
D5EB  2140F8      2759          LD      HL,TPQ3
D5EE  CD0EC4      2760          CALL    PUTBUF
```

```
D5F1  2178F9          2761          LD      HL,CTQ3         ;inc BOQ count
D5F4  34              2762          INC     (HL)
D5F5  C9              2763          RET
                      2764 ;
D5F6  2A56F9          2765 CLIND    LD      HL,(CURBUF)     ;form Clear pack
D5F9  3607            2766          LD      (HL),PLMIN+1    ;insert length
D5FB  23              2767          INC     HL
D5FC  23              2768          INC     HL
D5FD  23              2769          INC     HL
D5FE  3610            2770          LD      (HL),10H        ;insert ID field
D600  23              2771          INC     HL
D601  23              2772          INC     HL
D602  3613            2773          LD      (HL),13H        ;insert pack code
D604  23              2774          INC     HL
D605  72              2775          LD      (HL),D          ;insert reason
D606  C9              2776          RET
D607  2A56F9          2777 CLCONF   LD      HL,(CURBUF)     ;form Cl Conf pack
D60A  3606            2778          LD      (HL),PLMIN      ;insert length
D60C  23              2779          INC     HL
D60D  23              2780          INC     HL
D60E  23              2781          INC     HL
D60F  3610            2782          LD      (HL),10H        ;insert ID field
D611  23              2783          INC     HL
D612  23              2784          INC     HL
D613  3617            2785          LD      (HL),17H        ;insert pack code
D615  C9              2786          RET
D616  2A56F9          2787 HDREJ    LD      HL,(CURBUF)     ;form HDREJ pack
D619  3607            2788          LD      (HL),PLMIN+1    ;insert length
D61B  23              2789          INC     HL
D61C  23              2790          INC     HL
D61D  23              2791          INC     HL
D61E  3610            2792          LD      (HL),10H        ;insert ID field
D620  23              2793          INC     HL
D621  23              2794          INC     HL
D622  3683            2795          LD      (HL),HDRJ       ;insert pack code
D624  23              2796          INC     HL
D625  72              2797          LD      (HL),D          ;insert reason
D626  C9              2798          RET
D627  2A56F9          2799 HDCNF    LD      HL,(CURBUF)     ;form HDCNF pack
D62A  3606            2800          LD      (HL),PLMIN      ;insert length
D62C  23              2801          INC     HL
D62D  23              2802          INC     HL
D62E  23              2803          INC     HL
D62F  3610            2804          LD      (HL),10H        ;insert ID field
D631  23              2805          INC     HL
D632  23              2806          INC     HL
D633  3687            2807          LD      (HL),HDCF       ;insert pack code
D635  C9              2808          RET
D636  2A56F9          2809 DIAGN    LD      HL,(CURBUF)     ;form Diagnostic
D639  3609            2810          LD      (HL),PLMIN+3    ;insert length
D63B  23              2811          INC     HL
D63C  23              2812          INC     HL
D63D  23              2813          INC     HL
D63E  3610            2814          LD      (HL),10H        ;insert ID field
D640  23              2815          INC     HL
D641  23              2816          INC     HL
D642  36F1            2817          LD      (HL),0F1H       ;insert pack code
D644  23              2818          INC     HL
D645  72              2819          LD      (HL),D          ;insert reason
D646  23              2820          INC     HL
```

```
D647 3610        2821          LD      (HL),10H        ;insert ID field
D649 23          2822          INC     HL
D64A 73          2823          LD      (HL),E          ;insert local channel
D64B CDB6D5      2824          CALL    INT
D64E 2E50        2825          LD      L,M80
D650 63          2826          LD      H,E
D651 CDDDDB      2827          CALL    LDMES
D654 CDDCD6      2828          CALL    CLRSN
D657 C9          2829          RET
D658 2A56F9      2830 RSTIND   LD      HL,(CURBUF)     ;form Restart pack
D65B 3607        2831          LD      (HL),PLMIN+1    ;insert length
D65D 23          2832          INC     HL
D65E 23          2833          INC     HL
D65F 23          2834          INC     HL
D660 3610        2835          LD      (HL),10H        ;insert ID field
D662 23          2836          INC     HL
D663 23          2837          INC     HL
D664 36FB        2838          LD      (HL),0FBH       ;insert pack code
D666 23          2839          INC     HL
D667 72          2840          LD      (HL),D          ;insert reason
D668 C9          2841          RET
D669 2A56F9      2842 RSTCNF   LD      HL,(CURBUF)     ;form Rst Conf pack
D66C 3606        2843          LD      (HL),PLMIN      ;insert length
D66E 23          2844          INC     HL
D66F 23          2845          INC     HL
D670 23          2846          INC     HL
D671 3610        2847          LD      (HL),10H        ;insert ID field
D673 23          2848          INC     HL
D674 23          2849          INC     HL
D675 36FF        2850          LD      (HL),0FFH       ;insert pack code
D677 C9          2851          RET
                 2852 ;
D678 CDF6D5      2853 CLINT    CALL    CLIND           ;form Clear
D67B CDB6D5      2854          CALL    INT             ;transf to POQ
D67E 2E56        2855          LD      L,M86
D680 63          2856          LD      H,E
D681 CDDDDB      2857          CALL    LDMES
D684 CDDCD6      2858          CALL    CLRSN
D687 C9          2859          RET
D688 DD2A56F9    2860 CLINTA   LD      IX,(CURBUF)     ;get Clear reason
D68C DD5606      2861          LD      D,(IX+6)
D68F CDB6D5      2862          CALL    INT             ;transf to POQ
D692 CDDCD6      2863          CALL    CLRSN
D695 C9          2864          RET
D696 CD07D6      2865 CCINT    CALL    CLCONF          ;form Cl Conf
D699 CDB6D5      2866          CALL    INT             ;transf to POQ
D69C 2E58        2867          LD      L,M88
D69E 63          2868          LD      H,E
D69F CDDDDB      2869          CALL    LDMES
D6A2 C9          2870          RET
D6A3 2E5D        2871 CLEXT    LD      L,M93
D6A5 63          2872          LD      H,E
D6A6 CDDDDB      2873          CALL    LDMES
D6A9 CDF6D5      2874          CALL    CLIND           ;form Clear
D6AC CDCDD5      2875          CALL    EXT             ;transf to BOQ
D6AF DD2172FA    2876          LD      IX,CER
D6B3 CDE3D6      2877          CALL    INCRCD
D6B6 C9          2878          RET
D6B7 CD07D6      2879 CCEXT    CALL    CLCONF          ;form Cl Conf
D6BA CDCDD5      2880          CALL    EXT             ;transf to BOQ
```

```
D6BD  2E5F          2881          LD      L,M95
D6BF  63            2882          LD      H,E
D6C0  CDDDDB        2883          CALL    LDMES
D6C3  C9            2884          RET
D6C4  CD58D6        2885 RSTINT   CALL    RSTIND          ;form Restart
D6C7  CDB6D5        2886          CALL    INT             ;transf to POQ
D6CA  2E0F          2887          LD      L,M15
D6CC  2602          2888          LD      H,M02
D6CE  CDDDDB        2889          CALL    LDMES
D6D1  CDDCD6        2890          CALL    CLRSN
D6D4  DD216AFA      2891          LD      IX,DCERS
D6D8  CDE3D6        2892          CALL    INCRCD
D6DB  C9            2893          RET
D6DC  2E61          2894 CLRSN    LD      L,M97           ;reason message
D6DE  62            2895          LD      H,D
D6DF  CDDDDB        2896          CALL    LDMES
D6E2  C9            2897          RET
                    2898 ;
D6E3  DD6600        2899 INCRCD   LD      H,(IX)          ;inc record (IX)
D6E6  DD6E01        2900          LD      L,(IX+1)
D6E9  23            2901          INC     HL
D6EA  DD7400        2902          LD      (IX),H
D6ED  DD7501        2903          LD      (IX+1),L
D6F0  C9            2904          RET
D6F1  1600          2905 MAP      LD      D,00H
D6F3  2110F9        2906          LD      HL,RCI          ;point rem chan table
D6F6  19            2907          ADD     HL,DE           ;offset chan
D6F7  3A4AF9        2908          LD      A,(REMOTE)      ;get rem chan
D6FA  77            2909          LD      (HL),A          ;insert rem chan
D6FB  C9            2910          RET
D6FC  3A4BF9        2911 DECHC    LD      A,(REMOTE+1)    ;dec set-up calls
D6FF  47            2912          LD      B,A
D700  21D0F8        2913          LD      HL,RDCEHC
D703  85            2914          ADD     L
D704  6F            2915          LD      L,A
D705  35            2916          DEC     (HL)
D706  56            2917          LD      D,(HL)
D707  21E0F8        2918          LD      HL,RDCEST       ;point LAU state
D70A  78            2919          LD      A,B
D70B  85            2920          ADD     L
D70C  6F            2921          LD      L,A
D70D  CBA6          2922          RES     TONI,(HL)       ;reset tim (resp recv)
D70F  AF            2923          XOR     A
D710  BA            2924          CP      D
D711  C0            2925          RET     NZ
D712  CB9E          2926          RES     COMI,(HL)       ;no resp expected
D714  C9            2927          RET
D715  3A4BF9        2928 DECNC    LD      A,(REMOTE+1)    ;dec LAUi calls
D718  21C0F8        2929          LD      HL,RDCENC
D71B  85            2930          ADD     L
D71C  6F            2931          LD      L,A
D71D  35            2932          DEC     (HL)
D71E  C9            2933          RET
D71F  3A6CF9        2934 CLPAK    LD      A,(CTQ2)        ;clear PIQ
D722  B7            2935          OR      A
D723  C8            2936          RET     Z
D724  C5            2937          PUSH    BC
D725  47            2938          LD      B,A
D726  216DF9        2939 CLPAK1   LD      HL,DFQ2
D729  7E            2940          LD      A,(HL)
```

```
D72A 34          2941          INC     (HL)
D72B 2120F8      2942          LD      HL,TPQ2         ;get buffer
D72E CD01C4      2943          CALL    GETBUF
D731 CD40C4      2944          CALL    RELBUF          ;release buffer
D734 10F0        2945          DJNZ    CLPAK1          ;repeat all buff
D736 C1          2946          POP     BC
D737 AF          2947          XOR     A
D738 326CF9      2948          LD      (CTQ2),A
D73B C9          2949          RET
                 2950 ;
D73C AF          2951 RST      XOR     A               ;X.25 Rst/process
D73D 2142F9      2952          LD      HL,NRCH         ;test active channels
D740 BE          2953          CP      (HL)
D741 2851        2954          JR      Z,ENDRST
D743 77          2955          LD      (HL),A          ;reset counters
D744 3244F9      2956          LD      (TONC),A
D747 3243F9      2957          LD      (STMC),A
D74A 3245F9      2958          LD      (SPCC),A
D74D 2100F9      2959          LD      HL,CSI          ;point channel table
D750 0607        2960          LD      B,MXCH
D752 3E00        2961          LD      A,P1
D754 23          2962 RST1     INC     HL
D755 77          2963          LD      (HL),A          ;reset channel (p1)
D756 10FC        2964          DJNZ    RST1            ;repeat all channels
D758 CB91        2965          RES     SCN1,C
D75A 21C0F8      2966          LD      HL,RDCENC       ;remote recovery
D75D 060B        2967          LD      B,NDCE
D75F 110000      2968          LD      DE,00
D762 AF          2969 RST2     XOR     A
D763 BE          2970          CP      (HL)            ;test LAUi calls
D764 281C        2971          JR      Z,RST3
D766 3600        2972          LD      (HL),00         ;reset LAUi calls
D768 E5          2973          PUSH    HL
D769 21D0F8      2974          LD      HL,RDCEHC
D76C 19          2975          ADD     HL,DE
D76D 3600        2976          LD      (HL),00
D76F 3A49F9      2977          LD      A,(CLRST)
D772 21F0F8      2978          LD      HL,RDCERS       ;insert reason (Rst)
D775 19          2979          ADD     HL,DE
D776 77          2980          LD      (HL),A
D777 21E0F8      2981          LD      HL,RDCEST
D77A 19          2982          ADD     HL,DE
D77B 3607        2983          LD      (HL),SCONI      ;update LAUi state
D77D 2146F9      2984          LD      HL,BSCC         ;inc LAU scan count
D780 34          2985          INC     (HL)
D781 E1          2986          POP     HL
D782 23          2987 RST3     INC     HL
D783 1C          2988          INC     E
D784 10DC        2989          DJNZ    RST2            ;repeat all LAU's
D786 3A46F9      2990          LD      A,(BSCC)        ;update LAU scan param
D789 B7          2991          OR      A
D78A 2808        2992          JR      Z,ENDRST
D78C CBD9        2993          SET     SCN2,C
D78E 21DFF8      2994          LD      HL,RDCEST-1
D791 225AF9      2995          LD      (SCPT2),HL
D794 C9          2996 ENDRST   RET
D795 3A4BF9      2997 BUSRST   LD      A,(REMCTE+1)    ;LAUi Restart/process
D798 5F          2998          LD      E,A
D799 1600        2999          LD      D,00
D79B E5          3000          PUSH    HL
```

```
D79C  21C0F8      3001          LD      HL,RDCENC
D79F  19          3002          ADD     HL,DE
D7A0  7E          3003          LD      A,(HL)
D7A1  B7          3004          OR      A               ;test LAUi calls
D7A2  CA20D8      3005          JP      Z,BSRST3
D7A5  C5          3006          PUSH    BC
D7A6  47          3007          LD      B,A
D7A7  3600        3008          LD      (HL),00         ;reset LAUi calls
D7A9  21D0F8      3009          LD      HL,RDCEHC
D7AC  19          3010          ADD     HL,DE
D7AD  3600        3011          LD      (HL),00
D7AF  21F0F8      3012          LD      HL,RDCERS
D7B2  19          3013          ADD     HL,DE
D7B3  3A49F9      3014          LD      A,(CLRST)       ;get reason (Rst)
D7B6  77          3015          LD      (HL),A          ;insert reason
D7B7  2108F9      3016          LD      HL,RDCE
D7BA  FD2100F9    3017          LD      IY,CSI          ;scan chan (LAUi calls)
D7BE  3A4BF9      3018 BSRST0   LD      A,(REMOTE+1)
D7C1  23          3019 BSRST1   INC     HL
D7C2  FD23        3020          INC     IY
D7C4  BE          3021          CP      (HL)            ;test LAUi call
D7C5  20FA        3022          JR      NZ,BSRST1
D7C7  E5          3023          PUSH    HL
D7C8  21C0DD      3024          LD      HL,RSTAB        ;point Rst action table
D7CB  FD7E00      3025          LD      A,(IY)          ;get channel state
D7CE  E60F        3026          AND     STAMSK
D7D0  07          3027          RLCA
D7D1  07          3028          RLCA
D7D2  85          3029          ADD     L
D7D3  6F          3030          LD      L,A
D7D4  E9          3031          JP      (HL)            ;jump to 'jump tab'
                  3032 ;
                  3033 ;in the following routines IY points to the
                  3034 ;channel state; each routine updates the channel
                  3035 ;state and counters
                  3036 ;
D7D5  04          3037 RSTA     INC     B
D7D6  1836        3038          JR      BSRST2
D7D8  CD93D5      3039 RSTB     CALL    STOP
D7DB  CDA7D5      3040          CALL    INCSPC
D7DE  FD36008E    3041          LD      (IY),P14
D7E2  182A        3042          JR      BSRST2
D7E4  CD98D5      3043 RSTC     CALL    STOP2
D7E7  CDA7D5      3044          CALL    INCSPC
D7EA  FD36008E    3045          LD      (IY),P14
D7EE  181E        3046          JR      BSRST2
D7F0  FD36008E    3047 RSTD     LD      (IY),P14
D7F4  1818        3048          JR      BSRST2
D7F6  FD360017    3049 RSTE     LD      (IY),P7
D7FA  1812        3050          JR      BSRST2
D7FC  CD98D5      3051 RSTF     CALL    STOP2
D7FF  CDA7D5      3052          CALL    INCSPC
D802  FD36008F    3053          LD      (IY),P15
D806  1806        3054          JR      BSRST2
D808  FD36008B    3055 RSTG     LD      (IY),P11
D80C  1800        3056          JR      BSRST2
D80E  E1          3057 BSRST2   POP     HL
D80F  10AD        3058          DJNZ    BSRST0          ;repeat all chan (LAUi)
D811  C1          3059          POP     BC
D812  3A45F9      3060          LD      A,(SPCC)        ;update ch scan variab
```

```
D815  B7          3061          OR      A
D816  2808        3062          JR      Z,BSRST3
D818  CBD1        3063          SET     SCN1,C
D81A  2108F9      3064          LD      HL,CSI+MXCH+1
D81D  2258F9      3065          LD      (SCPT1),HL
D820  E1          3066 BSRST3   POP     HL
D821  C9          3067          RET
                  3068 ;
D822  3A78F9      3069 BOQOV    LD      A,(CTQ3)        ;BOQ overflow
D825  FE0D        3070          CP      NBUF-NFB0       ;test overfl condition
D827  D8          3071          RET     C
D828  47          3072          LD      B,A             ;overflow/nbr BOQ pack
D829  2179F9      3073 BOQOV1   LD      HL,DFQ3         ;get buffer
D82C  7E          3074          LD      A,(HL)
D82D  34          3075          INC     (HL)
D82E  2140F8      3076          LD      HL,TPQ3
D831  CD01C4      3077          CALL    GETBUF
D834  CD40C4      3078          CALL    RELBUF          ;release buffer
D837  10F0        3079          DJNZ    BOQOV1          ;repeat all buff (BOQ)
D839  AF          3080          XOR     A
D83A  3278F9      3081          LD      (CTQ3),A
D83D  2164F9      3082          LD      HL,NOGB
D840  7E          3083          LD      A,(HL)
D841  3604        3084          LD      (HL),NGB0       ;reassign NOFB/NOGB
D843  D604        3085          SUB     NGB0
D845  2163F9      3086          LD      HL,NOFB
D848  F3          3087          DI
D849  86          3088          ADD     (HL)
D84A  77          3089          LD      (HL),A
D84B  FB          3090          EI
D84C  2140F9      3091          LD      HL,BUFREG
D84F  CB86        3092          RES     BY,(HL)
D851  AF          3093          XOR     A
D852  3246F9      3094          LD      (BSCC),A
D855  CB99        3095          RES     SCN2,C
D857  CDCDC0      3096          CALL    INDCE           ;reset LAU tables
D85A  CB49        3097          BIT     DCERST,C        ;test Rst (X.25 int)
D85C  2012        3098          JR      NZ,BOQOV2
D85E  2149F9      3099          LD      HL,CLRST
D861  3605        3100          LD      (HL),05H
D863  CD3CD7      3101          CALL    RST
D866  CBC9        3102          SET     DCERST,C        ;initiate Rst (X.25)
D868  CD1DC4      3103          CALL    GTPOOL
D86B  1603        3104          LD      D,03
D86D  CDC4D6      3105          CALL    RSTINT          ;form Rst pack
D870  CD9CD4      3106 BOQOV2   CALL    INRST           ;initiate LAU recov
D873  C9          3107          RET
                  3108 ;
                  3109 ;        **********************
                  3110 ;        * MONITOR PROCEDURE *
                  3111 ;        **********************
                  3112 ;
D874  79          3113 MNRQ     LD      A,C             ;save pack state
D875  3241F9      3114          LD      (GENST),A
D878  23          3115          INC     HL              ;point EMU com field
D879  7E          3116          LD      A,(HL)          ;read EMU command
D87A  23          3117          INC     HL              ;point handsh field
D87B  46          3118          LD      B,(HL)          ;read handsh
D87C  23          3119          INC     HL              ;point param2
D87D  5E          3120          LD      E,(HL)          ;read param2
```

```
D87E 23          3121         INC    HL              ;point param3
D87F 56          3122         LD     D,(HL)          ;read param3
D880 ED5380F9    3123         LD     (PARAM),DE      ;save parameters
D884 FE50        3124 MNCMD   CP     A,'P'           ;decode EMU command
D886 CAECD8      3125         JP     Z,MPOLL
D889 FE49        3126         CP     A,'I'
D88B CAF2D8      3127         JP     Z,MINIT
D88E FE53        3128         CP     A,'S'
D890 CA14D9      3129         JP     Z,MSTAT
D893 FE41        3130         CP     A,'A'
D895 CA4BDA      3131         JP     Z,MACTV
D898 FE44        3132         CP     A,'D'
D89A CA8FDA      3133         JP     Z,MDISP
D89D FE54        3134         CP     A,'T'
D89F CAF1DA      3135         JP     Z,MTRAC
D8A2 FE52        3136         CP     A,'R'
D8A4 CA72DB      3137         JP     Z,MRECD
D8A7 3E04        3138 CMREJ   LD     A,M04           ;reject com (unknown)
D8A9 CDB5D8      3139         CALL   CMRES
D8AC C3D5DB      3140         JP     ENDMN
D8AF 3E03        3141 CMEXC   LD     A,M03           ;acc com message
D8B1 CDB5D8      3142         CALL   CMRES
D8B4 C9          3143         RET
D8B5 21A1F9      3144 CMRES   LD     HL,MSBUF+1      ;form short message
D8B8 36FF        3145         LD     (HL),MEOB       ;insert MEOB
D8BA 2B          3146         DEC    HL
D8BB 77          3147         LD     (HL),A
D8BC 0600        3148         LD     B,00            ;handsh=no more
D8BE 0E02        3149         LD     C,02            ;message length
D8C0 CDC4D8      3150         CALL   MNCF            ;format response
D8C3 C9          3151         RET
D8C4 EB          3152 MNCF    EX     DE,HL           ;point message buf
D8C5 2A56F9      3153         LD     HL,(CURBUF)     ;point current buf
D8C8 3E08        3154         LD     A,MCHEAD        ;header length
D8CA 81          3155         ADD    C               ;add to mess length
D8CB 77          3156         LD     (HL),A          ;insert total length
D8CC 23          3157         INC    HL
D8CD 23          3158         INC    HL
D8CE 23          3159         INC    HL
D8CF 23          3160         INC    HL
D8D0 23          3161         INC    HL              ;point pack field
D8D1 36F7        3162         LD     (HL),MNCNF      ;insert pack (MNCNF)
D8D3 23          3163         INC    HL
D8D4 23          3164         INC    HL              ;point handsh field
D8D5 70          3165         LD     (HL),B          ;insert handsh
D8D6 23          3166         INC    HL              ;point message field
D8D7 EB          3167         EX     DE,HL
D8D8 0600        3168         LD     B,00            ;transf from message
D8DA EDB0        3169         LDIR                   ;to current buffer
D8DC 217AF9      3170         LD     HL,DLQ3         ;transfer to BOQ
D8DF 7E          3171         LD     A,(HL)
D8E0 34          3172         INC    (HL)
D8E1 2140F8      3173         LD     HL,TPQ3
D8E4 CD0EC4      3174         CALL   PUTBUF
D8E7 2178F9      3175         LD     HL,CTQ3
D8EA 34          3176         INC    (HL)            ;inc BOQ count
D8EB C9          3177         RET
                 3178 ;
D8EC CDAFD8      3179 MPOLL   CALL   CMEXC           ;Poll command
D8EF C3D5DB      3180         JP     ENDMN
```

```
                        3181 ;
D8F2  CDAFD8            3182 MINIT   CALL    CMEXC           ;Init command
D8F5  CD1FD7            3183         CALL    CLPAK           ;clear PIO
D8F8  2147F9            3184         LD      HL,IRLP
D8FB  CBD6              3185         SET     PD,(HL)         ;signal lev2
D8FD  3A41F9            3186         LD      A,(GENST)       ;read pack state
D900  4F                3187         LD      C,A
D901  CB89              3188         RES     DCERST,C        ;clear Rst
D903  CB81              3189         RES     TONRST,C
D905  2149F9            3190         LD      HL,CLRST        ;clear reason
D908  3611              3191         LD      (HL),11H
D90A  CD3CD7            3192         CALL    RST
D90D  79                3193         LD      A,C             ;save pack state
D90E  3241F9            3194         LD      (GENST),A
D911  C3D5DB            3195         JP      ENDMN
                        3196 ;
D914  CDAFD8            3197 MSTAT   CALL    CMEXC           ;State command
D917  CD20D9            3198         CALL    GETST           ;get state inf
D91A  CD99D9            3199         CALL    CHMAP           ;get chan inf
D91D  C3D5DB            3200         JP      ENDMN
D920  DD2120DF          3201 GETST   LD      IX,STATIT       ;point st mess titles
D924  1181FA            3202         LD      DE,STBUF+1      ;point State buffer
D927  2130F9            3203         LD      HL,LSTATE       ;point link state
D92A  CD79D9            3204         CALL    TABTR
D92D  2140F9            3205         LD      HL,BUFREG       ;point pack state
D930  CD79D9            3206         CALL    TABTR
D933  3E1B              3207         LD      A,M27
D935  12                3208         LD      (DE),A
D936  13                3209         INC     DE
D937  2160F9            3210         LD      HL,QR           ;point Queue variab
D93A  CD7FD9            3211         CALL    TABTR1
D93D  CD7FD9            3212         CALL    TABTR1
D940  CD79D9            3213         CALL    TABTR
D943  CD79D9            3214         CALL    TABTR
D946  CD79D9            3215         CALL    TABTR
D949  CD79D9            3216         CALL    TABTR
D94C  2110DF            3217         LD      HL,DCENB        ;point LAU table
D94F  CD79D9            3218         CALL    TABTR
D952  21C0F8            3219         LD      HL,RDCENC
D955  CD79D9            3220         CALL    TABTR
D958  21D0F8            3221         LD      HL,RDCEHC
D95B  CD79D9            3222         CALL    TABTR
D95E  21E0F8            3223         LD      HL,RDCEST
D961  CD79D9            3224         CALL    TABTR
D964  21F0F8            3225         LD      HL,RDCERS
D967  CD79D9            3226         CALL    TABTR
D96A  3EFF              3227         LD      A,MEOB          ;insert MEOB
D96C  12                3228         LD      (DE),A
D96D  7B                3229         LD      A,E
D96E  E67F              3230         AND     DBL-1
D970  3D                3231         DEC     A
D971  3280FA            3232         LD      (STBUF),A       ;insert mess length
D974  AF                3233         XOR     A
D975  3295F9            3234         LD      (NDBUF),A       ;reset Disp buf count
D978  C9                3235         RET
D979  CD87D9            3236 TABTR   CALL    TABTR0
D97C  EDB0              3237         LDIR
D97E  C9                3238         RET
D97F  CD87D9            3239 TABTR1  CALL    TABTR0
D982  F3                3240         DI
```

```
D983 EDB0        3241          LDIR
D985 FB          3242          EI
D986 C9          3243          RET
D987 DD7E00      3244 TABTR0   LD      A,(IX)          ;get message (tab type)
D98A 12          3245          LD      (DE),A
D98B 13          3246          INC     DE
D98C DD23        3247          INC     IX
D98E DD7E00      3248          LD      A,(IX)
D991 12          3249          LD      (DE),A
D992 13          3250          INC     DE
D993 DD23        3251          INC     IX
D995 0600        3252          LD      B,00
D997 4F          3253          LD      C,A
D998 C9          3254          RET
D999 AF          3255 CHMAP    XOR     A               ;get chan inf
D99A 3290F9      3256          LD      (MFCH),A        ;curr first ch (=0)
D99D 3291F9      3257          LD      (MNBYT),A       ;curr mess length (=0)
D9A0 3E08        3258          LD      A,NBCH          ;nbr of channels
D9A2 2100FB      3259          LD      HL,MBUF1        ;first buffer
D9A5 2284F9      3260 CHMAP1   LD      (MBUF),HL       ;save curr buf
D9A8 23          3261          INC     HL              ;reserve length byte
D9A9 2286F9      3262          LD      (MBPT),HL       ;save buf pointer
D9AC 0618        3263          LD      B,NCHBF         ;max nbr ch/buf
D9AE B8          3264          CP      B
D9AF 3001        3265          JR      NC,CHMAP2
D9B1 47          3266          LD      B,A
D9B2 90          3267 CHMAP2   SUB     B
D9B3 3292F9      3268          LD      (MNCH),A        ;ch to access (total)
D9B6 78          3269          LD      A,B
D9B7 0608        3270 CHMAP3   LD      B,MBL           ;max nbr ch/block
D9B9 B8          3271          CP      B
D9BA 3001        3272          JR      NC,CHMAP4
D9BC 47          3273          LD      B,A
D9BD 3293F9      3274 CHMAP4   LD      (MBCH),A        ;ch to access (buf)
D9C0 78          3275          LD      A,B
D9C1 3294F9      3276          LD      (MCBL),A        ;ch being accessed
D9C4 C602        3277          ADD     02              ;message length
D9C6 07          3278          RLCA
D9C7 07          3279          RLCA
D9C8 2191F9      3280          LD      HL,MNBYT        ;previous length
D9CB 86          3281          ADD     (HL)
D9CC 77          3282          LD      (HL),A          ;curr total length
D9CD 3A90F9      3283          LD      A,(MFCH)        ;curr first chan
D9D0 2A86F9      3284          LD      HL,(MBPT)       ;read buf point
D9D3 36E8        3285          LD      (HL),M104       ;ch nbr message
D9D5 23          3286          INC     HL
D9D6 70          3287          LD      (HL),B          ;nbr param to follow
D9D7 23          3288 CHMAP5   INC     HL
D9D8 77          3289          LD      (HL),A          ;get ch nbrs
D9D9 3C          3290          INC     A
D9DA 10FB        3291          DJNZ    CHMAP5
D9DC 23          3292          INC     HL
D9DD EB          3293          EX      DE,HL           ;buffer pointer
D9DE 2100F9      3294          LD      HL,CSI          ;get ch states
D9E1 3EE9        3295          LD      A,M105
D9E3 CD33DA      3296          CALL    MAPTR1
D9E6 2108F9      3297          LD      HL,RDCE         ;get rem LAU's
D9E9 3EEA        3298          LD      A,M106
D9EB CD21DA      3299          CALL    MAPTR
D9EE 2110F9      3300          LD      HL,RCI          ;get rem channels
```

```
D9F1  3EEB      3301          LD      A,M107
D9F3  CD21DA    3302          CALL    MAPTR
D9F6  EB        3303          EX      DE,HL
D9F7  2286F9    3304          LD      (MBPT),HL       ;save pointer
D9FA  3A94F9    3305          LD      A,(MCBL)
D9FD  2190F9    3306          LD      HL,MFCH
DA00  86        3307          ADD     (HL)
DA01  77        3308          LD      (HL),A          ;update cur first ch
DA02  3A93F9    3309          LD      A,(MBCH)        ;test ch in block
DA05  B7        3310          OR      A
DA06  C2B7D9    3311          JP      NZ,CHMAP3
DA09  2A84F9    3312          LD      HL,(MBUF)       ;last/insert length
DA0C  3A91F9    3313          LD      A,(MNBYT)
DA0F  77        3314          LD      (HL),A
DA10  118000    3315          LD      DE,DBL          ;get next buffer
DA13  19        3316          ADD     HL,DE
DA14  3A92F9    3317          LD      A,(MNCH)
DA17  B7        3318          OR      A               ;test ch to access
DA18  C2A5D9    3319          JP      NZ,CHMAP1
DA1B  2A86F9    3320          LD      HL,(MBPT)       ;end ch inf access
DA1E  36FF      3321          LD      (HL),MEOB       ;insert MEOB
DA20  C9        3322          RET
DA21  12        3323 MAPTR    LD      (DE),A          ;get ch tab inf
DA22  13        3324          INC     DE
DA23  0600      3325          LD      B,00
DA25  3A90F9    3326          LD      A,(MFCH)
DA28  4F        3327          LD      C,A
DA29  09        3328          ADD     HL,BC
DA2A  3A94F9    3329          LD      A,(MCBL)
DA2D  4F        3330          LD      C,A
DA2E  12        3331          LD      (DE),A
DA2F  13        3332          INC     DE
DA30  EDB0      3333          LDIR
DA32  C9        3334          RET
DA33  12        3335 MAPTR1   LD      (DE),A          ;get ch state
DA34  13        3336          INC     DE
DA35  0600      3337          LD      B,00
DA37  3A90F9    3338          LD      A,(MFCH)
DA3A  4F        3339          LD      C,A
DA3B  09        3340          ADD     HL,BC
DA3C  3A94F9    3341          LD      A,(MCBL)
DA3F  47        3342          LD      B,A
DA40  12        3343          LD      (DE),A
DA41  13        3344          INC     DE
DA42  7E        3345 MAPTR2   LD      A,(HL)
DA43  E60F      3346          AND     STAMSK
DA45  12        3347          LD      (DE),A
DA46  23        3348          INC     HL
DA47  13        3349          INC     DE
DA48  10F8      3350          DJNZ    MAPTR2
DA4A  C9        3351          RET
                3352 ;
DA4B  CDAFD8    3353 MACTV    CALL    CMEXC           ;Activity command
DA4E  CD5ADA    3354          CALL    INITRC          ;init trace variab
DA51  3A80F9    3355          LD      A,(PARAM)       ;read options
DA54  CD78DA    3356          CALL    STRTRC          ;init trace timer
DA57  C3D5DB    3357          JP      ENDMN
DA5A  2150F9    3358 INITRC   LD      HL,INTREG       ;init trace funct
DA5D  CBAE      3359          RES     TRACE,(HL)
DA5F  CBB6      3360          RES     ENDTRC,(HL)
```

```
DA61  AF          3361          XOR     A
DA62  3296F9      3362          LD      (TRCN0),A       ;reset counters
DA65  3297F9      3363          LD      (TRCN1),A
DA68  3298F9      3364          LD      (ACTN0),A
DA6B  3299F9      3365          LD      (ACTN1),A
DA6E  2180FB      3366          LD      HL,ACTBUF
DA71  228AF9      3367          LD      (ACTPT),HL      ;reset pointers
DA74  228CF9      3368          LD      (TRCPT),HL
DA77  C9          3369          RET
DA78  060A        3370 STRTRC   LD      B,ACTIM1        ;set trace interval
DA7A  B7          3371          OR      A
DA7B  2004        3372          JR      NZ,STRTR1
DA7D  3E01        3373          LD      A,01            ;default
DA7F  0602        3374          LD      B,02
DA81  219AF9      3375 STRTR1   LD      HL,ACTC1
DA84  70          3376          LD      (HL),B
DA85  219BF9      3377          LD      HL,ACTC2
DA88  77          3378          LD      (HL),A
DA89  2150F9      3379          LD      HL,INTREG
DA8C  CBEE        3380          SET     TRACE,(HL)
DA8E  C9          3381          RET
                  3382 ;
DA8F  CB40        3383 MDISP    BIT     RMORE,B         ;Display command
DA91  202B        3384          JR      NZ,MDISP4
DA93  3A80F9      3385          LD      A,(PARAM)       ;read options
DA96  B7          3386          OR      A
DA97  2806        3387          JR      Z,MDISP1
DA99  CD20D9      3388          CALL    GETST           ;current state
DA9C  CD99D9      3389          CALL    CHMAP           ;current ch inf
DA9F  3A81F9      3390 MDISP1   LD      A,(PARAM+1)     ;test display opt
DAA2  B7          3391          OR      A
DAA3  280B        3392          JR      Z,MDISP2
DAA5  FE01        3393          CP      A,01
DAA7  280E        3394          JR      Z,MDISP3
DAA9  3E02        3395          LD      A,MNBUF+1       ;state+ch inf
DAAB  2180FA      3396          LD      HL,STBUF        ;point state buf
DAAE  182B        3397          JR      MDISP6
DAB0  3E01        3398 MDISP2   LD      A,01            ;state only
DAB2  2180FA      3399          LD      HL,STBUF        ;point state buf
DAB5  1824        3400          JR      MDISP6
DAB7  3E01        3401 MDISP3   LD      A,MNBUF         ;ch inf only
DAB9  2100FB      3402          LD      HL,MBUF1        ;point ch inf buff
DABC  181D        3403          JR      MDISP6
DABE  3A95F9      3404 MDISP4   LD      A,(NDBUF)       ;more request
DAC1  B7          3405          OR      A               ;test buf to display
DAC2  2809        3406          JR      Z,MDISP5
DAC4  2A88F9      3407          LD      HL,(CDBUF)      ;get next buf
DAC7  118000      3408          LD      DE,DBL
DACA  19          3409          ADD     HL,DE
DACB  180E        3410          JR      MDISP6
DACD  21A0F9      3411 MDISP5   LD      HL,MSBUF        ;no more to display
DAD0  36FF        3412          LD      (HL),MEOB       ;insert MEOB
DAD2  0600        3413          LD      B,00            ;handsh field
DAD4  0E01        3414          LD      C,01            ;mess length
DAD6  CDC4D8      3415          CALL    MNCF
DAD9  1813        3416          JR      MDISP8
DADB  2288F9      3417 MDISP6   LD      (CDBUF),HL      ;display one buf
DADE  4E          3418          LD      C,(HL)          ;mess length
DADF  3D          3419          DEC     A
DAE0  0601        3420          LD      B,01            ;handsh=more
```

```
DAE2  3295F9        3421          LD      (NDBUF),A       ;buf to display
DAE5  2003          3422          JR      NZ,MDISP7
DAE7  0C            3423          INC     C               ;to iclude MEOB
DAE8  0600          3424          LD      B,00            ;handsh=no more
DAEA  23            3425 MDISP7   INC     HL
DAEB  CDC4D8        3426          CALL    MNCF
DAEE  C3D5DB        3427 MDISP8   JP      ENDMN
                    3428 ;
DAF1  CB40          3429 MTRAC    BIT     RMORE,B         ;Trace command
DAF3  2023          3430          JR      NZ,MTRAC2
DAF5  3A80F9        3431          LD      A,(PARAM)       ;read options
DAF8  B7            3432          OR      A
DAF9  280B          3433          JR      Z,MTRAC1
DAFB  CD5ADA        3434          CALL    INITRC          ;current/init trace
DAFE  3A81F9        3435          LD      A,(PARAM+1)     ;read time interval
DB01  CD78DA        3436          CALL    STRTRC          ;set trace interval
DB04  1812          3437          JR      MTRAC2
DB06  3A98F9        3438 MTRAC1   LD      A,(ACTN0)       ;stored
DB09  3296F9        3439          LD      (TRCN0),A       ;update trace count
DB0C  3A99F9        3440          LD      A,(ACTN1)
DB0F  3297F9        3441          LD      (TRCN1),A
DB12  2180FB        3442          LD      HL,ACTBUF       ;reset trace point
DB15  228CF9        3443          LD      (TRCPT),HL
DB18  AF            3444 MTRAC2   XOR     A
DB19  2197F9        3445          LD      HL,TRCN1
DB1C  BE            3446          CP      (HL)            ;test mess length
DB1D  2026          3447          JR      NZ,MTRAC6
DB1F  2196F9        3448          LD      HL,TRCN0        ;<one buf
DB22  BE            3449          CP      (HL)
DB23  201C          3450          JR      NZ,MTRAC5
DB25  2150F9        3451          LD      HL,INTREG       ;buff empty
DB28  CB6E          3452          BIT     TRACE,(HL)      ;test if tracing
DB2A  2806          3453          JR      Z,MTRAC3
DB2C  3E02          3454          LD      A,M02           ;tracing/blank mess
DB2E  0600          3455          LD      B,00            ;handsh=no more
DB30  1804          3456          JR      MTRAC4
DB32  3EFF          3457 MTRAC3   LD      A,MEOB          ;trace ended/MEOB mess
DB34  0602          3458          LD      B,02            ;handsh=end trace
DB36  0E01          3459 MTRAC4   LD      C,01            ;form short message
DB38  21A0F9        3460          LD      HL,MSBUF
DB3B  77            3461          LD      (HL),A
DB3C  CDC4D8        3462          CALL    MNCF
DB3F  182E          3463          JR      ENDMTR
DB41  4E            3464 MTRAC5   LD      C,(HL)          ;buf not empty/length
DB42  77            3465          LD      (HL),A          ;update length (=0)
DB43  1809          3466          JR      MTRAC7
DB45  3E80          3467 MTRAC6   LD      A,DBL           ;>=one full buffer
DB47  4F            3468          LD      C,A             ;length=one buff
DB48  35            3469          DEC     (HL)            ;update length
DB49  2003          3470          JR      NZ,MTRAC7
DB4B  3A96F9        3471          LD      A,(TRCN0)
DB4E  2A8CF9        3472 MTRAC7   LD      HL,(TRCPT)      ;current trace point
DB51  54            3473          LD      D,H             ;save (DE)
DB52  5D            3474          LD      E,L
DB53  0600          3475          LD      B,00
DB55  09            3476          ADD     HL,BC
DB56  228CF9        3477          LD      (TRCPT),HL      ;update trace point
DB59  EB            3478          EX      DE,HL
DB5A  0601          3479          LD      B,01            ;handsh=more
DB5C  B7            3480          OR      A               ;test if last buf
```

```
DB5D  200D        3481          JR      NZ,MTRAC9
DB5F  3A50F9      3482          LD      A,(INTREG)      ;current last mess
DB62  CB6F        3483          BIT     TRACE,A         ;test tracing
DB64  2804        3484          JR      Z,MTRAC8
DB66  0600        3485          LD      B,00            ;trac/handsh=no more
DB68  18C2        3486          JR      MTRAC9
DB6A  0602        3487 MTRAC8   LD      B,02            ;handsh=end trace
DB6C  CDC4D8      3488 MTRAC9   CALL    MNCF
DB6F  C3D5DB      3489 ENDMTR   JP      ENDMN
                  3490 ;
DB72  DD21B0F9    3491 MRECD    LD      IX,TRCDP        ;Record com
DB76  FD2140FA    3492          LD      IY,CRCDP
DB7A  061C        3493          LD      B,NRECD
DB7C  DD6600      3494 MRECD1   LD      H,(IX+0)        ;update total records
DB7F  DD6E01      3495          LD      L,(IX+1)
DB82  FD5600      3496          LD      D,(IY+0)
DB85  FD5E01      3497          LD      E,(IY+1)
DB88  19          3498          ADD     HL,DE
DB89  DD7400      3499          LD      (IX+0),H
DB8C  DD7501      3500          LD      (IX+1),L
DB8F  DD23        3501          INC     IX
DB91  DD23        3502          INC     IX
DB93  FD23        3503          INC     IY
DB95  FD23        3504          INC     IY
DB97  10E3        3505          DJNZ    MRECD1          ;repeat for all rec
DB99  3A80F9      3506          LD      A,(PARAM)       ;test option
DB9C  B7          3507          OR      A
DB9D  2805        3508          JR      Z,MRECD2
DB9F  21B0F9      3509          LD      HL,TRCDP        ;total/point tot rec
DBA2  18C3        3510          JR      MRECD3
DBA4  2140FA      3511 MRECD2   LD      HL,CRCDP        ;curr/point curr rec
DBA7  DD213ADF    3512 MRECD3   LD      IX,RCDTIT       ;point Rec mess
DBAB  1100FA      3513          LD      DE,RCDBUF       ;point Rec buffer
DBAE  3E1C        3514          LD      A,M28           ;insert title
DBB0  12          3515          LD      (DE),A
DBB1  13          3516          INC     DE
DBB2  0611        3517          LD      B,DRECD+SRECD   ;nbr lines
DBB4  C5          3518 MRECD4   PUSH    BC              ;load Record buf
DBB5  CD79D9      3519          CALL    TABTR
DBB8  C1          3520          POP     BC
DBB9  10F9        3521          DJNZ    MRECD4          ;repeat (all records)
DBBB  3EFF        3522          LD      A,MEOB
DBBD  12          3523          LD      (DE),A          ;insert MEOB
DBBE  7B          3524          LD      A,E
DBBF  E67F        3525          AND     DBL-1
DBC1  4F          3526          LD      C,A
DBC2  0C          3527          INC     C               ;message length
DBC3  0600        3528          LD      B,00            ;handsh=no more
DBC5  2100FA      3529          LD      HL,RCDBUF       ;point Record buffer
DBC8  CDC4D8      3530          CALL    MNCF
DBCB  2140FA      3531          LD      HL,CRCDP        ;point curr rec
DBCE  AF          3532          XOR     A               ;reset curr rec
DBCF  CD7AC1      3533          CALL    INRCD0
DBD2  C3D5DB      3534          JP      ENDMN
DBD5  3A41F9      3535 ENDMN    LD      A,(GENST)       ;end EMU com
DBD8  4F          3536          LD      C,A             ;read pack state
DBD9  CD22D8      3537          CALL    BOQOV           ;test BOQ overflow
DBDC  C9          3538          RET
                  3539 ;
DBDD  2282F9      3540 LDMES    LD      (MESG),HL       ;save message
```

```
DBE0  2150F9      3541          LD      HL,INTREG
DBE3  CB6E        3542          BIT     TRACE,(HL)      ;test if tracing
DBE5  2847        3543          JR      Z,ENDMES
DBE7  F5          3544          PUSH    AF              ;tracing
DBE8  3A96F9      3545          LD      A,(TRCN0)
DBEB  C602        3546          ADD     02
DBED  FE80        3547          CP      DBL             ;test trace buff
DBEF  2005        3548          JR      NZ,LDMES1
DBF1  AF          3549          XOR     A               ;full
DBF2  2197F9      3550          LD      HL,TRCN1
DBF5  34          3551          INC     (HL)            ;inc buff count
DBF6  3296F9      3552 LDMES1   LD      (TRCN0),A       ;update trace count
DBF9  3A98F9      3553          LD      A,(ACTN0)
DBFC  C602        3554          ADD     02
DBFE  FE80        3555          CP      DBL             ;test activity buf
DC00  2018        3556          JR      NZ,LDMES3
DC02  2199F9      3557          LD      HL,ACTN1        ;full
DC05  34          3558          INC     (HL)            ;inc buff count
DC06  3E06        3559          LD      A,NBACT
DC08  BE          3560          CP      (HL)            ;test overflow
DC09  200E        3561          JR      NZ,LDMES2
DC0B  2E05        3562          LD      L,M05           ;act buf overflow
DC0D  26FF        3563          LD      H,MEOB          ;overfl/MEOB mess
DC0F  2282F9      3564          LD      (MESG),HL
DC12  2150F9      3565          LD      HL,INTREG
DC15  CBAE        3566          RES     TRACE,(HL)      ;end trace
DC17  CBB6        3567          RES     ENDTRC,(HL)
DC19  AF          3568 LDMES2   XOR     A
DC1A  3298F9      3569 LDMES3   LD      (ACTN0),A       ;update act count
DC1D  2A8AF9      3570          LD      HL,(ACTPT)      ;read act point
DC20  3A82F9      3571          LD      A,(MESG)        ;get mess (two bytes)
DC23  77          3572          LD      (HL),A          ;load message
DC24  23          3573          INC     HL
DC25  3A83F9      3574          LD      A,(MESG+1)
DC28  77          3575          LD      (HL),A
DC29  23          3576          INC     HL
DC2A  228AF9      3577          LD      (ACTPT),HL      ;save pointer
DC2D  F1          3578          POP     AF
DC2E  C9          3579 ENDMES   RET
DC2F  2150F9      3580 MONTM    LD      HL,INTREG       ;monitor timer
DC32  CB76        3581          BIT     ENDTRC,(HL)     ;test end trace
DC34  C8          3582          RET     Z
DC35  CBAE        3583          RES     TRACE,(HL)      ;end trace
DC37  CBB6        3584          RES     ENDTRC,(HL)
DC39  2A8AF9      3585          LD      HL,(ACTPT)      ;read act point
DC3C  36FF        3586          LD      (HL),MEOB       ;insert MEOB
DC3E  2198F9      3587          LD      HL,ACTN0        ;update counters
DC41  34          3588          INC     (HL)
DC42  2196F9      3589          LD      HL,TRCN0
DC45  34          3590          INC     (HL)
DC46  C9          3591          RET
                  3592 ;
                  3593 ;
                  3594 ;        ************************
                  3595 ;        * BUS TRANSFER PROCESS *
                  3596 ;        ************************
                  3597 ;
                  3598 ;
DC47  CD4EDC      3599 BUS      CALL    RBUS            ;bus scheduler
DC4A  CD82DC      3600          CALL    TBUS
```

```
DC4D  C9          3601          RET
                  3602 ;
                  3603 ;        ******************
                  3604 ;        * BUS RECEPTION *
                  3605 ;        ******************
                  3606 ;
DC4E  DBBE        3607 RBUS     IN      A,(PIODB)       ;rcv/read PIO reg
DC50  CB77        3608          BIT     DBUF,A          ;test dest buf
DC52  282D        3609          JR      Z,RBUS0
DC54  3A64F9      3610          LD      A,(NOGB)        ;buf full
DC57  B7          3611          OR      A               ;test buf available
DC58  2827        3612          JR      Z,RBUS0
DC5A  CD1DC4      3613          CALL    GTPOOL          ;get buffer
DC5D  0EBC        3614          LD      C,PIODA
DC5F  ED40        3615          IN      B,(C)           ;dummy read
DC61  ED40        3616          IN      B,(C)           ;read length byte
DC63  70          3617          LD      (HL),B          ;insert length byte
DC64  05          3618          DEC     B
DC65  23          3619          INC     HL
DC66  EDB2        3620          INIR                    ;pack transfer
DC68  DBBE        3621          IN      A,(PIODB)
DC6A  F620        3622          OR      PMASK3          ;pack read pulse
DC6C  D3BE        3623          OUT     (PIODB),A
DC6E  E6DF        3624          AND     PMASK4
DC70  D3BE        3625          OUT     (PIODB),A
DC72  2177F9      3626          LD      HL,DLQ4         ;transf to BIQ
DC75  7E          3627          LD      A,(HL)
DC76  34          3628          INC     (HL)
DC77  2160F8      3629          LD      HL,TPQ4
DC7A  CD0EC4      3630          CALL    PUTBUF
DC7D  2175F9      3631          LD      HL,CTQ4         ;inc BIQ count
DC80  34          3632          INC     (HL)
DC81  C9          3633 RBUS0    RET
                  3634 ;
                  3635 ;        **********************
                  3636 ;        * BUS TRANSMISSION *
                  3637 ;        **********************
                  3638 ;
DC82  AF          3639 TBUS     XOR     A               ;bus transm
DC83  2178F9      3640          LD      HL,CTQ3
DC86  BE          3641          CP      (HL)            ;test pack transm
DC87  282C        3642          JR      Z,TBUS0
DC89  DBBE        3643          IN      A,(PIODB)       ;read PIO reg
DC8B  CB7F        3644          BIT     SBUF,A          ;test source buf
DC8D  2826        3645          JR      Z,TBUS0
DC8F  35          3646          DEC     (HL)            ;buf empty
DC90  2179F9      3647          LD      HL,DFQ3         ;get pack (BOQ)
DC93  7E          3648          LD      A,(HL)
DC94  34          3649          INC     (HL)
DC95  2140F8      3650          LD      HL,TPQ3
DC98  CD01C4      3651          CALL    GETBUF
DC9B  46          3652          LD      B,(HL)          ;read length
DC9C  0EBC        3653          LD      C,PIODA
DC9E  23          3654          INC     HL
DC9F  56          3655          LD      D,(HL)          ;read dest address
DCA0  3A4CF9      3656          LD      A,(LOCADD)
DCA3  77          3657          LD      (HL),A          ;insert local addr
DCA4  DBBE        3658          IN      A,(PIODB)
DCA6  E6E0        3659          AND     PMASK1
DCA8  B2          3660          OR      D
```

```
DCA9  D3BE        3661          OUT     (PIODB),A       ;write dest addr (PIO)
DCAB  2B          3662          DEC     HL
DCAC  EDB3        3663          OTIR                    ;pack transf
DCAE  F610        3664          OR      PMASK2
DCB0  D3BE        3665          OUT     (PIODB),A       ;pack write pulse
DCB2  CD40C4      3666          CALL    RELBUF
DCB5  C9          3667 TBUS0    RET
                  3668 ;
                  3669 ;
                  3670 ;        **************
                  3671 ;        * ROM TABLES *
                  3672 ;        **************
                  3673 ;
                  3674 ;
DCB6  (004A)      3675 ROM      DS      [256D-[ROM MOD 256D]]MOD 256D
DD00  1BCE        3676 PKTAB    DW      R1,R26,R10,R6,R10,R13,R13,R26
      81D0
      86CF
      3BCF
      86CF
      ADCF
      ADCF
      81D0
DD10  ADCF        3677          DW      R13,R16,R13,R4,R21,R23,R19,R19
      DDCF
      ADCF
      19CF
      32D0
      54D0
      10D0
      10D0
DD20  01CF        3678          DW      R2,R26,R10,R7,R10,R13,R13,R26
      81D0
      86CF
      4DCF
      86CF
      ADCF
      ADCF
      81D0
DD30  ADCF        3679          DW      R13,R16,R13,R4,R21,R23,R19,R19
      DDCF
      ADCF
      19CF
      32D0
      54D0
      10D0
      10D0
DD40  13CF        3680          DW      R3,R26,R11,R8,R11,R14,R14,R15
      81D0
      92CF
      64CF
      92CF
      BDCF
      BDCF
      CDCF
DD50  BDCF        3681          DW      R14,R17,R14,R5,R22,R24,R20,R20
      EACF
      BDCF
      2BCF
      44D0
```

```
        61D0
        22D0
        22D0
DD60    01CF            3682            DW      R2,R26,R10,R9,R10,R13,R13,R15
        81D0
        86CF
        74CF
        86CF
        ADCF
        ADCF
        CDCF
DD70    ADCF            3683            DW      R13,R18,R13,R5,R21,R25,R19,R19
        FDCF
        ADCF
        2BCF
        32D0
        71D0
        10D0
        10D0
DD80    01CF            3684            DW      R2,R26,R10,R9,R12,R13,R13,R26
        81D0
        86CF
        74CF
        9CCF
        ADCF
        ADCF
        81D0
DD90    ADCF            3685            DW      R13,R16,R13,R4,R21,R23,R19,R19
        DDCF
        ADCF
        19CF
        32D0
        54D0
        10D0
        10D0
DDA0    01CF            3686            DW      R2,R26,R10,R9,R10,R13,R13,R26
        81D0
        86CF
        74CF
        86CF
        ADCF
        ADCF
        81D0
DDB0    ADCF            3687            DW      R13,R16,R13,R4,R21,R23,R19,R19
        DDCF
        ADCF
        19CF
        32D0
        54D0
        10D0
        10D0
DDC0    C3              3688 RSTAB      DB      0C3H
DDC1    D5D7            3689            DW      RSTA,0C300H,RSTA,0C300H
        00C3
        D5D7
        00C3
DDC9    E4D7            3690            DW      RSTC,0C300H,RSTB,0C300H
        00C3
        D8D7
        00C3
```

```
DDD1  E4D7        3691          DW    RSTC,0C300H,RSTC,0C300H
      00C3
      E4D7
      00C3
DDD9  FCD7        3692          DW    RSTF,0C300H,RSTA,0C300H
      00C3
      D5D7
      00C3
DDE1  E4D7        3693          DW    RSTC,0C300H,RSTE,0C300H
      00C3
      F6D7
      00C3
DDE9  E4D7        3694          DW    RSTC,0C300H,RSTA,0C300H
      00C3
      D5D7
      00C3
DDF1  F0D7        3695          DW    RSTD,0C300H,RSTG,0C300H
      00C3
      08D8
      00C3
DDF9  D5D7        3696          DW    RSTA,0C300H,RSTA
      00C3
      D5D7
DDFF  00          3697          DB    00H
DE00  20D2        3698 PKTAB1   DW    RB1,RB1,RB2,RB1,RB1,RB1,RB1,RB1
      20D2
      26D2
      20D2
      20D2
      20D2
      20D2
      20D2
DE10  20D2        3699          DW    RB1,RB1,RB1,RB1,RB1,RB1,RB1,RB1
      20D2
      20D2
      20D2
      20D2
      20D2
      20D2
      20D2
DE20  20D2        3700          DW    RB1,RB1,RB4,RB3,RB4,RB6,RB7,RB1
      20D2
      47D2
      37D2
      47D2
      68D2
      87D2
      20D2
DE30  ADD2        3701          DW    RB9,RB1,RB1,RB1,RB16,RB1,RB1,RB1
      20D2
      20D2
      20D2
      49D3
      20D2
      20D2
      20D2
DE40  20D2        3702          DW    RB1,RB1,RB1,RB1,RB1,RB17,RB8,RB1
      20D2
      20D2
      20D2
```

```
        20D2
        59D3
        9AD2
        20D2
DE50    6ED3            3703            DW      RB18,RB1,RB1,RB1,RB1,RB1,RB1,RB1
        20D2
        20D2
        20D2
        20D2
        20D2
        20D2
        20D2
DE60    20D2            3704            DW      RB1,RB1,RB1,RB1,RB5,RB1,RB1,RB1
        20D2
        20D2
        20D2
        57D2
        20D2
        20D2
        20D2
DE70    20D2            3705            DW      RB1,RB1,RB1,RB1,RB1,RB1,RB1,RB1
        20D2
        20D2
        20D2
        20D2
        20D2
        20D2
        20D2
DE80    20D2            3706            DW      RB1,RB1,RB1,RB1,RB1,RB1,RB11,RB1
        20D2
        20D2
        20D2
        20D2
        20D2
        DCD2
        20D2
DE90    EDD2            3707            DW      RB12,RB1,RB10,RB1,RB1,RB1,RB1,RB1
        20D2
        CCD2
        20D2
        20D2
        20D2
        20D2
        20D2
DEA0    20D2            3708            DW      RB1,RB1,RB1,RB1,RB1,RB1,RB14,RB1
        20D2
        20D2
        20D2
        20D2
        20D2
        1BD3
        20D2
DEB0    31D3            3709            DW      RB15,RB1,RB13,RB1,RB1,RB1,RB1,RB1
        20D2
        FED2
        20D2
        20D2
        20D2
        20D2
        20D2
```

```
DEC0  C3          3710 SCNTAB  DB    0C3H
DEC1  77D5        3711         DW    SCNF,0C300H,SCNF,0C300H
      00C3
      77D5
      00C3
DEC9  77D5        3712         DW    SCNF,0C300H,SCNF,0C300H
      00C3
      77D5
      00C3
DED1  77D5        3713         DW    SCNF,0C300H,SCNF,0C300H
      00C3
      77D5
      00C3
DED9  77D5        3714         DW    SCNF,0C300H,SCNF,0C300H
      00C3
      77D5
      00C3
DEE1  77D5        3715         DW    SCNF,0C300H,SCNF,0C300H
      00C3
      77D5
      00C3
DEE9  77D5        3716         DW    SCNF,0C300H,SCNE,0C300H
      00C3
      6AD5
      00C3
DEF1  4DD5        3717         DW    SCNC,0C300H,SCNA,0C300H
      00C3
      3CD5
      00C3
DEF9  57D5        3718         DW    SCND,0C300H,SCNB
      00C3
      42D5
DEFF  00          3719         DB    00H
                  3720 ;
DF00  (0000)      3721 ROM1    DS    [10H-[ROM1 MOD 10H]]MOD 10H
DF00  01010000    3722 ADTAB   DB    ADDSX,01,00,00,00,00,00,00
      00000000
DF08  00000000    3723         DB    00,00,0C,00,00,00,00,00
      00000000
DF10  00010203    3724 DCENB   DB    00,01,02,03,04,05,06,07
      04050607
DF18  08090A0B    3725         DB    08,09,10,11,12,13,14,15
      0C0D0E0F
DF20  EC08ED07    3726 STATIT  DB    M100,08,M109,07,M110,06,M111,06
      EE06EF06
DF28  F003F106    3727         DB    M112,03,M113,06,M114,03,M115,03
      F203F303
DF30  F40BF50B    3728         DB    M116,NDCE,M117,NDCE,M118,NDCE
      F60B
DF36  F70BF80B    3729         DB    M119,NDCE,M120,NDCE
DF3A  A004A104    3730 RCDTIT  DB    M32,04,M33,04,M34,04,M35,04
      A204A304
DF42  A404A504    3731         DB    M36,04,M37,04,M38,04,M39,04
      A604A704
DF4A  A804A904    3732         DB    M40,04,M41,04,M42,04,M43,02
      AA04AB02
DF52  AC02AD02    3733         DB    M44,02,M45,02,M46,02,M47,02
      AE02AF02
DF5A  B002        3734         DB    M48,02
DF5C  00FC        3735 BUFTAB  DW    BUF1,BUF2,BUF3,BUF4
```

```
        88F0
        10F1
        98F1
DF64    20F2            3736            DW      BUF5,BUF6,BUF7,BUF8
        A8F2
        30F3
        B8F3
DF6C    40F4            3737            DW      BUF9,BUF10,BUF11,BUF12
        C8F4
        50F5
        D8F5
DF74    60F6            3738            DW      BUF13,BUF14,BUF15
        E8F6
        70F7
        (0040)          3739 INTMOD     EQU     40H
                        3740 ;
DF7A    (0006)          3741 ROM2       DS      [INTMOD-[ROM2 MOD INTMOD]]MOD INTMOD
DF80    (0000)          3742 INTAB      DS      00H
DF80    89C1            3743 SIOINT     DW      TRB,EXTB,RECB,SRB
        C0C1
        0CC2
        0DC2
DF88    18C2            3744            DW      TRA,EXTA,RECA,SRA
        25C2
        68C2
        D6C2
DF90    57C3            3745 TIMINT     DW      ENDINT,ENDINT,ENDINT,TIMER
        57C3
        57C3
        4EC3
DF98    8F07CFC0        3746 PIOTAB     DB      8FH,07H,0CFH,0C0H,07H,00H
        0700
DF9E    150E9003        3747 TIMTAB     DB      15H,14D,TVEC,03H,15H,85D,0D5H,163D
        1555D5A3
DFA6    14201119        3748 SIOTAB     DB      14H,20H,11H,19H,05H,00H,07H,7EH
        0500077E
DFAE    43D9            3749            DB      43H,0D9H
DFB0    12801420        3750            DB      12H,SVEC,14H,20H,11H,07H,05H,69H
        11070569
DFB8    877E            3751            DB      87H,7EH
                        3752 ;
                        3753 ;          ************************************************
                        3754 ;          * PERIPHERAL ADDRESSES AND INTERRUPT VECTORS *
                        3755 ;          ************************************************
                        3756 ;
        (00DF)          3757 IVEC       EQU     INTAB SHR 8
        (0080)          3758 SVEC       EQU     SIOINT AND 00FFH
        (0090)          3759 TVEC       EQU     TIMINT AND 00FFH
        (00BC)          3760 PIODA      EQU     0BCH
        (00BD)          3761 PIOCA      EQU     PIODA+1
        (00BE)          3762 PIODB      EQU     PIODA+2
        (00BF)          3763 PIOCB      EQU     PIODA+3
        (007C)          3764 SIODA      EQU     7CH
        (007D)          3765 SIOCA      EQU     SIODA+1
        (007E)          3766 SIODB      EQU     SIODA+2
        (007F)          3767 SIOCB      EQU     SIODA+3
        (00DC)          3768 CTC0       EQU     0DCH
        (00DD)          3769 CTC1       EQU     CTC0+1
        (00DE)          3770 CTC2       EQU     CTC0+2
        (00DF)          3771 CTC3       EQU     CTC0+3
```

```
                3772 ;
                3773 ;
                3774 ;       ********************************************
                3775 ;       * RAM TABLES/BUFFERS AND SYSTEM PARAMETERS *
                3776 ;       ********************************************
                3777 ;
                3778 ;
(F000)          3779 RAMTAB  EQU     0F000H          ;RAM area
(FF00)          3780 STACK   EQU     RAMTAB+0F000H   ;Top of Stack
(0001)          3781 TRESAD  EQU     01H             ;frame address field
(0003)          3782 TCOMAD  EQU     03H
(0003)          3783 RRESAD  EQU     03H
(0001)          3784 RCOMAD  EQU     01H
(0003)          3785 RETMAX  EQU     03H             ;ret count limits
(0003)          3786 RIMAX   EQU     03H
(0007)          3787 WINDOW  EQU     07H             ;window size
(0002)          3788 NFB0    EQU     02H             ;nbr buf (busy)/input
(0004)          3789 NGB0    EQU     04H             ;transit
(00F3)          3790 MNREQ   EQU     0F3H            ;internal pack types
(00F7)          3791 MNCNF   EQU     0F7H
(0087)          3792 HDCF    EQU     87H
(0083)          3793 HDRJ    EQU     83H
(001F)          3794 MNDCE   EQU     16              ;max number LAU's
(000B)          3795 NDCE    EQU     11D             ;number of LAU's
(0000)          3796 ASX0    EQU     00              ;address hierarchy
(0000)          3797 ASX1    EQU     00
(0001)          3798 ASX2    EQU     01
(0064)          3799 TIME    EQU     100D            ;100 ms (basic time)
(0001)          3800 LLTIME  EQU     001D            ;1 s (link )
(0001)          3801 PLTIME  EQU     001D            ;1 s
(000F)          3802 PLTIMA  EQU     015D            ;15 s (pack X.25 int)
(0001)          3803 PLTIMB  EQU     001D            ;1 s (pack LAU int)
(001F)          3804 PLTIMC  EQU     031D            ;31 s (pack recovery)

(0001)          3805 TRCTIM  EQU     001D            ;1 s (trace)
(000A)          3806 EXTDCE  EQU     0AH             ;IGI address
(0001)          3807 HEAD1   EQU     01H             ;packet field sizes
(0002)          3808 HEAD2   EQU     02H
(0003)          3809 HEAD3   EQU     03H
(0008)          3810 MCHEAD  EQU     08H
(0002)          3811 CRC     EQU     02H
(0080)          3812 DATAL   EQU     80H
(0010)          3813 CUD     EQU     10H
(0008)          3814 MBL     EQU     08H             ;chan/ block display
(000F)          3815 NBUF    EQU     0FH             ;number pool buffers
(0006)          3816 NBACT   EQU     06H             ;nbr Activity buf
(0010)          3817 QE      EQU     10H             ;queue size
(0008)          3818 SDBUFL  EQU     08H             ;SARM/DISC buf length
(0008)          3819 NBCH    EQU     08H             ;nbr logical channels
(0010)          3820 CHMOD   EQU     10H
(0020)          3821 MNRCD   EQU     32              ;max nbr records
(000B)          3822 DRECD   EQU     11              ;double records
(0006)          3823 SRECD   EQU     06              ;single records
(000D)          3824 CR      EQU     13              ;carriage return
(000A)          3825 LNF     EQU     10              ;line feed
(0003)          3826 ETX     EQU     03              ;end of text
(00FF)          3827 MEOB    EQU     0FFH            ;message end of block
                3828 ;
(0003)          3829 FLMIN   EQU     HEAD1+HEAD2     ;min frame length
(0006)          3830 PLMIN   EQU     FLMIN+HEAD3     ;min pack length
```

```
(0008)      3831 NONDAT  EQU   PLMIN+CRC          ;non data fields
(0083)      3832 ILMAX   EQU   DATAL+HEAD3        ;max INF frame length
(0088)      3833 BUFL    EQU   DATAL+NONDAT       ;buffer length
(008C)      3834 DBL     EQU   DATAL              ;data length (MNCNF)
(0020)      3835 QL      EQU   2*QE
(000F)      3836 QMASK   EQU   QE-1
(000F)      3837 CHMASK  EQU   CHMOD-1
(000F)      3838 DCEMSK  EQU   MNDCE-1
(0007)      3839 MXCH    EQU   NBCH-1             ;max channel number
(0001)      3840 ADDSX   EQU   ASX0+ASX1+ASX2
(0006)      3841 N0      EQU   NFB0+NGB0          ;buf pool thresholds
(0008)      3842 N1      EQU   N0+2
(000A)      3843 TIME2   EQU   LLTIME*1000/TIME;timers in 100ms units
(000A)      3844 TIME3   EQU   PLTIME*1000/TIME
(000F)      3845 TIME3A  EQU   PLTIMA/PLTIME
(0001)      3846 TIME3B  EQU   PLTIMB/PLTIME
(001F)      3847 TIME3C  EQU   PLTIMC/PLTIME
(000A)      3848 ACTIM1  EQU   TRCTIM*1000/TIME
(0028)      3849 MBML    EQU   [MBL+2]*4          ;monitor parameters
(001C)      3850 NRECD   EQU   2*DRECD+SRECD
(0003)      3851 NBLBF   EQU   [DBL-1]/MBML
(0018)      3852 NCHBF   EQU   MBL*NBLBF
(0001)      3853 MNBUF   EQU   [NBCH+[NCHBF-NBCH]MOD NCHBF]/NCHBF
            3854 ;
(0001)      3855 LD      EQU   01H                ;L2/L3 interface flags
(0002)      3856 PD      EQU   02H
            3857 ;
(0000)      3858 T       EQU   00H                ;transm/rcv flags
(0001)      3859 ECM     EQU   01H
(0006)      3860 UND     EQU   06H
(000C)      3861 RC      EQU   0CH
(0004)      3862 HT      EQU   04H
(0007)      3863 AB      EQU   07H
(0010)      3864 INRCA   EQU   10H
(0040)      3865 INTRB   EQU   40H
(0003)      3866 TRABT   EQU   03H
(0001)      3867 TRFRM   EQU   01H
            3868 ;
(0000)      3869 T2OUT   EQU   00H                ;timer flags
(0001)      3870 T2ON    EQU   01H
(0002)      3871 T3OUT   EQU   02H
(0003)      3872 TBOUT   EQU   03H
(0004)      3873 TBOUT1  EQU   04H
(0005)      3874 TRACE   EQU   05H
(0006)      3875 ENDTRC  EQU   06H
(0007)      3876 MTIM    EQU   07H
            3877 ;
(0000)      3878 BY      EQU   00H                ;busy flag
            3879 ;
(0000)      3880 LF      EQU   00H                ;link state flags
(0001)      3881 PRIM    EQU   01H
(0002)      3882 DIS     EQU   02H
(0003)      3883 SEC     EQU   03H
(0004)      3884 SEC0    EQU   04H
(0005)      3885 SEC1    EQU   05H
(0006)      3886 SEC2    EQU   06H
            3887 ;
(0000)      3888 COM     EQU   00H
(0001)      3889 SD      EQU   01H
(0002)      3890 IAL     EQU   02H
```

```
(0003)        3891 WAIT    EQU     03H
(0004)        3892 REC     EQU     04H
(0005)        3893 PS      EQU     05H
(0006)        3894 RBY     EQU     06H
(0007)        3895 PGB     EQU     07H
(0100)        3896 INLST   EQU     0100H
              3897 ;
(0000)        3898 TONRST  EQU     00H            ;pack state flags
(0001)        3899 DCERST  EQU     01H
(0002)        3900 SCN1    EQU     02H
(0003)        3901 SCN2    EQU     03H
(0004)        3902 BF      EQU     04H
              3903 ;
(0004)        3904 CHTIM1  EQU     04H            ;channel state flags
(0005)        3905 CHTIM2  EQU     05H
(0006)        3906 CHTON   EQU     06H
(0007)        3907 CHSCN   EQU     07H
              3908 ;
(0000)        3909 P1      EQU     00H            ;channel states
(0022)        3910 P2      EQU     22H
(0013)        3911 P3      EQU     13H
(0024)        3912 P4      EQU     24H
(0025)        3913 P5      EQU     25H
(0026)        3914 P6      EQU     26H
(0017)        3915 P7      EQU     17H
(0028)        3916 P8      EQU     28H
(0019)        3917 P9      EQU     19H
(002A)        3918 P10     EQU     2AH
(008B)        3919 P11     EQU     8BH
(008C)        3920 P12     EQU     8CH
(008D)        3921 P13     EQU     8DH
(008E)        3922 P14     EQU     8EH
(008F)        3923 P15     EQU     8FH
              3924 ;
(0030)        3925 IDMASK  EQU     30H            ;pack ID field mask
(000F)        3926 STAMSK  EQU     0FH            ;channel state masks
(00F0)        3927 STMSK1  EQU     0F0H
              3928 ;
(0007)        3929 SBUF    EQU     07H            ;PIO control signals
(0006)        3930 DBUF    EQU     06H
(0005)        3931 RDDB    EQU     05H
(0004)        3932 WRSB    EQU     04H
              3933 ;
(00E0)        3934 PMASK1  EQU     0E0H           ;PIO control masks
(0010)        3935 PMASK2  EQU     10H
(0020)        3936 PMASK3  EQU     20H
(00DF)        3937 PMASK4  EQU     0DFH
              3938 ;
(00C0)        3939 RMORE   EQU     00H            ;handshake field
              3940 ;
(0000)        3941 ONI     EQU     00H            ;LAUi state flags
(0001)        3942 SCNI    EQU     01H
(0002)        3943 RSTI    EQU     02H
(0003)        3944 COMI    EQU     03H
(0004)        3945 TONI    EQU     04H
(0005)        3946 BFI     EQU     05H
              3947 ;
(0001)        3948 INONI   EQU     01H            ;ONI=1
(0000)        3949 INOFFI  EQU     00H            ;ALL=0
(0007)        3950 SCONI   EQU     07H            ;ONI=SCNI=RSTI=1
```

```
(0006)        3951 SCOFFI   EQU    06H              ;SCNI=RSTI=1
(000D)        3952 RSCMI    EQU    0DH              ;ONI=RSTI=COMI=1
              3953 ;
(F000)        3954 BUFAR    EQU    RAMTAB           ;Buffer area
(F000)        3955 BUF1     EQU    BUFAR
(F088)        3956 BUF2     EQU    BUF1+BUFL
(F110)        3957 BUF3     EQU    BUF2+BUFL
(F198)        3958 BUF4     EQU    BUF3+BUFL
(F220)        3959 BUF5     EQU    BUF4+BUFL
(F2A8)        3960 BUF6     EQU    BUF5+BUFL
(F330)        3961 BUF7     EQU    BUF6+BUFL
(F3B8)        3962 BUF8     EQU    BUF7+BUFL
(F440)        3963 BUF9     EQU    BUF8+BUFL
(F4C8)        3964 BUF10    EQU    BUF9+BUFL
(F550)        3965 BUF11    EQU    BUF10+BUFL
(F5D8)        3966 BUF12    EQU    BUF11+BUFL
(F660)        3967 BUF13    EQU    BUF12+BUFL
(F6E8)        3968 BUF14    EQU    BUF13+BUFL
(F770)        3969 BUF15    EQU    BUF14+BUFL
(F7F8)        3970 SDBUF    EQU    BUF15+BUFL
              3971 ;
(F800)        3972 QAR      EQU    SDBUF+SDBUFL
(0000)        3973 RAM0     EQU    [QL-[QAR MOD QL]]MOD QL
(F800)        3974 TPOOL    EQU    QAR+RAM0         ;queue pointers
(F800)        3975 TPQ1     EQU    TPOOL
(F820)        3976 TPQ2     EQU    TPQ1+QL
(F840)        3977 TPQ3     EQU    TPQ2+QL
(F860)        3978 TPQ4     EQU    TPQ3+QL
(F880)        3979 TPQ5     EQU    TPQ4+QL
(F8A0)        3980 TPQ6     EQU    TPQ5+QL
              3981 ;
(F8C0)        3982 DCEAR    EQU    TPQ6+QL
(0000)        3983 RAM1     EQU    [MNDCE-[DCEAR MOD MNDCE]]MOD MNDCE
(F8C0)        3984 RDCENC   EQU    DCEAR+RAM1       ;LAUi tables
(F8D0)        3985 RDCEHC   EQU    RDCENC+MNDCE
(F8E0)        3986 RDCEST   EQU    RDCEHC+MNDCE
(F8F0)        3987 RDCERS   EQU    RDCEST+MNDCE
              3988 ;
(F900)        3989 CHAR     EQU    RDCERS+MNDCE
(000C)        3990 RAM2     EQU    [CHMOD-[CHAR MOD CHMOD]]MOD CHMOD
(F900)        3991 CSI      EQU    CHAR+RAM2        ;channel tables
(F908)        3992 RDCE     EQU    CSI+NBCH
(F910)        3993 RCI      EQU    RDCE+NBCH
              3994 ;
(F918)        3995 RGAR     EQU    RCI+NBCH
(0008)        3996 RAM3     EQU    [10H-[RGAR MOD 10H]]MOD 10H
(F920)        3997 RG1      EQU    RGAR+RAM3
(F920)        3998 RCREGA   EQU    RG1              ;rcv/transm registers
(F921)        3999 TRREGB   EQU    RG1+1
(F922)        4000 RBYTE    EQU    RG1+2
(F923)        4001 TBYTE    EQU    RG1+3
(F924)        4002 RPOINT   EQU    RG1+4
(F926)        4003 TPOINT   EQU    RG1+6
(F928)        4004 RBUF     EQU    RG1+8
              4005 ;
(F930)        4006 RG2      EQU    RG1+10H
(F930)        4007 LSTATE   EQU    RG2              ;link level registers
(F932)        4008 MAXNS    EQU    RG2+2
(F933)        4009 MINNS    EQU    RG2+3
(F934)        4010 LASTNR   EQU    RG2+4
```

```
(F935)        4011 SEQNBR  EQU    RG2+5
(F936)        4012 RETCNT  EQU    RG2+6
(F937)        4013 RICNT   EQU    RG2+7
(F938)        4014 REJRSN  EQU    RG2+8
(F939)        4015 REJCTR  EQU    RG2+9
(F93A)        4016 NTF     EQU    RG2+0AH
(F93B)        4017 NRF     EQU    RG2+0BH
              4018 ;
(F940)        4019 RG3     EQU    RG2+10H
(F940)        4020 BUFREG  EQU    RG3              ;buf manager register
(F941)        4021 GENST   EQU    RG3+1            ;pack level registers
(F942)        4022 NRCH    EQU    RG3+2
(F943)        4023 STMC    EQU    RG3+3
(F944)        4024 TONC    EQU    RG3+4
(F945)        4025 SPCC    EQU    RG3+5
(F946)        4026 BSCC    EQU    RG3+6
(F947)        4027 IRLP    EQU    RG3+7
(F948)        4028 CRDL    EQU    RG3+8
(F949)        4029 CLRST   EQU    RG3+9
(F94A)        4030 REMOTE  EQU    RG3+0AH
(F94C)        4031 LOCADD  EQU    RG3+0CH
(F94D)        4032 SIGDIG  EQU    RG3+0DH
              4033 ;
(F950)        4034 RG      EQU    RG3+10H
(F950)        4035 INTREG  EQU    RG               ;timer register
(F951)        4036 TIMRG2  EQU    RG+1             ;timer counters
(F952)        4037 TIMRG3  EQU    RG+2
(F953)        4038 TMRG3A  EQU    RG+3
(F954)        4039 TMRG3B  EQU    RG+4
(F955)        4040 TMRG3C  EQU    RG+5
(F956)        4041 CURBUF  EQU    RG+6             ;buffer pointers
(F958)        4042 SCPT1   EQU    RG+8
(F95A)        4043 SCPT2   EQU    RG+0AH
(F95C)        4044 CRADD   EQU    RG+0CH
              4045 ;
(F960)        4046 QR      EQU    RG+10H           ;queue count/point
(F960)        4047 CTQ1    EQU    QR               ;LIQ=Q1
(F961)        4048 DFQ1    EQU    QR+1
(F962)        4049 DLQ1    EQU    QR+2
(F963)        4050 NOFB    EQU    QR+3
(F964)        4051 NOGB    EQU    QR+4
(F965)        4052 DPOOL   EQU    QR+5
              4053 ;
(F966)        4054 NIF5    EQU    QR+6             ;LOQ=Q6
(F967)        4055 DFQ6    EQU    QR+7
(F968)        4056 DLQ6    EQU    QR+8
(F969)        4057 NIF3    EQU    QR+9
(F96A)        4058 NIF4    EQU    QR+0AH
(F96B)        4059 DTMQ6   EQU    QR+0BH
              4060 ;
(F96C)        4061 CTQ2    EQU    QR+0CH           ;PIQ=Q2
(F96D)        4062 DFQ2    EQU    QR+0DH
(F96E)        4063 DLQ2    EQU    QR+0EH
              4064 ;
(F96F)        4065 NIF1    EQU    QR+0FH           ;POQ=Q5
(F970)        4066 DFQ5    EQU    QR+10H
(F971)        4067 DLQ5    EQU    QR+11H
(F972)        4068 NIF0    EQU    QR+12H
(F973)        4069 NIF2    EQU    QR+13H
(F974)        4070 DTFQ5   EQU    QR+14H
```

```
                4071 ;
(F975)          4072 CTQ4    EQU     QR+15H          ;BIQ=Q4
(F976)          4073 DFQ4    EQU     QR+16H
(F977)          4074 DLQ4    EQU     QR+17H
                4075 ;
(F978)          4076 CTQ3    EQU     QR+18H          ;BOQ=Q3
(F979)          4077 DFQ3    EQU     QR+19H
(F97A)          4078 DLQ3    EQU     QR+1AH
                4079 ;
(F980)          4080 MA      EQU     QR+20H
(F980)          4081 PARAM   EQU     MA              ;monitor registers
(F982)          4082 MESG    EQU     MA+2
(F984)          4083 MBUF    EQU     MA+4
(F986)          4084 MBPT    EQU     MA+6
(F988)          4085 CDBUF   EQU     MA+8
(F98A)          4086 ACTPT   EQU     MA+0AH
(F98C)          4087 TRCPT   EQU     MA+0CH
                4088 ;
(F990)          4089 MA1     EQU     MA+10H
(F990)          4090 MFCH    EQU     MA1
(F991)          4091 MNBYT   EQU     MA1+1
(F992)          4092 MNCH    EQU     MA1+2
(F993)          4093 MBCH    EQU     MA1+3
(F994)          4094 MCBL    EQU     MA1+4
(F995)          4095 NDBUF   EQU     MA1+5
(F996)          4096 TRCN0   EQU     MA1+6
(F997)          4097 TRCN1   EQU     MA1+7
(F998)          4098 ACTN0   EQU     MA1+8
(F999)          4099 ACTN1   EQU     MA1+9
(F99A)          4100 ACTC1   EQU     MA1+0AH
(F99B)          4101 ACTC2   EQU     MA1+0BH
                4102 ;
(F9A0)          4103 MSBUF   EQU     MA1+10H         ;monitor buffers
                4104 ;
(F9B0)          4105 MA2     EQU     MSBUF+10H
(F9B0)          4106 TRCDP   EQU     MA2
                4107 ;
(F9F0)          4108 MA3     EQU     TRCDP+DBL/2
(0010)          4109 RAM4    EQU     [DBL-[MA3 MOD DBL]]MOD DBL
(FA00)          4110 MBAR    EQU     MA3+RAM4
(FA00)          4111 RCDBUF  EQU     MBAR
(FA40)          4112 CRCDP   EQU     RCDBUF+DBL/2
(FA80)          4113 STBUF   EQU     RCDBUF+DBL
(FB00)          4114 MBUF1   EQU     STBUF+DBL
(FB80)          4115 ACTBUF  EQU     MBUF1+MNBUF*DBL
                4116 ;
(FA40)          4117 RM      EQU     CRCDP           ;Record buffer area
(FA40)          4118 FRMA    EQU     RM
(FA42)          4119 FRMB    EQU     RM+02
(FA44)          4120 INFA    EQU     RM+04
(FA46)          4121 INFB    EQU     RM+06
(FA48)          4122 COMA    EQU     RM+08
(FA4A)          4123 COMB    EQU     RM+10
(FA4C)          4124 RRA     EQU     RM+12
(FA4E)          4125 RRB     EQU     RM+14
(FA50)          4126 RNRA    EQU     RM+16
(FA52)          4127 RNRB    EQU     RM+18
(FA54)          4128 REJA    EQU     RM+20
(FA56)          4129 REJB    EQU     RM+22
(FA58)          4130 CMRA    EQU     RM+24
```

```
(FA5A)          4131 CMRB    EQU    RM+26
(FA5C)          4132 PAKA    EQU    RM+28
(FA5E)          4133 PAKB    EQU    RM+30
(FA60)          4134 CALA    EQU    RM+32
(FA62)          4135 CALB    EQU    RM+34
(FA64)          4136 DTEDAT  EQU    RM+36
(FA66)          4137 DCEDAT  EQU    RM+38
(FA68)          4138 DTERS   EQU    RM+40
(FA6A)          4139 DCERS   EQU    RM+42
(FA6C)          4140 IVR     EQU    RM+44
(FA6E)          4141 TM2     EQU    RM+46
(FA70)          4142 LFL     EQU    RM+48
(FA72)          4143 CER     EQU    RM+50
(FA74)          4144 TM3     EQU    RM+52
(FA76)          4145 PFL     EQU    RM+54
                4146 ;
(0000)          4147 M00     EQU    00          ;message codes
(0001)          4148 M01     EQU    01
(0002)          4149 M02     EQU    02
(0003)          4150 M03     EQU    03
(0004)          4151 M04     EQU    04
(0005)          4152 M05     EQU    05
                4153 ;
(0008)          4154 M08     EQU    08
(0009)          4155 M09     EQU    09
(000A)          4156 M10     EQU    10
(000B)          4157 M11     EQU    11
(000C)          4158 M12     EQU    12
(000D)          4159 M13     EQU    13
(000E)          4160 M14     EQU    14
(000F)          4161 M15     EQU    15
(0010)          4162 M16     EQU    16
(0011)          4163 M17     EQU    17
(0012)          4164 M18     EQU    18
(0013)          4165 M19     EQU    19
(0014)          4166 M20     EQU    20
(0015)          4167 M21     EQU    21
(0016)          4168 M22     EQU    22
(0017)          4169 M23     EQU    23
(0018)          4170 M24     EQU    24
(0019)          4171 M25     EQU    25
(001A)          4172 M26     EQU    26
(001B)          4173 M27     EQU    27
(001C)          4174 M28     EQU    28
                4175 ;
(00A0)          4176 M32     EQU    32+128
(00A1)          4177 M33     EQU    33+128
(00A2)          4178 M34     EQU    34+128
(00A3)          4179 M35     EQU    35+128
(00A4)          4180 M36     EQU    36+128
(00A5)          4181 M37     EQU    37+128
(00A6)          4182 M38     EQU    38+128
(00A7)          4183 M39     EQU    39+128
(00A8)          4184 M40     EQU    40+128
(00A9)          4185 M41     EQU    41+128
(00AA)          4186 M42     EQU    42+128
(00AB)          4187 M43     EQU    43+128
(00AC)          4188 M44     EQU    44+128
(00AD)          4189 M45     EQU    45+128
(00AE)          4190 M46     EQU    46+128
```

```
(00AF)          4191 M47        EQU     47+128
(00B0)          4192 M48        EQU     48+128
                4193 ;
(0040)          4194 M64        EQU     64
(0041)          4195 M65        EQU     65
(0042)          4196 M66        EQU     66
(0043)          4197 M67        EQU     67
(0044)          4198 M68        EQU     68
(0045)          4199 M69        EQU     69
(0046)          4200 M70        EQU     70
(0047)          4201 M71        EQU     71
(0048)          4202 M72        EQU     72
(0049)          4203 M73        EQU     73
(004A)          4204 M74        EQU     74
(004B)          4205 M75        EQU     75
(004C)          4206 M76        EQU     76
(004D)          4207 M77        EQU     77
(004E)          4208 M78        EQU     78
(004F)          4209 M79        EQU     79
(0050)          4210 M80        EQU     80
(0051)          4211 M81        EQU     81
(0052)          4212 M82        EQU     82
(0053)          4213 M83        EQU     83
(0054)          4214 M84        EQU     84
(0055)          4215 M85        EQU     85
(0056)          4216 M86        EQU     86
(0057)          4217 M87        EQU     87
(0058)          4218 M88        EQU     88
(0059)          4219 M89        EQU     89
(005A)          4220 M90        EQU     90
(005B)          4221 M91        EQU     91
(005C)          4222 M92        EQU     92
(005D)          4223 M93        EQU     93
(005E)          4224 M94        EQU     94
(005F)          4225 M95        EQU     95
(0060)          4226 M96        EQU     96
(0061)          4227 M97        EQU     97
(0062)          4228 M98        EQU     98
                4229 ;
(00E8)          4230 M104       EQU     104+128
(00E9)          4231 M105       EQU     105+128
(00EA)          4232 M106       EQU     106+128
(00EB)          4233 M107       EQU     107+128
(00EC)          4234 M108       EQU     108+128
(00ED)          4235 M109       EQU     109+128
(00EE)          4236 M110       EQU     110+128
(00EF)          4237 M111       EQU     111+128
(00F0)          4238 M112       EQU     112+128
(00F1)          4239 M113       EQU     113+128
(00F2)          4240 M114       EQU     114+128
(00F3)          4241 M115       EQU     115+128
(00F4)          4242 M116       EQU     116+128
(00F5)          4243 M117       EQU     117+128
(00F6)          4244 M118       EQU     118+128
(00F7)          4245 M119       EQU     119+128
(00F8)          4246 M120       EQU     120+128

Errors          0
```

APPENDIX C

EXCHANGE PROGRAM LISTINGS - EMU CONTROL SOFTWARE

```
                        0001 ;       ************************
                        0002 ;       * MESSAGE ADDRESS TABLE *
                        0003 ;       ************************
                        0004 ;
                        0005 ;
E700  58ED              0006 MADTAB  DW      MA00,MA01,MA02,MA03
      5DED
      65ED
      66ED
E708  6EED              0007         DW      MA04,MA05,ME,ME
      76ED
      55ED
      55ED
E710  7BED              0008         DW      MA08,MA09,MA10,MA11
      80ED
      85ED
      89ED
E718  8EED              0009         DW      MA12,MA13,MA14,MA15
      93ED
      97ED
      9CED
E720  A1ED              0010         DW      MA16,MA17,MA18,MA19
      A6ED
      ABED
      B3ED
E728  BBED              0011         DW      MA20,MA21,MA22,MA23
      C3ED
      CBED
      D0ED
E730  D5ED              0012         DW      MA24,MA25,MA26,MA27
      DAED
      DFED
      E4ED
E738  FCED              0013         DW      MA28,ME,ME,ME
      55ED
      55ED
      55ED
E740  0BEE              0014         DW      MA32,MA33,MA34,MA35
      0FEE
      13EE
      17EE
E748  1BEE              0015         DW      MA36,MA37,MA38,MA39
      1FEE
      23EE
      27EE
E750  2BEE              0016         DW      MA40,MA41,MA42,MA43
      2FEE
      33EE
      37EE
E758  3DEE              0017         DW      MA44,MA45,MA46,MA47
      41EE
      45EE
      4BEE
E760  4FEE              0018         DW      MA48,ME,ME,ME
      55ED
      55ED
      55ED
E768  55ED              0019         DW      ME,ME,ME,ME
      55ED
      55ED
```

```
                55ED
E770            55ED            0020            DW      ME,ME,ME,ME
                55ED
                55ED
                55ED
E778            55ED            0021            DW      ME,ME,ME,ME
                55ED
                55ED
                55ED
E780            53EE            0022            DW      MA64,MA65,MA66,MA67
                5BEE
                63EE
                6BEE
E788            73EE            0023            DW      MA68,MA69,MA70,MA71
                7BEE
                83EE
                8BEE
E790            93EE            0024            DW      MA72,MA73,MA74,MA75
                9BEE
                A3EE
                ABEE
E798            B3EE            0025            DW      MA76,MA77,MA78,MA79
                BBEE
                C3EE
                CBEE
E7A0            D3EE            0026            DW      MA80,MA81,MA82,MA83
                DBEE
                E3EE
                EBEE
E7A8            F3EE            0027            DW      MA84,MA85,MA86,MA87
                FBEE
                03EF
                0BEF
E7B0            13EF            0028            DW      MA88,MA89,MA90,MA91
                1BEF
                23EF
                2BEF
E7B8            33EF            0029            DW      MA92,MA93,MA94,MA95
                34EF
                3CEF
                3DEF
E7C0            45EF            0030            DW      MA96,MA97,M98,ME
                4DEF
                6200
                55ED
E7C8            55ED            0031            DW      ME,ME,ME,ME
                55ED
                55ED
                55ED
E7D0            5DEF            0032            DW      MA104,MA105,MA106,MA107
                63EF
                67EF
                6BEF
E7D8            6FEF            0033            DW      MA108,MA109,MA110,MA111
                8AEF
                A4EF
                A8EF
E7E0            ACEF            0034            DW      MA112,MA113,MA114,MA115
                B0EF
                B4EF
```

```
      B8EF
E7E8  BCEF          0035          DW      MA116,MA117,MA118,MA119
      C2EF
      C6EF
      CAEF
E7F0  CEEF          0036          DW      MA120,ME,ME,ME
      55ED
      55ED
      55ED
E7F8  55ED          0037          DW      ME,ME,ME,ME
      55ED
      55ED
      55ED
                    0038 ;
                    0039 ;
                    0040 ;        **********************
                    0041 ;        * PROCESS SCHEDULER *
                    0042 ;        **********************
                    0043 ;
                    0044 ;
E800  F3            0045 BEGIN    DI                            ;entry (from DDT)
E801  ED7BE6FF      0046          LD      SP,(MNSP)             ;init stack pointer
E805  CD19E8        0047          CALL    INIT                  ;monitor init
E808  CD7BE8        0048 RUN      CALL    INPUT
E80B  CD39EA        0049          CALL    TR
E80E  CD73EA        0050          CALL    REC
E811  CD99EB        0051          CALL    TMOUT
E814  CD33EC        0052          CALL    PRINT
E817  18EF          0053          JR      RUN
                    0054 ;
                    0055 ;
                    0056 ;        ****************************
                    0057 ;        * MONITOR INITIALIZATION *
                    0058 ;        ****************************
                    0059 ;
                    0060 ;
E819  ED5E          0061 INIT     IM      2                     ;set interrupt mode
E81B  3EEF          0062          LD      A,IVEC
E81D  ED47          0063          LD      I,A
E81F  32FBFF        0064          LD      (DDTI),A
E822  010800        0065          LD      BC,2*NBUF             ;init pool
E825  21F0EF        0066          LD      HL,BUFTAB
E828  1130F2        0067          LD      DE,TPOOL
E82B  EDB0          0068          LDIR
E82D  3E04          0069          LD      A,NBUF
E82F  326CF2        0070          LD      (DPOOL),A
E832  3256F2        0071          LD      (NOFB),A
E835  AF            0072          XOR     A
E836  326AF2        0073          LD      (DFQ),A               ;init rcv queue
E839  326BF2        0074          LD      (DLQ),A
E83C  3255F2        0075          LD      (RCNT),A
E83F  CD1AEA        0076          CALL    STATIN                ;init EMU/LAU's state
E842  3E0B          0077          LD      A,MNADD               ;EMU address
E844  325FF2        0078          LD      (LOCADD),A            ;save in LOCADD
E847  3E0A          0079          LD      A,TIME                ;time-out interval
E849  325EF2        0080          LD      (TI),A                ;save in TI
E84C  3E03          0081 PERES    LD      A,03                  ;periph reset
E84E  D3DC          0082          OUT     (CTC0),A
E850  D3DD          0083          OUT     (CTC1),A
E852  D3DE          0084          OUT     (CTC2),A
```

```
E854  D3DF        0085          OUT     (CTC3),A
E856  21E6EF      0086 PINIT    LD      HL,PIOTAB       ;PIO init (PIOTAB)
E859  0EBD        0087          LD      C,PIOCA
E85B  EDA3        0088          OUTI
E85D  EDA3        0089          OUTI
E85F  0EBF        0090          LD      C,PIOCB
E861  EDA3        0091          OUTI
E863  EDA3        0092          OUTI
E865  0D          0093          DEC     C
E866  EDA3        0094          OUTI
E868  21EBEF      0095 TINIT    LD      HL,TIMTAB       ;CTC init (TIMTAB)
E86B  0EDC        0096          LD      C,CTC0
E86D  EDA3        0097          OUTI
E86F  0EDE        0098          LD      C,CTC2
E871  EDA3        0099          OUTI
E873  EDA3        0100          OUTI
E875  0C          0101          INC     C
E876  EDA3        0102          OUTI
E878  EDA3        0103          OUTI
E87A  C9          0104          RET
                  0105 ;
                  0106 ;
                  0107 ;        ******************
                  0108 ;        * CONSOLE INPUT *
                  0109 ;        ******************
                  0110 ;
                  0111 ;
E87B  1E80        0112 INPUT    LD      E,80H
E87D  CD22E5      0113          CALL    RDCHR           ;input char (DDT)
E880  CB7B        0114          BIT     7,E             ;test if char read
E882  C208E9      0115          JP      NZ,ENDINP
E885  2150F2      0116          LD      HL,STATE
E888  CB4E        0117          BIT     COM,(HL)        ;test command sent
E88A  282A        0118          JR      Z,INPUT3
E88C  FE4B        0119          CP      A,'K'           ;com sent/Kill?
E88E  200E        0120          JR      NZ,INPUT1
E890  CD1AEA      0121          CALL    STATIN          ;abort command
E893  1E00        0122          LD      E,00
E895  2148ED      0123          LD      HL,MRET
E898  CDC7E3      0124          CALL    PTXT
E89B  C308E9      0125          JP      ENDINP
E89E  FE48        0126 INPUT1   CP      A,'H'           ;Halt?
E8A0  C208E9      0127          JP      NZ,ENDINP
E8A3  CB5E        0128          BIT     PRES,(HL)       ;Halt/test display
E8A5  CA08E9      0129          JP      Z,ENDINP
E8A8  CB66        0130          BIT     STPR,(HL)       ;test if stopped
E8AA  2805        0131          JR      Z,INPUT2
E8AC  CBA6        0132          RES     STPR,(HL)       ;stopped/resume
E8AE  C308E9      0133          JP      ENDINP
E8B1  CBE6        0134 INPUT2   SET     STPR,(HL)       ;stop display
E8B3  C308E9      0135          JP      ENDINP
E8B6  1E00        0136 INPUT3   LD      E,00
E8B8  57          0137          LD      D,A
E8B9  CD27E5      0138          CALL    WRCHR           ;echo character
E8BC  7A          0139          LD      A,D
E8BD  CB46        0140          BIT     MODE,(HL)       ;test current mode
E8BF  2022        0141          JR      NZ,INPUT5
E8C1  FE58        0142          CP      A,'X'           ;DDT mode/swap?
E8C3  200D        0143          JR      NZ,INPUT4
E8C5  CBC6        0144          SET     MODE,(HL)       ;set X25 mode
```

```
E8C7  2143ED      0145          LD      HL,MX25
E8CA  1E00        0146          LD      E,00
E8CC  CDC7E3      0147          CALL    PTXT          ;prompt >X25
E8CF  C308E9      0148          JP      ENDINP
E8D2  ED73E6FF    0149 INPUT4   LD      (MNSP),SP     ;stay in DDT
E8D6  31E6FF      0150          LD      SP,DDTSP
E8D9  CD36E1      0151          CALL    DDT80
E8DC  ED7BE6FF    0152          LD      SP,(MNSP)     ;restore monit stack
E8E0  C308E9      0153          JP      ENDINP
E8E3  FE58        0154 INPUT5   CP      A,'X'         ;X25 mode/swap?
E8E5  200D        0155          JR      NZ,INPUT6
E8E7  CB86        0156          RES     MODE,(HL)     ;set DDT mode
E8E9  214CED      0157          LD      HL,MDDT
E8EC  1E00        0158          LD      E,00
E8EE  CDC7E3      0159          CALL    PTXT          ;prompt .DDT
E8F1  C308E9      0160          JP      ENDINP
E8F4  321CFF      0161 INPUT6   LD      (DDTCMD),A    ;save char input
E8F7  1E00        0162          LD      E,00
E8F9  CDA5E5      0163          CALL    SPACE
E8FC  CD14E4      0164          CALL    INPAR         ;input param (DDT)
E8FF  3A1CFF      0165          LD      A,(DDTCMD)
E902  3259F2      0166          LD      (CURCMD),A    ;save command
E905  CD09E9      0167          CALL    DECOM         ;decode command
E908  C9          0168 ENDINP   RET
                  0169 ;
E909  FE50        0170 DECOM    CP      A,'P'         ;dec/validate command
E90B  2826        0171          JR      Z,DEC1
E90D  FE49        0172          CP      A,'I'
E90F  2822        0173          JR      Z,DEC1
E911  FE53        0174          CP      A,'S'
E913  281E        0175          JR      Z,DEC1
E915  FE41        0176          CP      A,'A'
E917  281A        0177          JR      Z,DEC1
E919  FE44        0178          CP      A,'D'
E91B  281B        0179          JR      Z,DEC2
E91D  FE54        0180          CP      A,'T'
E91F  2817        0181          JR      Z,DEC2
E921  FE52        0182          CP      A,'R'
E923  2813        0183          JR      Z,DEC2
E925  FE00        0184          CP      A,00
E927  2005        0185          JR      NZ,DEC0
E929  2148ED      0186          LD      HL,MRET
E92C  1817        0187          JR      DEC3
E92E  2138ED      0188 DEC0     LD      HL,MCMERR     ;inval command
E931  1812        0189          JR      DEC3
E933  CD53E9      0190 DEC1     CALL    RDPAR         ;read parameters
E936  1814        0191          JR      DEC4
E938  CD53E9      0192 DEC2     CALL    RDPAR
E93B  3A50F2      0193          LD      A,(STATE)
E93E  CB57        0194          BIT     GLB,A         ;test address mode
E940  280A        0195          JR      Z,DEC4
E942  212DED      0196          LD      HL,MADERR     ;invalid global addr
E945  1E00        0197 DEC3     LD      E,00          ;print error message
E947  CDC7E3      0198          CALL    PTXT
E94A  1806        0199          JR      ENDEC
E94C  CD89E9      0200 DEC4     CALL    FORM          ;form Monitor Request
E94F  CDB0E9      0201          CALL    PCOM          ;process command
E952  C9          0202 ENDEC    RET
                  0203 ;
E953  3A18FF      0204 RDPAR    LD      A,(OP3)       ;read parameter3
```

```
E956 CD7DE9      0205          CALL    BCDBIN
E959 325CF2      0206          LD      (PARAM3),A
E95C 3A16FF      0207          LD      A,(OP2)         ;read parameter2
E95F CD7DE9      0208          CALL    BCDBIN
E962 325BF2      0209          LD      (PARAM2),A
E965 3A14FF      0210          LD      A,(OP1)         ;read parameter1
E968 CD7DE9      0211          CALL    BCDBIN          ;(LAU address)
E96B 325AF2      0212          LD      (PARAM1),A
E96E 5F          0213          LD      E,A
E96F 3E0A        0214          LD      A,MXDCE         ;test address mode
E971 93          0215          SUB     E
E972 2150F2      0216          LD      HL,STATE
E975 3803        0217          JR      C,RDPAR1
E977 CB96        0218          RES     GLB,(HL)        ;set indiv address
E979 C9          0219          RET
E97A CBD6        0220 RDPAR1   SET     GLB,(HL)        ;set global address
E97C C9          0221          RET
                 0222 ;
E97D 4F          0223 BCDBIN   LD      C,A             ;convert decimal/HEX
E97E E6F0        0224          AND     0F0H
E980 0F          0225          RRCA
E981 0F          0226          RRCA
E982 47          0227          LD      B,A
E983 0F          0228          RRCA
E984 80          0229          ADD     A,B
E985 47          0230          LD      B,A
E986 79          0231          LD      A,C
E987 90          0232          SUB     B
E988 C9          0233          RET
                 0234 ;
E989 2120F2      0235 FORM     LD      HL,TBUF         ;get transm buffer
E98C 360A        0236          LD      (HL),MNRQL      ;insert length byte
E98E 23          0237          INC     HL              ;point source field
E98F 3A5FF2      0238          LD      A,(LOCADD)      ;read EMU address
E992 77          0239          LD      (HL),A          ;insert EMU address
E993 AF          0240          XOR     A
E994 23          0241          INC     HL
E995 77          0242          LD      (HL),A
E996 23          0243          INC     HL
E997 77          0244          LD      (HL),A
E998 23          0245          INC     HL
E999 77          0246          LD      (HL),A
E99A 23          0247          INC     HL              ;point to pack field
E99B 36F3        0248          LD      (HL),MNREQ      ;insert Mon Req code
E99D 23          0249          INC     HL              ;point to comm field
E99E 3A59F2      0250          LD      A,(CURCMD)      ;read current command
E9A1 77          0251          LD      (HL),A          ;insert comm (ASCII)
E9A2 23          0252          INC     HL              ;point handsh field
E9A3 3600        0253          LD      (HL),00         ;handsh=00
E9A5 23          0254          INC     HL              ;point to param2 field
E9A6 3A5BF2      0255          LD      A,(PARAM2)      ;read param2
E9A9 77          0256          LD      (HL),A          ;insert param2
E9AA 23          0257          INC     HL              ;point to param3 field
E9AB 3A5CF2      0258          LD      A,(PARAM3)      ;read param3
E9AE 77          0259          LD      (HL),A          ;insert param3
E9AF C9          0260          RET
                 0261 ;
E9B0 2100F3      0262 PCOM     LD      HL,RCMSP        ;init response pointer
E9B3 2262F2      0263          LD      (RCMPT),HL      ;(resp storage area)
E9B6 3A50F2      0264          LD      A,(STATE)       ;update EMU state
```

```
E9B9  CBCF      0265         SET   COM,A
E9BB  CBEF      0266         SET   FIRST,A
E9BD  3250F2    0267         LD    (STATE),A
E9C0  CB57      0268         BIT   GLB,A          ;test address mode
E9C2  2008      0269         JR    NZ,PCOM1
E9C4  3E01      0270         LD    A,01           ;indiv/single LAU
E9C6  4B        0271         LD    C,E            ;LAU address
E9C7  CD89EB    0272         CALL  RMADD          ;save LAU resp pointer
E9CA  1804      0273         JR    PCOM2
E9CC  3E0B      0274 PCOM1   LD    A,NDCE         ;global/all LAU's
E9CE  0E00      0275         LD    C,00           ;first LAU =00
E9D0  2157F2    0276 PCOM2   LD    HL,DCEIN
E9D3  71        0277         LD    (HL),C         ;first LAU to address
E9D4  2158F2    0278         LD    HL,DCEOUT
E9D7  71        0279         LD    (HL),C         ;first LAU to display
E9D8  3253F2    0280         LD    (OUTCNT),A     ;nbr resp to display
E9DB  213FF2    0281         LD    HL,STATE0-1    ;point LAU state tab
E9DE  0600      0282         LD    B,00
E9E0  09        0283         ADD   HL,BC          ;point first LAU state
E9E1  47        0284         LD    B,A
E9E2  23        0285 PCOM3   INC   HL
E9E3  CB46      0286         BIT   ONI,(HL)       ;test LAUi ON/OFF
E9E5  2804      0287         JR    Z,PCOM4
E9E7  3603      0288         LD    (HL),SCONI     ;ON/send comm packet
E9E9  181A      0289         JR    PCOM6
E9EB  3608      0290 PCOM4   LD    (HL),SCOFFI    ;OFF/local message
E9ED  E5        0291         PUSH  HL
E9EE  3D        0292         DEC   A
E9EF  2005      0293         JR    NZ,PCOM5
E9F1  2150F2    0294         LD    HL,STATE
E9F4  CBDE      0295         SET   PRES,(HL)      ;all OFF/set display
E9F6  59        0296 PCOM5   LD    E,C            ;insert local message
E9F7  CD89EB    0297         CALL  RMADD          ;save LAUi resp point
E9FA  EB        0298         EX    DE,HL
E9FB  3600      0299         LD    (HL),M00       ;message OFF
E9FD  23        0300         INC   HL
E9FE  36FF      0301         LD    (HL),MEOB      ;message end of block
EA00  23        0302         INC   HL
EA01  2262F2    0303         LD    (RCMPT),HL     ;update resp pointer
EA04  E1        0304         POP   HL
EA05  0C        0305 PCOM6   INC   C              ;next LAU
EA06  10DA      0306         DJNZ  PCOM3          ;repeat all addr LAU's
EA08  3252F2    0307         LD    (INCNT),A      ;update transm count
EA0B  3254F2    0308         LD    (RESCNT),A     ;update resp count
EA0E  B7        0309         OR    A
EA0F  2808      0310         JR    Z,PCOM7
EA11  2151F2    0311         LD    HL,TIMREG
EA14  CBD6      0312         SET   TON,(HL)       ;start timer
EA16  CD26EC    0313         CALL  START          ;(command sent)
EA19  C9        0314 PCOM7   RET
                0315 ;
EA1A  3E01      0316 STATIN  LD    A,INITST       ;init EMU state
EA1C  3250F2    0317         LD    (STATE),A
EA1F  AF        0318         XOR   A              ;init counters
EA20  3252F2    0319         LD    (INCNT),A      ;transm count
EA23  3253F2    0320         LD    (OUTCNT),A     ;display count
EA26  3254F2    0321         LD    (RESCNT),A     ;response count
EA29  060B      0322         LD    B,NDCE         ;nbr LAU's
EA2B  213FF2    0323         LD    HL,STATE0-1    ;point LAU state tab
EA2E  23        0324 STAT1   INC   HL
```

```
EA2F  7E        0325        LD      A,(HL)
EA30  E601      0326        AND     ONIMSK
EA32  77        0327        LD      (HL),A          ;init LAUi state
EA33  10F9      0328        DJNZ    STAT1           ;repeat all LAU's
EA35  CD1FEC    0329        CALL    STOP            ;stop timer
EA38  C9        0330        RET
                0331 ;
                0332 ;
                0333 ;      ************************
                0334 ;      * PACKET TRANSMISSION *
                0335 ;      ************************
                0336 ;
                0337 ;
EA39  2152F2    0338 TR     LD      HL,INCNT        ;nbr pack to transm
EA3C  AF        0339        XOR     A
EA3D  BE        0340        CP      (HL)
EA3E  2832      0341        JR      Z,ENDTR
EA40  DBBE      0342        IN      A,(PIODB)       ;read PIO register
EA42  CB7F      0343        BIT     SBUF,A          ;test source buffer
EA44  282C      0344        JR      Z,ENDTR
EA46  35        0345        DEC     (HL)            ;buf empty
EA47  3A57F2    0346        LD      A,(DCEIN)       ;get next LAU address
EA4A  1600      0347        LD      D,00
EA4C  5F        0348        LD      E,A
EA4D  213FF2    0349        LD      HL,STATE0-1     ;point LAU state tab
EA50  19        0350        ADD     HL,DE
EA51  23        0351 TR1    INC     HL              ;point LAUi address
EA52  57        0352        LD      D,A             ;LAU address
EA53  3C        0353        INC     A               ;next LAU address
EA54  CB4E      0354        BIT     SCNI,(HL)       ;LAUi addressed?
EA56  28F9      0355        JR      Z,TR1
EA58  CB8E      0356        RES     SCNI,(HL)       ;send command to LAUi
EA5A  CBD6      0357        SET     COMI,(HL)
EA5C  3257F2    0358        LD      (DCEIN),A       ;save next LAU address
EA5F  0EBC      0359        LD      C,PIODA
EA61  2120F2    0360        LD      HL,TBUF         ;point transm buf
EA64  DBBE      0361        IN      A,(PIODB)
EA66  E6E0      0362        AND     PMASK1
EA68  B2        0363        OR      D
EA69  D3BE      0364        OUT     (PIODB),A       ;write dest addr (PIO)
EA6B  46        0365        LD      B,(HL)          ;read length byte
EA6C  EDB3      0366        OTIR                    ;pack transf
EA6E  F610      0367        OR      PMASK2
EA70  D3BE      0368        OUT     (PIODB),A       ;pack write pulse
EA72  C9        0369 ENDTR  RET
                0370 ;
                0371 ;
                0372 ;      ********************
                0373 ;      * PACKET RECEPTION *
                0374 ;      ********************
                0375 ;
                0376 ;
EA73  DBBE      0377 REC    IN      A,(PIODB)       ;read PIO register
EA75  CB77      0378        BIT     DBUF,A          ;test dest buffer
EA77  2832      0379        JR      Z,REC0
EA79  2156F2    0380        LD      HL,NOFB         ;buf full/test pool
EA7C  7E        0381        LD      A,(HL)
EA7D  B7        0382        OR      A
EA7E  282B      0383        JR      Z,REC0
EA80  35        0384        DEC     (HL)            ;buf available
```

```
EA81  2155F2      0385          LD    HL,RCNT       ;inc receiver count
EA84  34          0386          INC   (HL)
EA85  216BF2      0387          LD    HL,DLQ        ;get buffer
EA88  7E          0388          LD    A,(HL)
EA89  34          0389          INC   (HL)
EA8A  2130F2      0390          LD    HL,TPOOL
EA8D  E603        0391          AND   QMASK
EA8F  07          0392          RLCA
EA90  B5          0393          OR    L
EA91  6F          0394          LD    L,A
EA92  7E          0395          LD    A,(HL)
EA93  2C          0396          INC   L
EA94  66          0397          LD    H,(HL)
EA95  6F          0398          LD    L,A
EA96  0EBC        0399          LD    C,PIODA
EA98  ED40        0400          IN    B,(C)         ;dummy read
EA9A  ED40        0401          IN    B,(C)         ;read length byte
EA9C  70          0402          LD    (HL),B        ;insert length byte
EA9D  05          0403          DEC   B
EA9E  23          0404          INC   HL
EA9F  EDB2        0405          INIR                ;packet transfer
EAA1  DBBE        0406          IN    A,(PIODB)
EAA3  F620        0407          OR    PMASK3
EAA5  D3BE        0408          OUT   (PIODB),A     ;packet read pulse
EAA7  E6DF        0409          AND   PMASK4
EAA9  D3BE        0410          OUT   (PIODB),A
EAAB  2155F2      0411 REC0     LD    HL,RCNT       ;nbr received pack
EAAE  AF          0412          XOR   A
EAAF  BE          0413          CP    (HL)
EAB0  CA5EEB      0414          JP    Z,ENDRC1
EAB3  35          0415          DEC   (HL)          ;process one packet
EAB4  216AF2      0416          LD    HL,DFQ        ;get packet
EAB7  7E          0417          LD    A,(HL)
EAB8  34          0418          INC   (HL)
EAB9  2130F2      0419          LD    HL,TPOOL
EABC  CD5FEB      0420          CALL  GETBUF
EABF  7E          0421          LD    A,(HL)        ;read length byte
EAC0  D608        0422          SUB   HEADER        ;validate length
EAC2  DA5BEB      0423          JP    C,ENDRC
EAC5  4F          0424          LD    C,A           ;data field length
EAC6  23          0425          INC   HL
EAC7  5E          0426          LD    E,(HL)        ;read source LAU
EAC8  23          0427          INC   HL
EAC9  23          0428          INC   HL
EACA  23          0429          INC   HL
EACB  23          0430          INC   HL
EACC  7E          0431          LD    A,(HL)        ;read packet type
EACD  FEF7        0432          CP    MNCNF         ;validate (Mon Cnf)
EACF  C25BEB      0433          JP    NZ,ENDRC
EAD2  23          0434          INC   HL
EAD3  46          0435          LD    B,(HL)        ;read command field
EAD4  3A59F2      0436          LD    A,(CURCMD)    ;validate command
EAD7  B8          0437          CP    B
EAD8  C25BEB      0438          JP    NZ,ENDRC
EADB  23          0439          INC   HL
EADC  46          0440          LD    B,(HL)        ;read handshake field
EADD  2264F2      0441          LD    (RBPNT),HL    ;save point (message)
EAE0  2140F2      0442          LD    HL,STATE0     ;point LAU state tab
EAE3  1600        0443          LD    D,00
EAE5  19          0444          ADD   HL,DE         ;point LAUi state
```

```
EAE6  CB56      0445          BIT   COMI,(HL)       ;test comm sent (LAUi)
EAE8  CA5BEB    0446          JP    Z,ENDRC
EAEB  CB96      0447          RES   COMI,(HL)       ;valid response
EAED  CBDE      0448          SET   RESI,(HL)
EAEF  3A50F2    0449          LD    A,(STATE)
EAF2  CB57      0450          BIT   GLB,A           ;test address mode
EAF4  2813      0451          JR    Z,REC1
EAF6  CD89EB    0452          CALL  RMADD           ;global
EAF9  2154F2    0453          LD    HL,RESCNT
EAFC  35        0454          DEC   (HL)            ;dec resp expected
EAFD  2049      0455          JR    NZ,REC3         ;test if all received
EAFF  CBDF      0456          SET   PRES,A          ;all recv/set display
EB01  3250F2    0457          LD    (STATE),A
EB04  CD1FEC    0458          CALL  STOP            ;stop timer
EB07  183F      0459          JR    REC3
EB09  CBDF      0460 REC1     SET   PRES,A          ;indiv/set display
EB0B  3250F2    0461          LD    (STATE),A
EB0E  CB40      0462          BIT   RMORE,B         ;test handshake
EB10  2820      0463          JR    Z,REC2
EB12  CBCE      0464          SET   SCNI,(HL)       ;=1/repeat command
EB14  CD1FEC    0465          CALL  STOP            ;stop timer
EB17  3E01      0466          LD    A,01            ;update counters
EB19  3254F2    0467          LD    (RESCNT),A      ;resp count
EB1C  3252F2    0468          LD    (INCNT),A       ;transm count
EB1F  7B        0469          LD    A,E             ;LAU to address
EB20  3257F2    0470          LD    (DCEIN),A
EB23  2151F2    0471          LD    HL,TIMREG
EB26  CBD6      0472          SET   TON,(HL)        ;start timer
EB28  CD26EC    0473          CALL  START
EB2B  3E01      0474          LD    A,01            ;set handshake
EB2D  3227F2    0475          LD    (MFIELD),A
EB30  1816      0476          JR    REC3
EB32  2154F2    0477 REC2     LD    HL,RESCNT       ;hand=0/not repeated
EB35  3600      0478          LD    (HL),00
EB37  CD1FEC    0479          CALL  STOP            ;stop timer
EB3A  3A59F2    0480          LD    A,(CURCMD)
EB3D  FE54      0481          CP    A,'T'           ;test if Trace
EB3F  2007      0482          JR    NZ,REC3
EB41  CB48      0483          BIT   EOTR,B          ;Trace/test end trace
EB43  2003      0484          JR    NZ,REC3
EB45  CD26EC    0485          CALL  START           ;not end/start timer
EB48  ED5B62F2  0486 REC3     LD    DE,(RCMPT)      ;transf response from
EB4C  0600      0487          LD    B,00            ;buf to storage area
EB4E  2A64F2    0488          LD    HL,(RBPNT)
EB51  23        0489          INC   HL
EB52  EDB0      0490          LDIR
EB54  ED5362F2  0491          LD    (RCMPT),DE
EB58  3EFE      0492          LD    A,MTEOB         ;insert synch message
EB5A  12        0493          LD    (DE),A
EB5B  CD6CEB    0494 ENDRC    CALL  RELBUF          ;release buf
EB5E  C9        0495 ENDRC1   RET
                0496 ;
                0497 ;        ********************
                0498 ;        * BUFFER ROUTINES *
                0499 ;        ********************
                0500 ;
EB5F  E603      0501 GETBUF   AND   QMASK           ;get buffer
EB61  07        0502          RLCA                  ;entry offset
EB62  B5        0503          OR    L
EB63  6F        0504          LD    L,A             ;point to entry
```

```
EB64  7E          0505          LD    A,(HL)          ;address LSB
EB65  2C          0506          INC   L
EB66  66          0507          LD    H,(HL)          ;address MSB
EB67  6F          0508          LD    L,A
EB68  2266F2      0509          LD    (CURBUF),HL     ;save in CURBUF
EB6B  C9          0510          RET
EB6C  216CF2      0511 RELBUF   LD    HL,DPOOL        ;release buffer
EB6F  7E          0512          LD    A,(HL)
EB70  34          0513          INC   (HL)
EB71  2130F2      0514          LD    HL,TPOOL
EB74  E603        0515          AND   QMASK
EB76  07          0516          RLCA                  ;entry offset
EB77  B5          0517          OR    L
EB78  6F          0518          LD    L,A             ;point to entry
EB79  3A66F2      0519          LD    A,(CURBUF)      ;address LSB
EB7C  77          0520          LD    (HL),A
EB7D  2C          0521          INC   L
EB7E  3A67F2      0522          LD    A,(CURBUF+1)    ;address MSB
EB81  77          0523          LD    (HL),A
EB82  2156F2      0524          LD    HL,NOFB         ;inc free buffers
EB85  F3          0525          DI
EB86  34          0526          INC   (HL)
EB87  FB          0527          EI
EB88  C9          0528          RET
                  0529 ;
EB89  2170F2      0530 RMADD    LD    HL,RADTAB       ;point resp addr tab
EB8C  1600        0531          LD    D,00
EB8E  CB23        0532          SLA   E               ;offset (LAU nbr)
EB90  19          0533          ADD   HL,DE           ;LAUi resp addr point
EB91  ED5B62F2    0534          LD    DE,(RCMPT)      ;get resp point
EB95  73          0535          LD    (HL),E          ;save in LAUi entry
EB96  23          0536          INC   HL
EB97  72          0537          LD    (HL),D
EB98  C9          0538          RET
                  0539 ;
                  0540 ;
                  0541 ;        *********
                  0542 ;        * TIMER *
                  0543 ;        *********
                  0544 ;
                  0545 ;
EB99  2151F2      0546 TMOUT    LD    HL,TIMREG
EB9C  CB46        0547          BIT   TOUT,(HL)       ;test time-out
EB9E  CA1EEC      0548          JP    Z,TMOUT1
EBA1  CB86        0549          RES   TOUT,(HL)       ;time-out
EBA3  CB56        0550          BIT   TON,(HL)        ;test resp expect
EBA5  2021        0551          JR    NZ,TERR
EBA7  3E01        0552          LD    A,01            ;not exp/tracing
EBA9  3254F2      0553          LD    (RESCNT),A      ;update counters
EBAC  3252F2      0554          LD    (INCNT),A
EBAF  3A5AF2      0555          LD    A,(PARAM1)      ;read LAU address
EBB2  3257F2      0556          LD    (DCEIN),A       ;save LAU address
EBB5  5F          0557          LD    E,A
EBB6  1600        0558          LD    D,00
EBB8  2140F2      0559          LD    HL,STATE0       ;point LAU state tab
EBBB  19          0560          ADD   HL,DE           ;point to LAUi
EBBC  CBCE        0561          SET   SCNI,(HL)       ;send pack to LAUi
EBBE  CD26EC      0562          CALL  START           ;start timer
EBC1  3E01        0563          LD    A,01            ;set hanshake field
EBC3  3227F2      0564          LD    (MFIELD),A
```

```
EBC6  1856      0565          JR    TMOUT1
EBC8  CB96      0566 TERR     RES   TON,(HL)        ;response failed
EBCA  2150F2    0567          LD    HL,STATE
EBCD  AF        0568          XOR   A
EBCE  CB56      0569          BIT   GLB,(HL)        ;test address mode
EBD0  2003      0570          JR    NZ,TERR1
EBD2  3A5AF2    0571          LD    A,(PARAM1)      ;indiv/get LAU address
EBD5  4F        0572 TERR1    LD    C,A
EBD6  3A54F2    0573          LD    A,(RESCNT)      ;nbr missing responses
EBD9  213FF2    0574          LD    HL,STATE0-1     ;point LAU state tab
EBDC  0600      0575          LD    B,00            ;get failed LAU's
EBDE  09        0576          ADD   HL,BC
EBDF  47        0577          LD    B,A
EBE0  0D        0578          DEC   C
EBE1  0C        0579 TERR2    INC   C
EBE2  23        0580          INC   HL
EBE3  CB4E      0581          BIT   SCNI,(HL)
EBE5  2008      0582          JR    NZ,TERR3
EBE7  CB56      0583          BIT   COMI,(HL)       ;test pack sent (LAUi)
EBE9  28F6      0584          JR    Z,TERR2
EBEB  CB96      0585          RES   COMI,(HL)
EBED  1802      0586          JR    TERR4
EBEF  CB8E      0587 TERR3    RES   SCNI,(HL)
EBF1  3E01      0588 TERR4    LD    A,M01           ;failure message
EBF3  CBDE      0589          SET   RESI,(HL)
EBF5  E5        0590          PUSH  HL
EBF6  2150F2    0591          LD    HL,STATE
EBF9  CB56      0592          BIT   GLB,(HL)        ;test address mode
EBFB  2005      0593          JR    NZ,TERR5
EBFD  2A62F2    0594          LD    HL,(RCMPT)      ;indiv/get resp point
EC00  1805      0595          JR    TERR6
EC02  59        0596 TERR5    LD    E,C             ;global
EC03  CD89EB    0597          CALL  RMADD           ;save LAUi resp point
EC06  EB        0598          EX    DE,HL
EC07  77        0599 TERR6    LD    (HL),A          ;insert fail message
EC08  23        0600          INC   HL
EC09  36FF      0601          LD    (HL),MECB       ;message end of block
EC0B  23        0602          INC   HL
EC0C  2262F2    0603          LD    (RCMPT),HL      ;save resp pointer
EC0F  E1        0604          POP   HL
EC10  10CF      0605          DJNZ  TERR2           ;repeat all fail LAU's
EC12  2150F2    0606          LD    HL,STATE
EC15  CBDE      0607          SET   PRES,(HL)       ;set display
EC17  AF        0608          XOR   A               ;reset counters
EC18  3254F2    0609          LD    (RESCNT),A      ;response count
EC1B  3252F2    0610          LD    (INCNT),A       ;transm count
EC1E  C9        0611 TMOUT1   RET
                0612 ;
EC1F  F3        0613 STOP     DI                    ;stop timer
EC20  2151F2    0614          LD    HL,TIMREG
EC23  3600      0615          LD    (HL),00
EC25  C9        0616          RET
EC26  3A5EF2    0617 START    LD    A,(TI)          ;start timer
EC29  325DF2    0618          LD    (TMCNT),A       ;load count
EC2C  2151F2    0619          LD    HL,TIMREG
EC2F  CBCE      0620          SET   TIM,(HL)
EC31  FB        0621          EI
EC32  C9        0622          RET
                0623 ;
                0624 ;
```

```
                        0625 ;       ******************
                        0626 ;       * CONSOLE OUTPUT *
                        0627 ;       ******************
                        0628 ;
                        0629 ;
EC33  2150F2            0630 PRINT   LD      HL,STATE
EC36  CB5E              0631         BIT     PRES,(HL)       ;test display enabled
EC38  CAD0EC            0632         JP      Z,ENDPR
EC3B  CB66              0633         BIT     STPR,(HL)       ;test stop display
EC3D  C2D0EC            0634         JP      NZ,ENDPR
EC40  CB6E              0635         BIT     FIRST,(HL)      ;display/new LAUi?
EC42  284F              0636         JR      Z,PRINT2
EC44  3A53F2            0637         LD      A,(OUTCNT)      ;nbr resp to display
EC47  B7                0638         OR      A
EC48  200F              0639         JR      NZ,PRINT0
EC4A  CB9E              0640         RES     PRES,(HL)       ;=0/end display
EC4C  CB8E              0641         RES     COM,(HL)        ;end command
EC4E  1E00              0642         LD      E,00
EC50  2148ED            0643         LD      HL,MRET
EC53  CDC7E3            0644         CALL    PTXT            ;prompt >
EC56  C3D0EC            0645         JP      ENDPR
EC59  3D                0646 PRINT0  DEC     A
EC5A  3253F2            0647         LD      (OUTCNT),A      ;update display count
EC5D  CBAE              0648         RES     FIRST,(HL)
EC5F  2123ED            0649         LD      HL,MTITLE       ;print title (*LAU )
EC62  1E00              0650         LD      E,00
EC64  CDC7E3            0651         CALL    PTXT
EC67  3A58F2            0652         LD      A,(DCEOUT)      ;LAU nbr (display)
EC6A  1600              0653         LD      D,00
EC6C  5F                0654         LD      E,A
EC6D  213FF2            0655         LD      HL,STATE0-1     ;point LAU state tab
EC70  19                0656         ADD     HL,DE
EC71  23                0657 PRINT1  INC     HL
EC72  3C                0658         INC     A
EC73  CB5E              0659         BIT     RESI,(HL)       ;test resp recv (LAUi)
EC75  28FA              0660         JR      Z,PRINT1
EC77  3258F2            0661         LD      (DCEOUT),A      ;save next LAU nbr
EC7A  3D                0662         DEC     A
EC7B  5F                0663         LD      E,A
EC7C  1600              0664         LD      D,00
EC7E  CB23              0665         SLA     E
EC80  2170F2            0666         LD      HL,RADTAB       ;point resp addr tab
EC83  19                0667         ADD     HL,DE           ;point LAUi entry
EC84  5E                0668         LD      E,(HL)          ;get LAUi resp address
EC85  23                0669         INC     HL
EC86  66                0670         LD      H,(HL)
EC87  6B                0671         LD      L,E
EC88  2260F2            0672         LD      (PRMPT),HL      ;save display pointer
EC8B  B7                0673         OR      A
EC8C  27                0674         DAA
EC8D  CDFBEC            0675         CALL    PRACC           ;print LAU add (title)
EC90  C3D0EC            0676         JP      ENDPR
EC93  ED5B60F2          0677 PRINT2  LD      DE,(PRMPT)      ;print response
EC97  1A                0678         LD      A,(DE)          ;get current message
EC98  FEFF              0679         CP      A,MEOB          ;end of block?
EC9A  2005              0680         JR      NZ,PRINT3
EC9C  CBEE              0681         SET     FIRST,(HL)      ;end current resp
EC9E  C3D0EC            0682         JP      ENDPR
ECA1  FEFE              0683 PRINT3  CP      A,MTEOB         ;synch message?
ECA3  2005              0684         JR      NZ,PRINT4
```

```
ECA5  CB9E      0685          RES   PRES,(HL)       ;disable disp (wait)
ECA7  C3D0EC    0686          JP    ENDPR
ECAA  CDE0EC    0687 PRINT4   CALL  GETMES          ;print one message
ECAD  2A60F2    0688          LD    HL,(PRMPT)
ECB0  CB7E      0689          BIT   TABLE,(HL)      ;message type?
ECB2  2804      0690          JR    Z,PRINT5
ECB4  23        0691          INC   HL              ;table mess (>1 param)
ECB5  46        0692          LD    B,(HL)          ;get number mess param
ECB6  1806      0693          JR    PRINT6
ECB8  CB76      0694 PRINT5   BIT   PAR,(HL)        ;0/1 param?
ECBA  2810      0695          JR    Z,PRINT7
ECBC  0601      0696          LD    B,01            ;one parameter
ECBE  C5        0697 PRINT6   PUSH  BC              ;print parameters
ECBF  1E00      0698          LD    E,00
ECC1  CDA5E5    0699          CALL  SPACE
ECC4  23        0700          INC   HL
ECC5  7E        0701          LD    A,(HL)
ECC6  CDFBEC    0702          CALL  PRACC           ;print one param
ECC9  C1        0703          POP   BC
ECCA  10F2      0704          DJNZ  PRINT6          ;repeat all param
ECCC  23        0705 PRINT7   INC   HL
ECCD  2260F2    0706          LD    (PRMPT),HL      ;save resp pointer
ECD0  C9        0707 ENDPR    RET
                0708 ;
ECD1  E60F      0709 PRINTA   AND   A,0FH           ;print HEX digit
ECD3  C690      0710          ADD   A,90H
ECD5  27        0711          DAA
ECD6  CE40      0712          ADC   A,40H
ECD8  27        0713          DAA
ECD9  57        0714          LD    D,A
ECDA  1E00      0715          LD    E,00
ECDC  CD27E5    0716          CALL  WRCHR
ECDF  C9        0717          RET
                0718 ;
ECE0  FE02      0719 GETMES   CP    A,M02           ;print single message
ECE2  C8        0720          RET   Z
ECE3  E67F      0721          AND   MMASK
ECE5  2100E7    0722          LD    HL,MADTAB       ;message addr table
ECE8  1600      0723          LD    D,00
ECEA  5F        0724          LD    E,A             ;message number
ECEB  CB23      0725          SLA   E
ECED  19        0726          ADD   HL,DE           ;point message entry
ECEE  5E        0727          LD    E,(HL)          ;get message address
ECEF  23        0728          INC   HL
ECF0  66        0729          LD    H,(HL)
ECF1  6B        0730          LD    L,E
ECF2  1E00      0731          LD    E,00
ECF4  CD9CE5    0732          CALL  CRLF
ECF7  CDC7E3    0733          CALL  PTXT            ;print string (DDT)
ECFA  C9        0734          RET
                0735 ;
ECFB  47        0736 PRACC    LD    B,A             ;print register A
ECFC  07        0737          RLCA
ECFD  07        0738          RLCA
ECFE  07        0739          RLCA
ECFF  07        0740          RLCA
ED00  CDD1EC    0741          CALL  PRINTA
ED03  78        0742          LD    A,B
ED04  CDD1EC    0743          CALL  PRINTA
ED07  C9        0744          RET
```

```
                        0745 ;
                        0746 ;
                        0747 ;       ************************************
                        0748 ;       * TIMER INTERRUPT SERVICE ROUTINE *
                        0749 ;       ************************************
                        0750 ;
                        0751 ;
ED08  F5                0752 TIMER   PUSH    AF
ED09  E5                0753         PUSH    HL
ED0A  2151F2            0754         LD      HL,TIMREG
ED0D  CB4E              0755         BIT     TIM,(HL)        ;test timer running
ED0F  280D              0756         JR      Z,TIMER0
ED11  3A5DF2            0757         LD      A,(TMCNT)       ;ON/dec timer count
ED14  3D                0758         DEC     A
ED15  325DF2            0759         LD      (TMCNT),A
ED18  2004              0760         JR      NZ,TIMER0       ;test time-out condit
ED1A  CBC6              0761         SET     TOUT,(HL)       ;time-out
ED1C  CB8E              0762         RES     TIM,(HL)        ;stop timer
ED1E  E1                0763 TIMER0  POP     HL
ED1F  F1                0764         POP     AF
ED20  FB                0765 ENDINT  EI
ED21  ED4D              0766         RETI
                        0767 ;
                        0768 ;
                        0769 ;       ********************
                        0770 ;       * CONSOLE MESSAGES *
                        0771 ;       ********************
                        0772 ;
                        0773 ;
      (00FF)            0774 MEOB    EQU     0FFH
      (00FE)            0775 MTEOB   EQU     0FEH
                        0776 ;
ED23  0D0A0D0A          0777 MTITLE  DB      CR,LNF,CR,LNF
ED27  2A4C4155          0778         DB      '*LAU ',ETX
      2003
ED2D  2D494E56          0779 MADERR  DB      '-INV AD'
      204144
ED34  0D0A3E03          0780         DB      CR,LNF,PROMPT,ETX
ED38  2D494E56          0781 MCMERR  DB      '-INV CM'
      20434D
ED3F  0D0A3E03          0782         DB      CR,LNF,PROMPT,ETX
ED43  0D0A              0783 MX25    DB      CR,LNF
ED45  583235            0784         DB      'X25'
ED48  0D0A3E03          0785 MRET    DB      CR,LNF,PROMPT,ETX
ED4C  0D0A              0786 MDDT    DB      CR,LNF
ED4E  444454            0787         DB      'DDT'
ED51  0D0A2E03          0788         DB      CR,LNF,'.',ETX
ED55  2D3F03            0789 ME      DB      '-?',ETX
                        0790 ;
ED58  2D4F4646          0791 MA00    DB      '-OFF',ETX
      03
ED5D  2D54494D          0792 MA01    DB      '-TIMOUT',ETX
      4F555403
ED65  03                0793 MA02    DB      ETX
ED66  434D4E44          0794 MA03    DB      'CMND AC',ETX
      20414303
ED6E  434D4E44          0795 MA04    DB      'CMND RJ',ETX
      20524A03
ED76  4F56464C          0796 MA05    DB      'OVFL',ETX
      03
```

```
                        0797 ;
ED7B  52534152          0798 MA08    DB      'RSAR',ETX
      03
ED80  52444953          0799 MA09    DB      'RDIS',ETX
      03
ED85  52554103          0800 MA10    DB      'RUA',ETX
ED89  53534152          0801 MA11    DB      'SSAR',ETX
      03
ED8E  53444953          0802 MA12    DB      'SDIS',ETX
      03
ED93  53554103          0803 MA13    DB      'SUA',ETX
ED97  52535441          0804 MA14    DB      'RSTA',ETX
      03
ED9C  52535442          0805 MA15    DB      'RSTB',ETX
      03
EDA1  52534341          0806 MA16    DB      'RSCA',ETX
      03
EDA6  52534342          0807 MA17    DB      'RSCB',ETX
      03
EDAB  544F5554          0808 MA18    DB      'TOUT LQ',ETX
      204C5103
EDB3  544F5554          0809 MA19    DB      'TOUT SR',ETX
      20535203
EDBB  544F5554          0810 MA20    DB      'TOUT DS',ETX
      20445303
EDC3  544F5554          0811 MA21    DB      'TOUT IF',ETX
      20494603
EDCB  494E464C          0812 MA22    DB      'INFL',ETX
      03
EDD0  4C4B464C          0813 MA23    DB      'LKFL',ETX
      03
EDD5  504B464C          0814 MA24    DB      'PKFL',ETX
      03
EDDA  494E5649          0815 MA25    DB      'INVI',ETX
      03
EDDF  2352454C          0816 MA26    DB      '#REL',ETX
      03
EDE4  0D0A2020          0817 MA27    DB      CR,LNF,'    QC FI LI C1 C2 XI',ETX
      20205143
      20464920
      4C492043
      31204332
      20584903
EDFC  20202020          0818 MA28    DB      '     REC   TRM',ETX
      20524543
      20202054
      524D03
                        0819 ;
EE0B  46524D03          0820 MA32    DB      'FRM',ETX
EE0F  494E4603          0821 MA33    DB      'INF',ETX
EE13  434F4D03          0822 MA34    DB      'COM',ETX
EE17  52522003          0823 MA35    DB      'RR ',ETX
EE1B  524E5203          0824 MA36    DB      'RNR',ETX
EE1F  52454A03          0825 MA37    DB      'REJ',ETX
EE23  434D5203          0826 MA38    DB      'CMR',ETX
EE27  50414B03          0827 MA39    DB      'PAK',ETX
EE2B  43414C03          0828 MA40    DB      'CAL',ETX
EE2F  44415403          0829 MA41    DB      'DAT',ETX
EE33  52535403          0830 MA42    DB      'RST',ETX
EE37  0D0A4956          0831 MA43    DB      CR,LNF,'IVR',ETX
```

```
      5203
EE3D  544D3203      0832 MA44    DB      'TM2',ETX
EE41  4C464C03      0833 MA45    DB      'LFL',ETX
EE45  0D0A4345      0834 MA46    DB      CR,LNF,'CER',ETX
      5203
EE4B  544D3303      0835 MA47    DB      'TM3',ETX
EE4F  50464C03      0836 MA48    DB      'PFL',ETX
                    0837 ;
EE53  52434D52      0838 MA64    DB      'RCMR RC',ETX
      20524303
EE5B  53434D52      0839 MA65    DB      'SCMR RC',ETX
      20524303
EE63  52494E46      0840 MA66    DB      'RINF CF',ETX
      20434603
EE6B  52525220      0841 MA67    DB      'RRR  CF',ETX
      20434603
EE73  52524E52      0842 MA68    DB      'RRNR CF',ETX
      20434603
EE7B  5252454A      0843 MA69    DB      'RREJ CF',ETX
      20434603
EE83  53494E46      0844 MA70    DB      'SINF CF',ETX
      20434603
EE8B  53525220      0845 MA71    DB      'SRR  CF',ETX
      20434603
EE93  53524E52      0846 MA72    DB      'SRNR CF',ETX
      20434603
EE9B  5352454A      0847 MA73    DB      'SREJ CF',ETX
      20434603
EEA3  54434F4D      0848 MA74    DB      'TCOM CF',ETX
      20434603
EEAB  54494E46      0849 MA75    DB      'TINF CF',ETX
      20434603
EEB3  54524553      0850 MA76    DB      'TRES CF',ETX
      20434603
EEBB  494E5646      0851 MA77    DB      'INVF RS',ETX
      20525303
EEC3  494E5652      0852 MA78    DB      'INVR CF',ETX
      20434603
EECB  494E5643      0853 MA79    DB      'INVC CF',ETX
      20434603
EED3  44494147      0854 MA80    DB      'DIAG CH',ETX
      20434803
EEDB  43524551      0855 MA81    DB      'CREQ CH',ETX
      20434803
EEE3  4943414C      0856 MA82    DB      'ICAL CH',ETX
      20434803
EEEB  43414343      0857 MA83    DB      'CACC CH',ETX
      20434803
EEF3  43434F4E      0858 MA84    DB      'CCON CH',ETX
      20434803
EEFB  434C5251      0859 MA85    DB      'CLRQ CH',ETX
      20434803
EF03  434C494E      0860 MA86    DB      'CLIN CH',ETX
      20434803
EF0B  43434641      0861 MA87    DB      'CCFA CH',ETX
      20434803
EF13  43434642      0862 MA88    DB      'CCFB CH',ETX
      20434803
EF1B  44415441      0863 MA89    DB      'DATA CH',ETX
      20434803
```

```
EF23  44415442      0864 MA90    DB      'DATB CH',ETX
      20434803
EF2B  494E5650      0865 MA91    DB      'INVP CH',ETX
      20434803
EF33  03            0866 MA92    DB      ETX
EF34  23434C52      0867 MA93    DB      '#CLR CH',ETX
      20434803
EF3C  03            0868 MA94    DB      ETX
EF3D  23434346      0869 MA95    DB      '#CCF CH',ETX
      20434803
EF45  52454A50      0870 MA96    DB      'REJP RS',ETX
      20525303
EF4D  20202020      0871 MA97    DB      '     RS',ETX
      20525303
EF55  544F5554      0872 MA98    DB      'TOUT CH',ETX
      20434803
                    0873 ;
EF5D  0D0A434E      0874 MA104   DB      CR,LNF,'CNB',ETX
      4203
EF63  43535403      0875 MA105   DB      'CST',ETX
EF67  4C415503      0876 MA106   DB      'LAU',ETX
EF6B  52434803      0877 MA107   DB      'RCH',ETX
EF6F  204C4120      0878 MA108   DB      ' LA LB MX MN NR SQ R1 R2'
      4C42204D
      58204D4E
      204E5220
      53512052
      31205232
EF87  0D0A03        0879         DB      CR,LNF,ETX
EF8A  0D0A2042      0880 MA109   DB      CR,LNF,' BM PK NC ST TM S1 S2'
      4D20504B
      204E4320
      53542054
      4D205331
      205332
EFA1  0D0A03        0881         DB      CR,LNF,ETX
EFA4  4C495103      0882 MA110   DB      'LIQ',ETX
EFA8  4C4F5103      0883 MA111   DB      'LOQ',ETX
EFAC  50495103      0884 MA112   DB      'PIQ',ETX
EFB0  504F5103      0885 MA113   DB      'POQ',ETX
EFB4  42495103      0886 MA114   DB      'BIQ',ETX
EFB8  424F5103      0887 MA115   DB      'BOQ',ETX
EFBC  0D0A4C41      0888 MA116   DB      CR,LNF,'LAU',ETX
      5503
EFC2  4E434803      0889 MA117   DB      'NCH',ETX
EFC6  4E484303      0890 MA118   DB      'NHC',ETX
EFCA  53544103      0891 MA119   DB      'STA',ETX
EFCE  52535403      0892 MA120   DB      'RST',ETX
                    0893 ;
                    0894 ;
                    0895 ;       ******************
                    0896 ;       * MESSAGE CODES *
                    0897 ;       ******************
                    0898 ;
                    0899 ;
      (0000)        0900 M00     EQU     00
      (0001)        0901 M01     EQU     01
      (0002)        0902 M02     EQU     02
      (0003)        0903 M03     EQU     03
      (0004)        0904 M04     EQU     04
```

```
(0005)          0905 M05    EQU     05
                0906 ;
(0008)          0907 M08    EQU     08
(0009)          0908 M09    EQU     09
(000A)          0909 M10    EQU     10
(000B)          0910 M11    EQU     11
(000C)          0911 M12    EQU     12
(000D)          0912 M13    EQU     13
(000E)          0913 M14    EQU     14
(000F)          0914 M15    EQU     15
(001C)          0915 M16    EQU     16
(0011)          0916 M17    EQU     17
(0012)          0917 M18    EQU     18
(0013)          0918 M19    EQU     19
(0014)          0919 M20    EQU     20
(0015)          0920 M21    EQU     21
(0016)          0921 M22    EQU     22
(0017)          0922 M23    EQU     23
(0018)          0923 M24    EQU     24
(0019)          0924 M25    EQU     25
(001A)          0925 M26    EQU     26
(001B)          0926 M27    EQU     27
(001C)          0927 M28    EQU     28
                0928 ;
(00A0)          0929 M32    EQU     32+128
(00A1)          0930 M33    EQU     33+128
(00A2)          0931 M34    EQU     34+128
(00A3)          0932 M35    EQU     35+128
(00A4)          0933 M36    EQU     36+128
(00A5)          0934 M37    EQU     37+128
(00A6)          0935 M38    EQU     38+128
(00A7)          0936 M39    EQU     39+128
(00A8)          0937 M40    EQU     40+128
(00A9)          0938 M41    EQU     41+128
(00AA)          0939 M42    EQU     42+128
(00AB)          0940 M43    EQU     43+128
(00AC)          0941 M44    EQU     44+128
(00AD)          0942 M45    EQU     45+128
(00AE)          0943 M46    EQU     46+128
(00AF)          0944 M47    EQU     47+128
(00B0)          0945 M48    EQU     48+128
                0946 ;
(0040)          0947 M64    EQU     64
(0041)          0948 M65    EQU     65
(0042)          0949 M66    EQU     66
(0043)          0950 M67    EQU     67
(0044)          0951 M68    EQU     68
(0045)          0952 M69    EQU     69
(0046)          0953 M70    EQU     70
(0047)          0954 M71    EQU     71
(0048)          0955 M72    EQU     72
(0049)          0956 M73    EQU     73
(004A)          0957 M74    EQU     74
(004B)          0958 M75    EQU     75
(004C)          0959 M76    EQU     76
(004D)          0960 M77    EQU     77
(004E)          0961 M78    EQU     78
(004F)          0962 M79    EQU     79
                0963 ;
(0050)          0964 M80    EQU     80
```

```
      (0051)        0965 M81     EQU     81
      (0052)        0966 M82     EQU     82
      (0053)        0967 M83     EQU     83
      (0054)        0968 M84     EQU     84
      (0055)        0969 M85     EQU     85
      (0056)        0970 M86     EQU     86
      (0057)        0971 M87     EQU     87
      (0058)        0972 M88     EQU     88
      (0059)        0973 M89     EQU     89
      (005A)        0974 M90     EQU     90
      (005B)        0975 M91     EQU     91
      (005C)        0976 M92     EQU     92
      (005D)        0977 M93     EQU     93
      (005E)        0978 M94     EQU     94
      (005F)        0979 M95     EQU     95
      (0060)        0980 M96     EQU     96
      (0061)        0981 M97     EQU     97
      (0062)        0982 M98     EQU     98
                    0983 ;
      (00E8)        0984 M104    EQU     104+128
      (00E9)        0985 M105    EQU     105+128
      (00EA)        0986 M106    EQU     106+128
      (00EB)        0987 M107    EQU     107+128
      (00EC)        0988 M108    EQU     108+128
      (00ED)        0989 M109    EQU     109+128
      (00EE)        0990 M110    EQU     110+128
      (00EF)        0991 M111    EQU     111+128
      (00F0)        0992 M112    EQU     112+128
      (00F1)        0993 M113    EQU     113+128
      (00F2)        0994 M114    EQU     114+128
      (00F3)        0995 M115    EQU     115+128
      (00F4)        0996 M116    EQU     116+128
      (00F5)        0997 M117    EQU     117+128
      (00F6)        0998 M118    EQU     118+128
      (00F7)        0999 M119    EQU     119+128
      (00F8)        1000 M120    EQU     120+128
                    1001 ;
                    1002 ;
                    1003 ;       **************
                    1004 ;       * ROM TABLES *
                    1005 ;       **************
                    1006 ;
                    1007 ;
EFD2                1008         ORG     0EFE6H
EFE6  8F07CFC0      1009 PIOTAB  DB      8FH,07,0CFH,0C0H,00
      00
EFEB  F81555D5      1010 TIMTAB  DB      TVEC,15H,85D,0D5H,163D
      A3
                    1011 ;
EFF0  00F0          1012 BUFTAB  DW      BUF1,BUF2,BUF3,BUF4
      88F0
      10F1
      98F1
EFF8  (0000)        1013 INTAB   DS      00
EFF8  20ED          1014 TIMINT  DW      ENDINT,ENDINT,ENDINT,TIMER
      20ED
      20ED
      08ED
                    1015 ;
                    1016 ;
```

```
                1017 ;       ***********************************************
                1018 ;       * PERIPHERAL ADDRESSES AND INTERRUPT VECTORS *
                1019 ;       ***********************************************
                1020 ;
                1021 ;
(00EF)          1022 IVEC    EQU     INTAB SHR 8
(00F8)          1023 TVEC    EQU     TIMINT AND 00FFH
(00BC)          1024 PIODA   EQU     0BCH
(00BD)          1025 PIOCA   EQU     PIODA+1
(00BE)          1026 PIODB   EQU     PIODA+2
(00BF)          1027 PIOCB   EQU     PIODA+3
(00DC)          1028 CTC0    EQU     0DCH
(00DD)          1029 CTC1    EQU     CTC0+1
(00DE)          1030 CTC2    EQU     CTC0+2
(00DF)          1031 CTC3    EQU     CTC0+3
                1032 ;
                1033 ;
                1034 ;       ****************************************
                1035 ;       * TABLES, BUFFERS AND MONITOR PARAMETERS *
                1036 ;       ****************************************
                1037 ;
                1038 ;
(E136)          1039 DDT80   EQU     0E136H          ;DDT entry address
(E414)          1040 INPAR   EQU     0E414H          ;DDT param handler
(E522)          1041 RDCHR   EQU     0E522H          ;read character (DDT)
(E527)          1042 WRCHR   EQU     0E527H          ;write character (DDT)
(E59C)          1043 CRLF    EQU     0E59CH          ;carr ret/line feed
(E5A5)          1044 SPACE   EQU     0E5A5H          ;print space (DDT)
(E3C7)          1045 PTXT    EQU     0E3C7H          ;print string (DDT)
                1046 ;
(FF1C)          1047 DDTCMD  EQU     0FF1CH          ;DDT comm register
(FFFB)          1048 DDTI    EQU     0FFFBH          ;DDT I register
(FF14)          1049 OP1     EQU     0FF14H          ;DDT param registers
(FF16)          1050 OP2     EQU     0FF16H
(FF18)          1051 OP3     EQU     0FF18H
(FFE6)          1052 MNSP    EQU     0FFE6H          ;top monitor stack
(FFE6)          1053 DDTSP   EQU     0FFE6H          ;DDT stack point reg
                1054 ;
(0088)          1055 BUFL    EQU     136             ;max buf length
(0010)          1056 TBUFL   EQU     16              ;transm buf length
(0004)          1057 NBUF    EQU     4               ;pool recv buffers
(00F3)          1058 MNREQ   EQU     0F3H            ;monitor packets
(00F7)          1059 MNCNF   EQU     0F7H
(000A)          1060 MXDCE   EQU     10              ;max LAU number
(000B)          1061 NDCE    EQU     11              ;number of LAU's
(000B)          1062 MNADD   EQU     11              ;EMU address
(000A)          1063 MNRQL   EQU     10              ;Mon Req length
(0008)          1064 HEADER  EQU     8               ;min response length
(000A)          1065 TIME    EQU     10              ;time-out (10*100ms)
(0004)          1066 QE      EQU     4               ;queue size
(0008)          1067 QL      EQU     2*QE
(0003)          1068 QMASK   EQU     QE-1
(003E)          1069 PROMPT  EQU     '>'
(0003)          1070 ETX     EQU     03              ;end of text (string)
(000D)          1071 CR      EQU     13              ;carriage return
(000A)          1072 LNF     EQU     10              ;line feed
                1073 ;
(0000)          1074 MODE    EQU     00              ;EMU state flags
(0001)          1075 COM     EQU     01
(0002)          1076 GLB     EQU     02
```

```
(0003)      1077 PRES    EQU   03
(0004)      1078 STPR    EQU   04
(0005)      1079 FIRST   EQU   05
(0001)      1080 INITST  EQU   01          ;MODE=1,OTHERS=0
            1081 ;
(0000)      1082 ONI     EQU   00          ;LAUi state flags
(0001)      1083 SCNI    EQU   01
(0002)      1084 COMI    EQU   02
(0003)      1085 RESI    EQU   03
(0003)      1086 SCONI   EQU   03
(0008)      1087 SCOFFI  EQU   08
(0001)      1088 ONIMSK  EQU   01          ;masks all but ONI
            1089 ;
(0000)      1090 TOUT    EQU   00          ;timer flags
(0001)      1091 TIM     EQU   01
(0002)      1092 TON     EQU   02
            1093 ;
(0006)      1094 PAR     EQU   06          ;message type flags
(0007)      1095 TABLE   EQU   07
            1096 ;
(0000)      1097 RMORE   EQU   00          ;handshake flags
(0001)      1098 EOTR    EQU   01
            1099 ;
(007F)      1100 MMASK   EQU   7FH         ;message mask
(00E0)      1101 PMASK1  EQU   0E0H        ;PIO control masks
(0010)      1102 PMASK2  EQU   10H
(0020)      1103 PMASK3  EQU   20H
(00DF)      1104 PMASK4  EQU   0DFH
            1105 ;
(0007)      1106 SBUF    EQU   07          ;PIO control signals
(0006)      1107 DBUF    EQU   06
(0005)      1108 RDDB    EQU   05
(0004)      1109 WRSB    EQU   04
            1110 ;
(F000)      1111 RAMTAB  EQU   0F000H
            1112 ;
(F000)      1113 BUFAR   EQU   RAMTAB      ;buffer area
(F000)      1114 BUF1    EQU   BUFAR
(F088)      1115 BUF2    EQU   BUF1+BUFL
(F110)      1116 BUF3    EQU   BUF2+BUFL
(F198)      1117 BUF4    EQU   BUF3+BUFL
(F220)      1118 TBUF    EQU   BUF4+BUFL
(F227)      1119 MFIELD  EQU   TBUF+7
            1120 ;
(F230)      1121 QAR     EQU   TBUF+TBUFL  ;queue area
(0000)      1122 RAM1    EQU   [QL-[QAR MOD QL]]MOD QL
(F230)      1123 TPOOL   EQU   QAR+RAM1
            1124 ;
(F238)      1125 RGAR    EQU   TPOOL+QL
(0008)      1126 RAM2    EQU   [16-[RGAR MOD 16]]MOD 16
(F240)      1127 RG1     EQU   RGAR+RAM2
(F240)      1128 STATE0  EQU   RG1         ;LAUi state registers
            1129 ;
(F250)      1130 RG2     EQU   RG1+16
(F250)      1131 STATE   EQU   RG2         ;EMU state register
(F251)      1132 TIMREG  EQU   RG2+1       ;EMU timer register
(F252)      1133 INCNT   EQU   RG2+2       ;EMU counters
(F253)      1134 OUTCNT  EQU   RG2+3
(F254)      1135 RESCNT  EQU   RG2+4
(F255)      1136 RCNT    EQU   RG2+5
```

```
(F256)        1137 NOFB    EQU    RG2+6
(F257)        1138 DCEIN   EQU    RG2+7
(F258)        1139 DCEOUT  EQU    RG2+8
(F259)        1140 CURCMD  EQU    RG2+9          ;command register
(F25A)        1141 PARAM1  EQU    RG2+10         ;param registers
(F25B)        1142 PARAM2  EQU    RG2+11
(F25C)        1143 PARAM3  EQU    RG2+12
(F25D)        1144 TMCNT   EQU    RG2+13         ;timer counter
(F25E)        1145 TI      EQU    RG2+14         ;timer interval reg
(F25F)        1146 LOCADD  EQU    RG2+15         ;EMU address register
              1147 ;
(F260)        1148 RG3     EQU    RG2+16
(F260)        1149 PRMPT   EQU    RG3            ;resp point (display)
(F262)        1150 RCMPT   EQU    RG3+2          ;resp point (receive)
(F264)        1151 RBPNT   EQU    RG3+4          ;recv buffer pointer
(F266)        1152 CURBUF  EQU    RG3+6
(F26A)        1153 DFQ     EQU    RG3+10         ;queue pointers
(F26B)        1154 DLQ     EQU    RG3+11
(F26C)        1155 DPOOL   EQU    RG3+12
              1156 ;
(F270)        1157 RG4     EQU    RG3+16
(F270)        1158 RADTAB  EQU    RG4            ;tab rcv resp addr
              1159 ;
(F290)        1160 RG5     EQU    RG4+32
(0070)        1161 RAM3    EQU    [256-[RG5 MOD 256]]MOD 256
(F300)        1162 RCMSP   EQU    RG5+RAM3       ;response storage area

Errors        0
```

APPENDIX D

LABCRATORY FACILITIES

Fig. D.1 - The laboratory development system

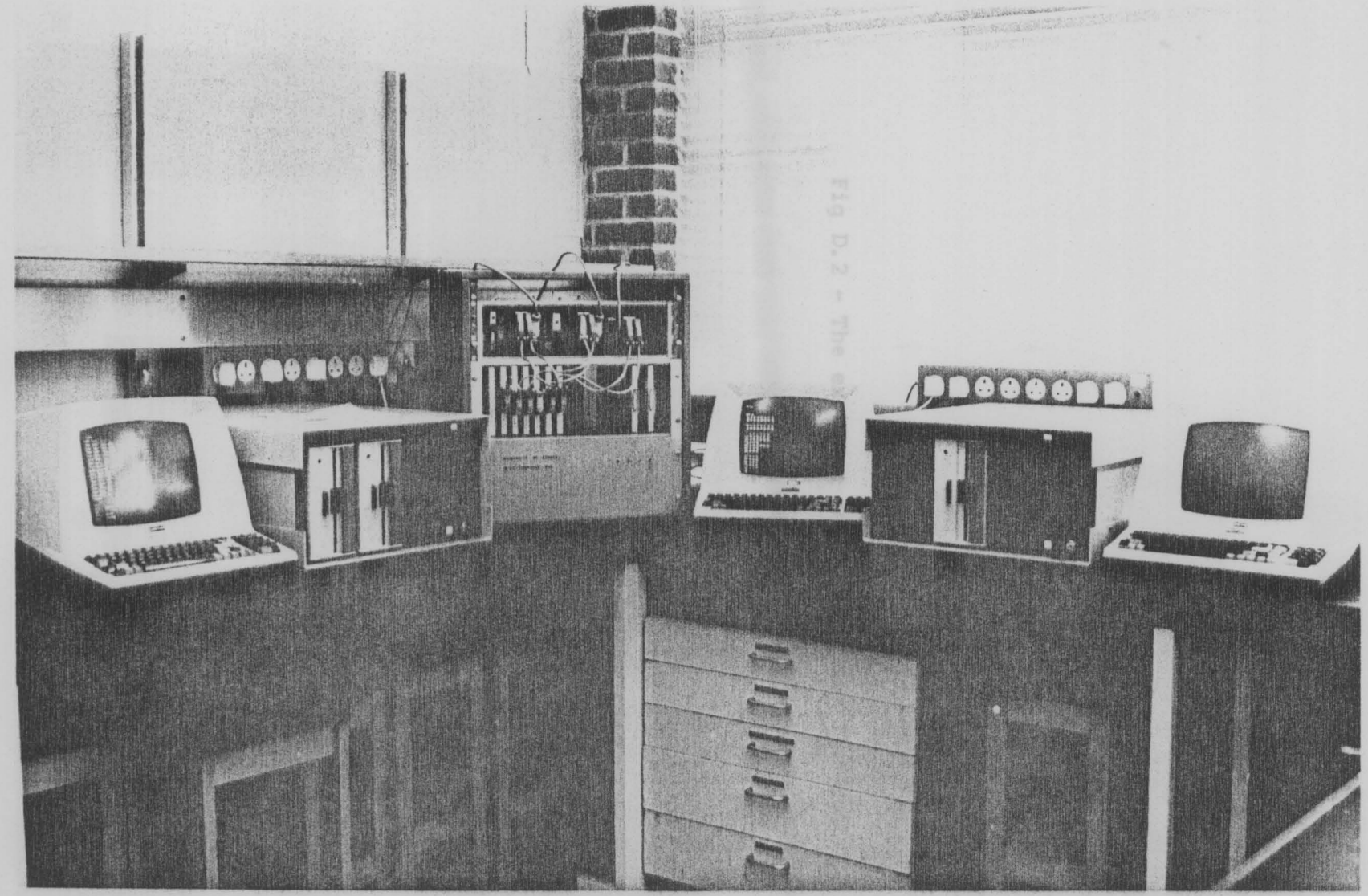
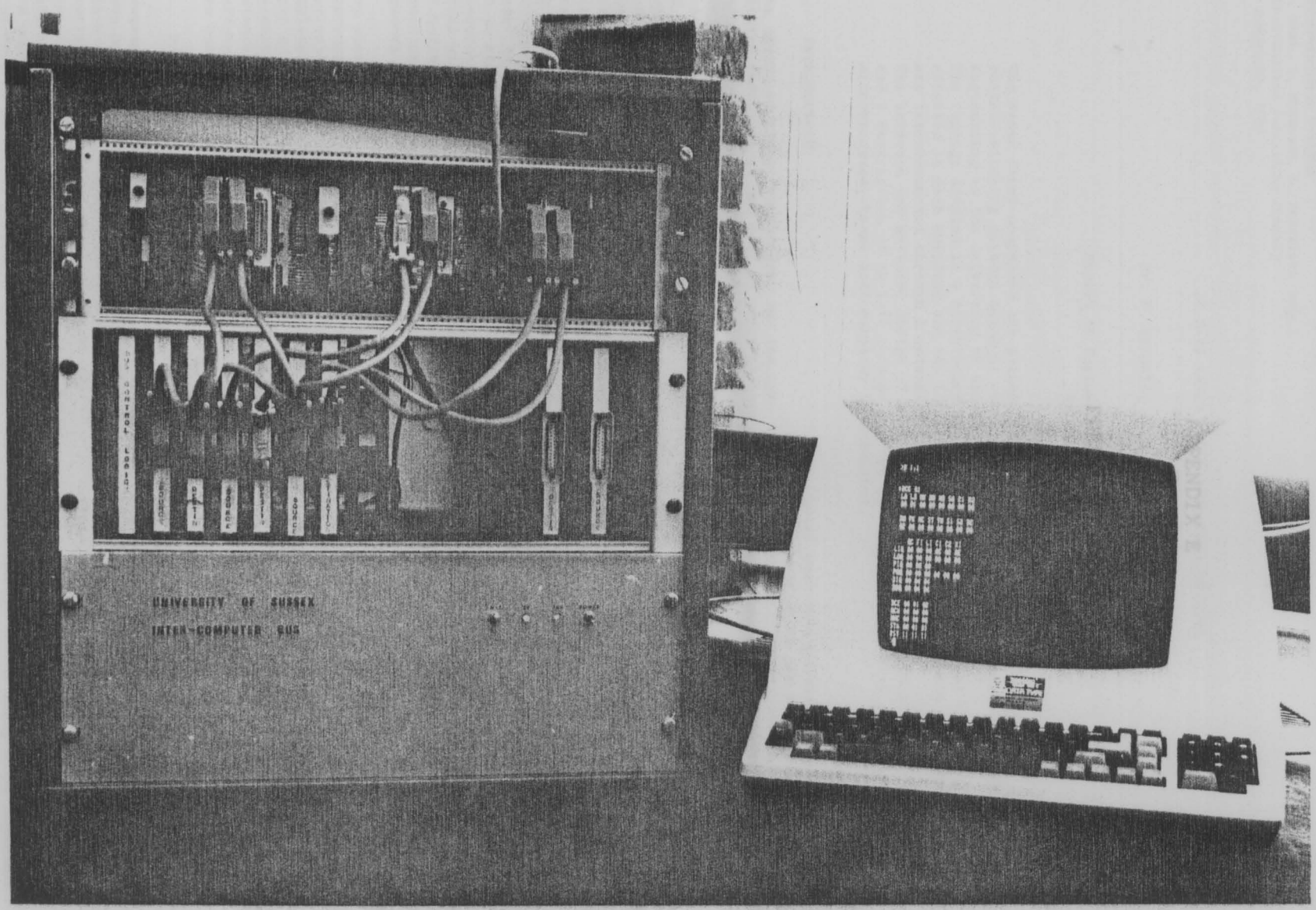


Fig D.2 - The exchange and the monitor terminal


UNIVERSITY OF SUSSEX
INTER-COMPUTER BUS

APPENDIX E

PUBLICATIONS

MICROPROCESSOR SYSTEMS
M. Sami, L. Thompson, L. Mezzalira (eds.)
North-Holland Publishing Company
© EUROMICRO, 1980

AN X-25 BASED LOCAL AREA DATA NETWORK

Dr. F. Halsall, J. A. Ruela and A. Al-Jaff

School of Engineering, University of Sussex,
Brighton, U.K.

The paper first describes the philosophy and topology of an X-25 based local area packet-switching network which is currently being implemented in the School of Engineering of the University of Sussex.
The toplogy adopted is a distributed star structure: a small local 'exchange' is situated in each building and the local exchanges distributed around an establishment are interconnected through a central 'tandem' exchange.
The hardware and software design of a multimicroprocessor local exchange is described and a description of an X-25 package written in Pascal for use with a single-user microprocessor-based host computer is presented.

1. INTRODUCTION

The advent of the microprocessor and the continuing reduction in the cost of computer hardware is having considerable impact on the computing resources found within an establishment. It is rapidly becoming common practice to find distributed throughout an organisation perhaps one or more main frame computers, a number of minicomputers and an increasing number of microprocessor-based systems. Although the availability of processing power within each of these systems means that many computing tasks can now be carried out locally, in many situations there is also a requirement for these systems to communicate with each other in order, for example, to share resources or to access shared information. Since these resources are distributed over a local area - often within a single building or small cluster of buildings - the communication facility provided is referred to as a local area data network.

The paper first presents the philosophy and topology of a local area network which is currently being implemented in the School of Engineering of the University of Sussex. It then describes the design of a multi-microprocessor packet-switch being used within the network and also the design of the interface hardware and driver software used within one of the connected host computers.

2. PHILOSOPHY

Geographically distributed computer networks have now been in existence for a number of years and indeed internationally agreed protocols have been established to control the transfer of data using such networks (1). Local data networks of the type just outlined, however, are comparitively new and perhaps not surprisingly a variety of different protocols and communication network structures are currently being investigated (2, 3, 4).

Local area data networks have a number of characteristics which are similar to those found in geographically distributed computer networks, since each is concerned with providing a communication facility between a number of distributed computers. Moreover, although a local area network may contain a possibly large number of microprocessor-based systems, the inclusion of processing resources within these systems and the continuing reduction in memory costs means they can readily support a computer network protocol.

The protocol adopted for the Sussex network therefore is X-25. This is now the agreed standard for packet-switched networks and many mainframe and minicomputer manufacturers offer X-25 communication packages for their products. Linking machines of this type into the network is therefore a more straightforward task than if a non-standard protocol is used. In addition, commercially available integrated circuits are now becoming available which implement parts of the X-25 protocol (5) and these will considerably simplify the interface hardware required to link a host computer into the network.

3. TOPOLOGY

In general, the topology adopted for local networks is either an extended bus structure - Ethernet, for example - or a ring (loop) structure - Cambridge Ring, for example. Although these topologies have the advantage of low installation costs and, in the case of an Ethernet structure, a passive operating mode, they have an intrinsically low reliability since a break in the communication channel can affect the operation of the complete network.

The topology adopted for the Sussex network, therefore, is a distributed star structure as shown in Fig. 1. A small local packet-switching 'exchange' is situated in each building which links the machines in that location together.

The various local exchanges distributed around the establishment are then linked through a central 'tandem' exchange. The links between the two exchanges also use the X-25 link-level protocol and hence the structure of the tandem exchange is similar to that of a local exchange. The structure is therefore modular and can be readily tailored to meet the requirements of different sized organisations.

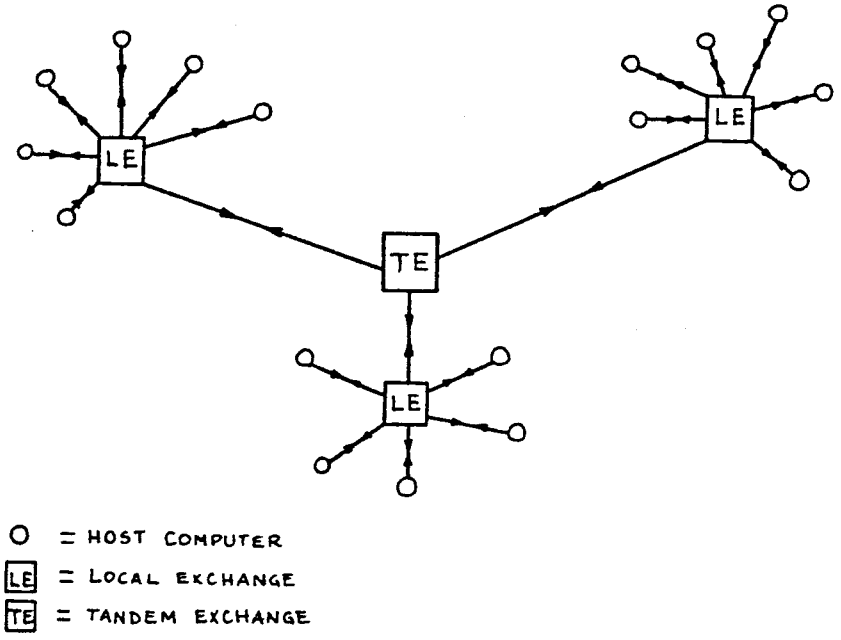


Figure 1: Network Topology

4. EXCHANGE HARDWARE

In the prototype system, an exchange is comprised of up to ten separate microprocessor-controlled channel units - DCE's - with provision for two additional units: one to perform supervisory and control functions and the other to link the exchange to an intermediate tandem exchange. A schematic diagram of a DCE is shown in Fig. 2. It is made up of a Z-80 microprocessor, a quantity of local memory (PROM and RAM), an HDLC serial input-output unit with associated DMA circuit and a bus interface unit.

The DCE's are interconnected by a separate parallel interchannel bus as shown in Fig. 3. As can be seen, loss of a DCE affects only the host computer(s) which is (are) connected to that channel. The bus interface circuitry within each DCE is responsible, under the control of the microprocessor within the unit, for the transfer of all packet data between that unit and the inter-channel bus. In addition, there are two packet buffers at each access port to the bus: one for output and one for input. Thus, whenever a packet is to be transferred from one DCE to another, the microprocessor within the source unit loads the packet data into the output buffer and then initiates a transfer request to the bus control logic. The latter is responsible for multiplexing transfers across the bus and hence, assuming no other transfer is currently in progress and the required destination input buffer is empty (there is a separate control line which indicates the status of each destination buffer on the bus), the packet is transferred across the bus to the destination buffer and subsequently to the destination DCE under control of the microprocessor within the destination unit. In this way the bus occupancy time for any particular packet transfer is kept to a minimum.

Since the bus must handle a variety of packet transfers ranging from short control packets to longer block transfer type data packets, the input and output packet buffers connected to the bus can accommodate a variable packet size up to a maximum of 128 data bytes plus the header bytes used by X-25. Thus, in addition to the source and destination DCE addresses, each packet contains a length byte at its head to indicate to the source and destination DCE's the size of the particular packet being transmitted. This is generated as the frame is being received by the exchange and is inserted at the head of the receiver buffer before the actual frame information. Packet transfers

across the exchange interchannel bus take place at 1.5 Mbyte/s.

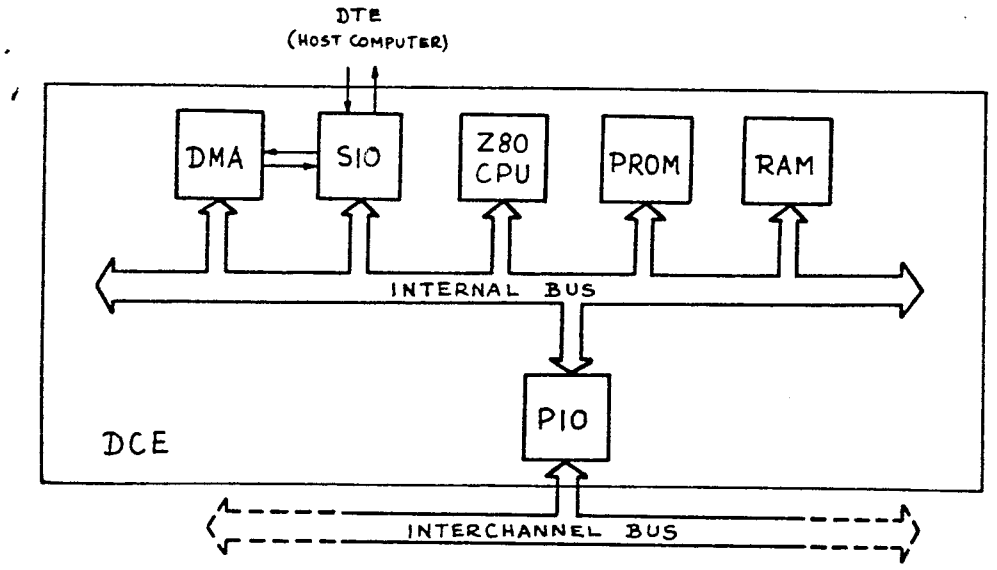


Figure 2: DCE Hardware Structure

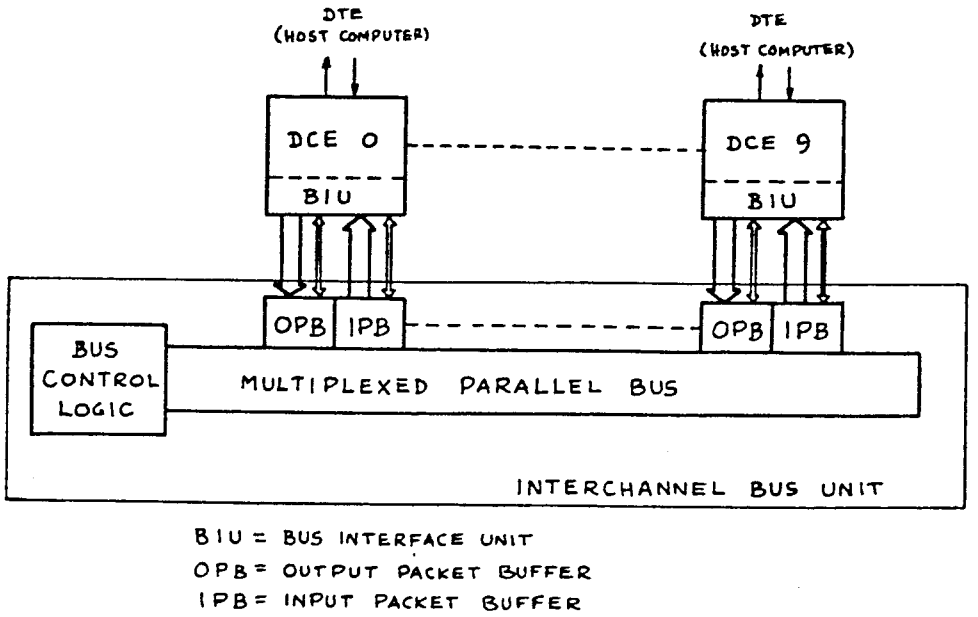


Figure 3: Interchannel Bus Unit

5. EXCHANGE SOFTWARE

The X-25 recommendation defines the interface between data terminal equipment (DTE) and data circuit equipment (DCE) for terminals operating in the packet mode. The DTE is thus the source or sink of packets to or from the network and hence is either a mainframe, mini or microcomputer. Similarly, the DCE is an exchange channel unit and hence X-25 defines the interface between each host computer and the network.

The X-25 protocol is comprised of three levels: level 1 which defines the physical interface, level 2 which defines the link access procedure (LAP) and level 3 which defines the packet level. Each level contains definitions to control the exchange of information between similar levels in the DTE (host) and the DCE (channel unit). This is illustrated in Fig. 4.

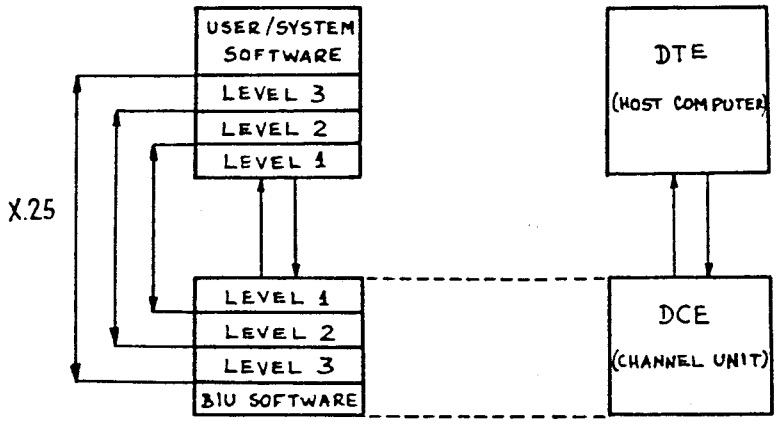


Figure 4: The X-25 Interface

The physical interface effectively provides a synchronous bit serial transmission facility and in the implemented system uses Phase encoding of the serial data. The link access procedure is based on the high-level data link control procedure HDLC. It is often referred to as the link or frame level and the frame format used is as shown in Fig. 5.

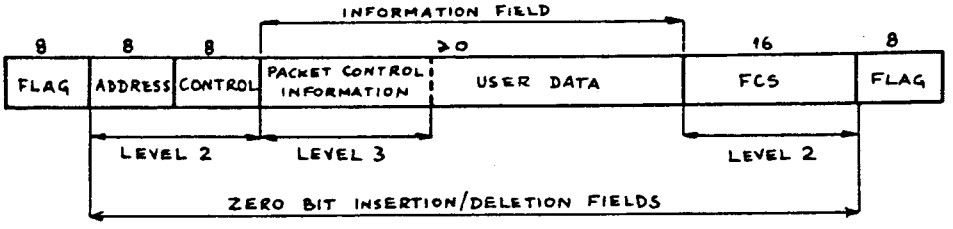


Figure 5: Frame Format

The flag sequence (01111110) defines the frame boundary; the address field is used to distinguish between the type of frames flowing in opposite directions on the link; the control field indicates the frame type and sequence numbering information; the information field is optional since it is used to carry level 3 packet information and the FCS is a 16-bit error frame check sequence. There are no special reserved characters within a frame and hence to avoid the receiver misinterpreting an information byte corresponding to the flag sequence, the transmitter automatically inserts a 0 bit after 5 contigious 1 bits. The receiver then automatically removes a 0 if it occurs after 5 contigious 1 bits. This is known as zero bit insertion and deletion and is performed automatically by the serial link hardware.

The link procedures adopted for the exchange are known as LAP A which means the set-up procedure for a link is performed in both directions once initiated by either the host or the connected DCE. In order to avoid possible deadlock situations, however, information frames may be sent in one direction as soon as the set-up is complete for that direction. The completion of the set-up procedure is thus independent for each direction even though the initialisation is started simultaneously.

In a public packet switched network each information packet may be routed through several intermediate network modes before reaching the

required destination. In such networks level 3 is therefore used to control the flow of packets across each internode link. In our system, however, the network appears like a single packet switching exchange and hence level 3 within the exchange does not perform any flow control: it simply routes data packets without checking or changing their sequence numbers. This avoids the necessity of buffering packets in the exchange and the necessity of receiving and allocating buffers to logical channels. The exchange does, however, perform the detection of protocol errors and initiates appropriate clear procedures if these are detected.

Each link controlled by level 2 is multiplexed by level 3 into a number of logical channels. For a multi-user host this means that different users are allocated different logical channels on the same physical link. For a single-user host this facility is not used. During a call set-up, the source DCE keeps a record of the channel number used on the incoming link in a routing directory and inserts this together with the source DCE address in the packet header - in the internally unused level 2 header field - before forwarding the request to the destination DCE. The latter also keeps a record of the source DCE address and channel number in its own directory together with the channel number used on the destination link. In addition, the destination DCE inserts the outgoing channel number at the head of the call accepted packet before forwarding this to the source DCE. The latter then completes the routing information for the call in its directory ready for the receipt of subsequent data packets.

The basic structure of the software within a DCE is shown in Fig. 6. A single buffer pool common to all levels is used and the various processes communicate with each other by passing the pointers to these buffers between the inter-level queues. The buffer size used is constant and

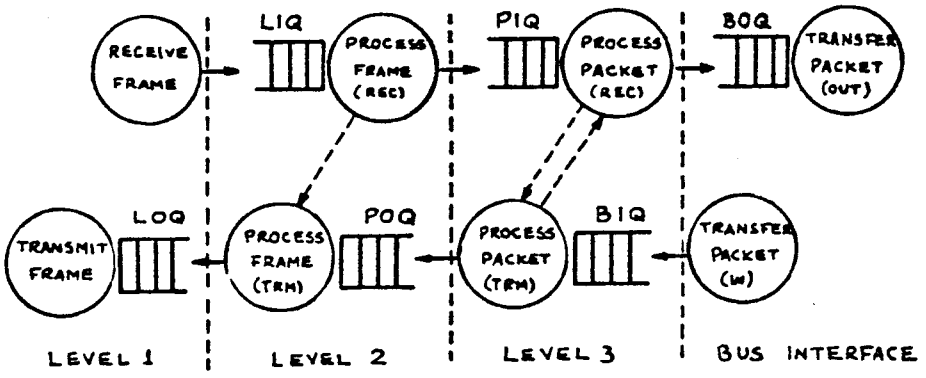


Figure 6: DCE Software Structure

can accommodate the standard data field of 128 bytes plus the header space used by levels 2 and 3. Level 1 processes are interrupt driven and a round-robin scheduling algorithm is used for the other processess. When run, a process accesses a buffer from its input queue (if any available) and performs the corresponding processing before returning to the scheduler.

6. HOST SOFTWARE

As has been mentioned, the main justification of using X-25 as the network protocol is the increasing number of manufacturers who are now offering communication packages based on X-25 as part of their products. In order to gain some experience in the use of X-25, however, and to investigate the feasibility of implementing X-25 on a small, possibly microprocessor-based, single user host system, some work has been carried out on implementing a package for such a system.

In order to achieve maximum portability, the majority of the package being developed is written in the high-level language Pascal with a small piece of interrupt driven code in assembly language to control the serial link interface. Initially no attempt has been made to make the package an integral part of the operating system. The package is simply run from the console whenever the user wishes to

send or receive data via the network.

In addition to the X-25 layers - levels 1, 2 and 3 - the package contains a transport layer which also handles the communication with the user. Effectively the transport layer controls the transfer of complete messages and hence it both fragments messages into multiple packets for transmission by the network and reassembles packets received from the network into complete messages before these are passed onto the user.

In order to avoid duplicate packets at the transport layer level 3, on receipt of a data packet with an out-of-sequence packet number, initiates a reset for the channel which automatically causes the transport level to retransmit the complete message. A typical sequence of packets exchanged between two DTE's to implement a call is shown in Fig. 7.

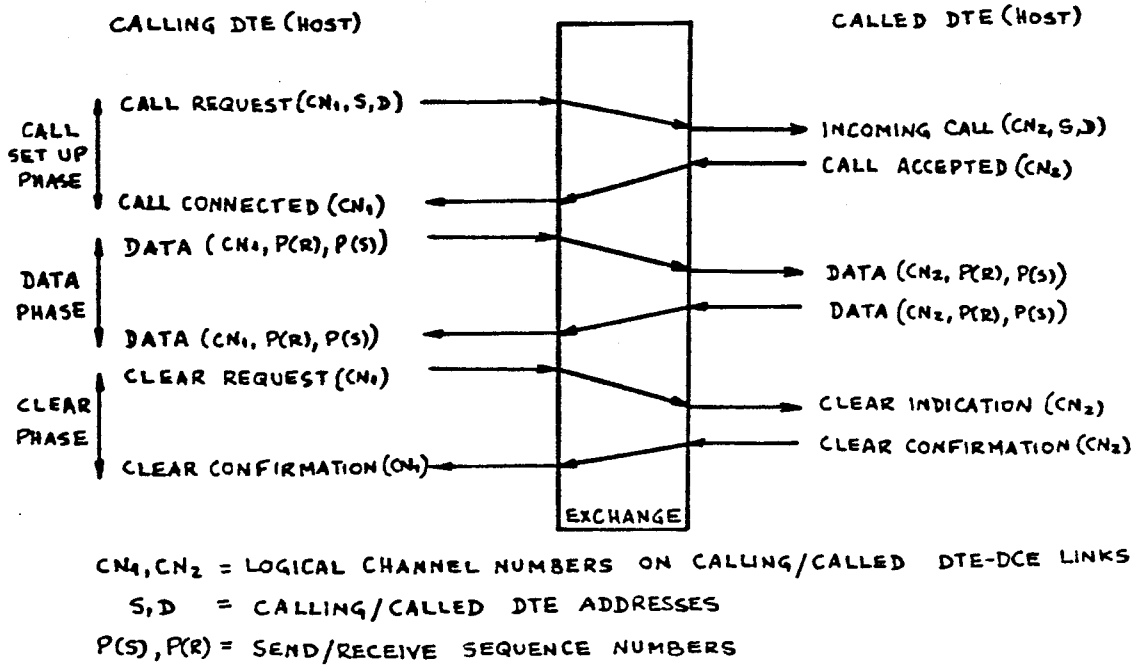


Figure 7: DTE-DTE Packet Sequence

The set of user commands currently being investigated are:

Connect < calling address, called address > - Initiate an X-25 Call Request

Accept < calling address, called address > - Accept a pending Call Request

Disconnect - Clear an existing connection

Transfer < Text string > - Send the text string on an already set-up channel

Receive - Display on the screen text received on an existing channel

The host systems currently being used for development purposes are an LSI-11 system and a Z-80 based S-100 system both with dual floppy discs. The LSI-11 is interfaced to the exchange (network) using a DUP-11 synchronous interface card and a similar S-100 compatible card is used for the Z-80 system.

7. CONCLUSIONS

This paper has presented the design and implementation details of an X-25 based packet-switched local area data network. The widespread acceptance of the X-25 protocol for many public packet-switched networks has resulted

in the availability of a range of LSI circuits and software packages from a number of manufacturers and as this trend continues, it is felt a local network based on X-25 will have increasing attractions.

8. REFERENCES

(1) CCITT Recommendation X-3, X-25, X-28 and X-29 (approved October 1977), Geneva, Switzerland.

(2) "Ethernet: Distributed Packet Switching for Local Computer Networks", R. M. Metcalf and D. R. Boggs, Comm. ACM July 1976, Vol. 19, No. 7, pp 395-404.

(3) "The System Architecture of the Distributed Computer System - The Communications System", D. J. Farber and K. C. Larson, Proc. Sym. on Computer Comms. Nets., Polytechnic of Brooklyn, April 1972.

(4) "The Cambridge Ring for Local Networks", Computer Laboratory, University of Cambridge, U.K.

(5) "LSI Ready to Make a Mark on Packet-Switching Networks", G. L. Leger, Electronics, December 20, 1979, pp 89-99.

Prototype X25 exchange for use in local area networks

F. Halsall, M.Sc., D.Phil., C.Eng., M.I.E.E., and
J.A. Ruela, E.E.

Indexing terms: *Computer networks, Multiprocessors, Packet switching*

Abstract: The paper is concerned with an investigation into the role of the CCITT standard protocol X25 in a local area network environment. The paper first describes a suitable topology for such networks, comprising a high-level network incorporating an X25-based packet-switching exchange. The design and implementation details of a multiple processor packet switch are then described and performance details of the implemented system are presented.

1 Introduction

Although off-the-shelf local area data networks are relatively new, large geographically distributed (wide area) networks have now been in existence for a number of years, and, indeed, internationally agreed protocols have been established for using such networks [1]. Moreover, although there are often considerable differences between the geographical separation of the computing resources found in a local area network and those in a wide area network, as each is concerned with providing a communications facility between a community of distributed computers and related devices, the requirements for both types of network have many similarities. The purpose behind the work described in this paper, therefore, was to investigate the role the standard protocols now being used for wide area networks might have in a local area networking environment. The paper first describes a possible topology for a local area network and then describes the design and implementation of a microprocessor-based packet-switching exchange which is suitable for use in such a network.

2 Network requirements

Before the topology or mode of operation of any network can be defined, it is essential that the requirements of the network are first analysed. Clearly, although these will vary in terms of the number and distribution of specific computers and devices from one network to another, in general, the requirements of a typical local area network can be summarised as in Fig. 1.

A primary requirement for many local area networks is to provide a communications facility for a number of large, possibly dissimilar, mainframes and/or minicomputers, either to allow communications between each other or to allow other network devices access to them. Secondly, the provision of a switched connection between a distributed community of unintelligent terminals and a number of large machines may be a central requirement for many networks.

Thirdly, the proliferation of large numbers of small single-user computer systems – word processors, intelligent workstations etc. – means that there is likely to be a requirement for a communications facility between such systems and other network resources. It is probable, however, that this will be concerned primarily with sharing a local resource – file station, printer etc. – between a localised community of similar systems, and only a relatively small number of transactions will require network-wide communications.

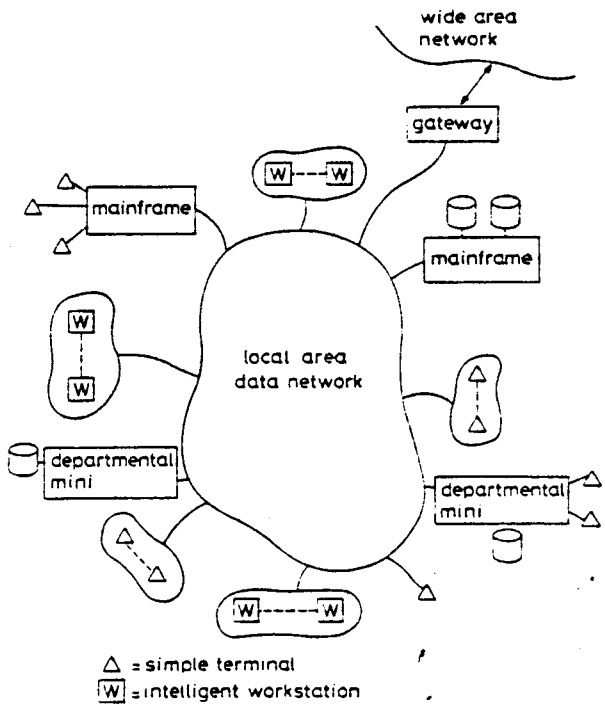


Fig. 1 Typical network requirements

Finally, there is possibly a requirement for a communications facility between some local devices and one or more external public networks. For example, for a large mainframe to submit or receive transactions to or from a remote centre. This implies the provision of some form of internetwork gateway to handle the possibly different protocols and line baud rates.

Paper SM14, received 29th January and in revised form 18th February 1982
The authors are with the School of Engineering & Applied Sciences, University of Sussex, Falmer, Brighton BN1 9QT, England

0261-3182/82/03069 + 05 $01.50/0

3 Proposed topology

It can be concluded from the preceding Section that the requirements of a local area network can be extremely varied. Thus, although a number of different network topologies have now been proposed – Ethernet [2], Cambridge Ring [3] etc. – it seems unlikely that a single type of network will be able to satisfy all the requirements of a typical network. Moreover, although it may be relatively straightforward to write a new communications package for a small single-user computer system, and integrate this into its operating system, the same does not apply for a large mainframe or minicomputer. Indeed, the existence of one or more such systems in a network creates perhaps the most demanding requirements for the network designer.

The proposed topology, therefore, is a hybrid network as shown in Fig. 2. As can be seen, a central component of the proposed architecture is an X25-based packet-switching exchange. This is now the internationally agreed standard for connecting packet-mode devices (computers etc.) to a data network, and is therefore supported by most computer manufacturers. The other network requirements are then met by providing a range of additional components.

Thus, unintelligent character-mode devices (VDUs) are integrated into the network by means of packet assembly/disassembly devices (PADs). A PAD appears to the exchange, therefore, as a multiple-call packet-mode device, and so can be situated either locally with a group of terminals, and hence remote from the exchange, or within the exchange itself. The latter, however, would clearly require separate communications lines between the exchange and each terminal device, and so, wherever possible, a remotely situated PAD would seem advantageous.

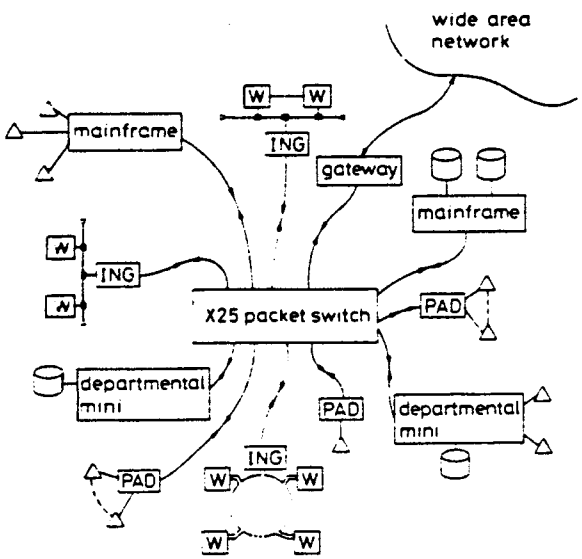


Fig. 2 Proposed network topology
ING = internetwork gateway
PAD = packet assembler/disassembler

As has been mentioned, the networking requirements for the possibly large number of local communities of small microprocessor-based systems are made up of two components: the first for access to some locally held shared resource, e.g. printer, file server etc., and the second for access to a network-wide resource, e.g. a shared data base. Thus, since it seems likely that for this type of network user the first requirement will predominate, it is proposed that these requirements are met by providing a number of separate local networks, based, say, on Ethernet, Cambridge Ring, etc., to satisfy the local communications needs of the community, with a small internetwork gateway to handle any network-wide communications. Again, these gateways are likely to be situated with the local community of users, and so remote from the exchange.

Finally, to enable a user of the network to communicate with users/systems of other external networks – public or private – the exchange includes an internetwork gateway interface. Typically, this would provide a user with access to a separate gateway unit which supports a network-independent message-transport service [4], and will therefore be transparent to the main packet-switching exchange. Clearly, however, if the two networks operate using the same protocol, its implementation will be much easier.

4 Multiprocessor packing-switching exchange

A central component of the proposed network structure is an X25 based packet switching exchange. Clearly, a number of such exchanges are now in existence in many of the large public packet-switching networks currently in operation. The requirements of an exchange in a public network, however, are significantly different from those for an exchange in a local area network; e.g. in terms of throughput, call charging, reliability etc. An outline design for an exchange suitable for use in a local area network environment has been presented by the authors elsewhere [5]. A prototype exchange has now been built, however, and so this Section, after briefly describing the design philosophy, presents implementation and performance details of the implemented system.

4.1 Design philosophy

The design aims for an exchange can be summarised as follows:

Modular construction: Since the required number of exchange lines will vary considerably from one network to another, the exchange must be modular and capable of incremental growth.
Standard access protocol: The exchange must present a standard X25 access protocol, and so be compatible with manufacturers networking hardware and software.
Good reliability: Failure of a single unit should not cause a catastrophic exchange failure, but rather a degradation of the services offered, and so overall performance.
Self-monitoring: The current status of the exchange should be continuously monitored and, for diagnostic purposes, faulty components readily determined.
Low cost: To be competitive with other local networking solutions, the exchange should have only a modest cost per line connected.

In an attempt to meet these aims, the design adopted is shown in Fig. 3. It comprises a readily extendable number of link access units (LAUs) which perform the basic call-processing functions of the exchange, an internetwork gateway interface (IGI), to provide for communications with other wide area networks, and an exchange management unit (EMU) for monitoring and diagnostic purposes.

Each link access unit can support a number of full-duplex links – determined by the collective baud rate of the links – and each presents a standard X25 interface to

the connected device. The exchange management unit can, under the control of an operator, collect data relating to specific calls, monitor the performance of the exchange and

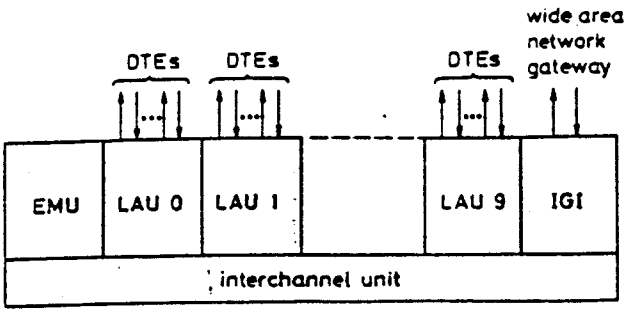


Fig. 3 **Exchange design**
EMU = exchange management unit
DTEs = data terminal equipments (mainframe, mini, PAD, ING)

indicate system fault conditions. The above units are then interconnected by means of an autonomous inter-channel parallel bus or highway. The design of each unit is now considered.

4.2 *Link access unit*

The organisation of a link access unit is shown in Fig. 4. Each unit has its own internal bus to which is connected a Z80 microprocessor, a readily varied quantity of PROM and RAM to hold the X25 control program and its associated tables and packet buffers, a counter timer to perform the time-out functions associated with the link protocol, one or more data-link controller (SIO) cards, which form the interface between the unit and the serial transmission lines, and an interunit interface (IUI) card which forms the interface between the unit and the interunit highway.

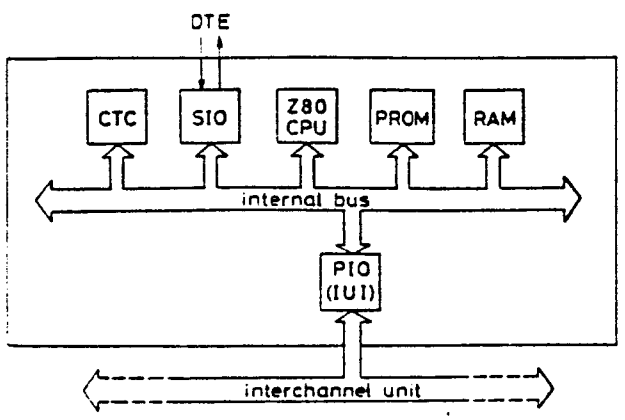


Fig. 4 **Schematic diagram of link access unit**

The X25 call-processing software comprises essentially four processes which communicate with each other through a set of queues. The processes are scheduled in a round-robin manner, and, when scheduled, a process will normally process just a single queue entry. The four processes are:

Link handler: Each serial link is interrupt-driven, indicating the reception/transmission of a byte of a frame, end of frame etc., and this process services these interrupts.

Frame handler: This process performs the X25 level-2 processing. It contains procedures for the setting up and disconnection of links, acknowledging received frames, handling retransmissions etc., according to the X25 link-level protocol.

Packet handler: This process performs the X25 level-3 processing, and comprises a number of procedures which process the different level-3 packet types. The current state of each X25 logical channel is contained in an associated event/state-transition table. Thus, on receipt of a packet, the corresponding event/state-transition table for that channel is consulted to determine both the processing action required for the packet and also the resulting new state for the channel.

IUI handler: This process controls the transfer of data between the internal packet buffers of the unit and the source and destination buffers of the interchannel unit (see below).

In addition, there is a global procedure – the buffer manager – which is used by each of the above processes to manage all the free frame/packet buffers of the unit, and an interrupt-driven process – the timer manager – which handles all the various time-out intervals associated with each process level. These include a timeout at the frame level to initiate frame retransmission and link-failure detection, at the packet level to initiate individual or global call clearance, and at the interunit interface to indicate failed link access units.

4.3 *Exchange management unit*

The exchange management unit does not play any part in the normal call-processing activities of the exchange, but rather allows an operator to monitor the performance of the exchange by entering specific commands at the operator's console. A range of different commands may be entered, and typically these initiate the sending of a corresponding message type to the specified link access unit. Each command specifies a specific action to be carried out; the specified link access unit then responds either with the appropriate response, e.g. its current status, or an acknowledgment of correct receipt, e.g. to a command to start tracing the activity on a link.

The complete list of commands and their functions are listed in Fig. 5*a*. For simplicity, each command is specified using a single character and as an example, a typical set of commands and their responses are shown in Fig. 5*b*. This shows logical channel 07 in LAU 1 and logical channel 01 in LAU 2 being used for a call which is currently in the data-transfer phase (04). All the remaining channels are in the ready state (00).

4.4 *Internetwork gateway interface*

The internetwork gateway interface provides a communications facility between the local network and an external internetwork gateway unit. In the implemented exchange, therefore, the interface is simply an additional link access unit which recognises and routes packets addressed to external DTEs and any necessary protocol conversion procedures, and, when appropriate, flow control functions are performed in the external gateway unit.

The interchannel unit is a separate autonomous unit which provides a fast (1·8 Mbyte/s) communications facility between the various units which are connected to it. Essentially, it comprises a number of full-duplex parallel ports, each supporting one input and one output channel as shown in Fig. 6. Each channel contains a separate first-in first-out (FIFO) packet buffer and an associated control

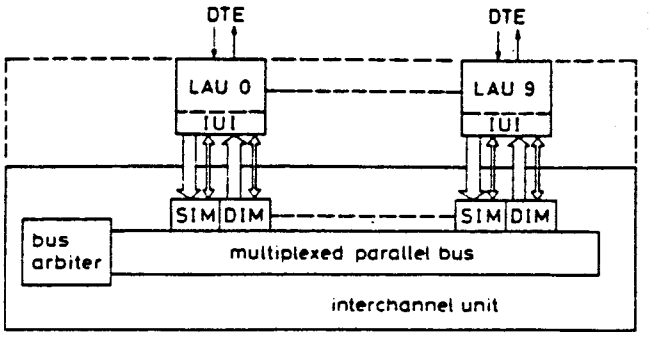


Fig. 6 **Schematic diagram of interchannel unit**
IUI = interunit interface
SIM = source interface module
DIM = destination interface module

P p1	Poll the specified LAU to determine its current state (active/failed)
I p1	Initialise the specified LAU; that is, clear all calls currently in progress
S p1	Request the specified LAU to store all the current states of its call processing variables in its internal buffer storage area
D p1,p2,p3	Retrieve and display either the stored (p2=0) or the current (p2=1) state of the call processing variables relating to the specified LAU with options as specified by p3
A p1, p2	Requests the specified LAU to trace all Activity on a link for p2 seconds and to store the sequence in its internal buffer storage area
T p1,p2	Trace either the stored (p2=0) or current (p2=1) sequence of activity on the specified LAU link
R p1,p2	Display either current (p2=0) or total (p2=1) statistical records (number of calls, information frames, data packets, time-outs etc. gathered by the specified LAU
	p1=specified LAU address

a

```
X25

>A 1, 20          ;initiate trace activity in LAU 1 for 20 seconds

*LAU 01
CMND AC           ;EMU command accepted

>T 1·0            ;trace stored activity on LAU 1

*LAU 01
RSAR              ;SARM frame received (link set-up)
SUA               ;generate UA frame
SSAR              ;generate SARM frame (link set-up)
TRES CF 63        ;UA frame (CF=63) transmitted
TCOM CF 0F        ;SARM frame (CF=0F) transmitted
RUA               ;UA frame received
RINF CF 00        ;INFORMATION frame (CF=00) received
SRR CF 21         ;generate RECEIVER READY frame (CF=21)
CREQ CH 07        ;CALL REQUEST (CH=07) received from local DTE
TRES CF 21        ;RECEIVER READY frame (CF=21) transmitted
CCON CH 07        ;CALL CONNECTED (CH=07) received from remote
                  DTE
SINF CF 20        ;generate INFORMATION frame (CF=20)
TINF CF 20        ;INFORMATION frame (CF=20) transmitted
RRR CF 21         ;RECEIVER READY frame (CF=21) received

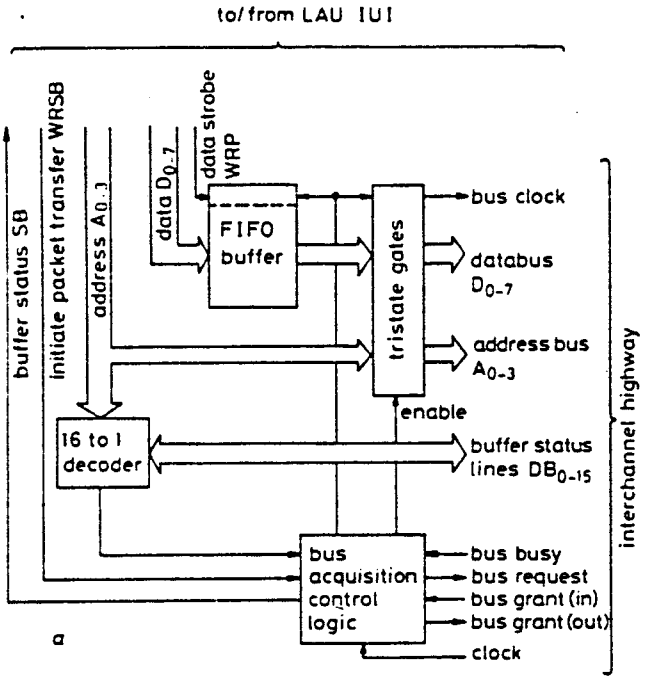
>D 1,0,1          ;display the current state of calls in LAU 1

*LAU 01

CNR 00 01 02 03 04 05 06 07   ;Eight Logical Channel numbers
CST 00 00 00 00 00 00 00 04   ;Channel state (00=ready; 04=data)
LAU 01 01 01 01 01 01 01 02   ;Remote LAU address
RCH 00 00 00 00 00 00 00 01   ;Remote LAU channel number
```

b

Fig. 5 **Exchange management unit functions and their uses**
a List of EMU commands and their functions
b Typical dialogue with EMU



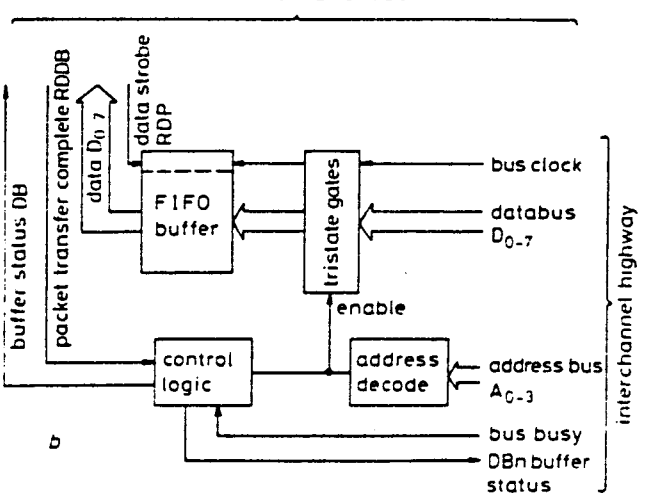


Fig. 7 **Schematic diagram of source and destination interface modules**
a Source *b* Destination

line indicating its current status – full/empy – as shown in Fig. 7.

Thus, whenever a packet is to be transferred from one unit to another, the IUI handler process within the source unit first tests the status of the channel input buffer and, assuming the buffer is empty, loads the packet data into the buffer, sets the required output port address on the address bus and initiates a transfer request. The control logic within the source channel unit then waits until no other transfer is currently in progress across the highway – BBUSY inactive – and the required destination channel output buffer is empty – DBn high – before making an inter-unit transfer request to the bus arbiter. The latter is responsible for multiplexing packet transfers across the highway and. on receipt of the request, it issues a bus grant signal. Then, assuming the source unit is the highest priority requestor, on receipt of the grant signal it enables the required destination channel address onto the address bus and proceeds to transfer the packet contents across the bus to the addressed channel output buffer using the bus clock. On completion, the source buffer status becomes empty and the destination buffer full and, when this is detected, the IUI handler process within the destination link access unit transfers the packet data into an internal buffer for further processing. A timing diagram illustrating a typical packet transfer using the interchannel unit is shown in Fig. 8.

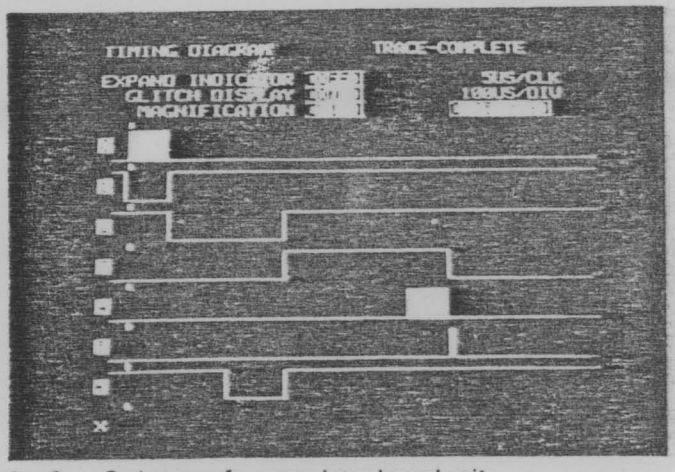


Fig. 8 Packet transfer across interchannel unit

0	WRP	Data strobe (input)
1	WRSB	Initiate package transfer
2	SB	Source buffer status (0=full, 1=empty)
3	DB	Destination buffer status (0=empty, 1=full)
4	RDP	Data strobe (output)
5	RDDB	Packet transfer complete
6	BBUSY	Bus state (0=busy, 1=idle)

4.6 Performance details

These. performance details have been derived based on the following assumption. First, as has been indicated, each serial link is interrupt driven, and so each byte transmitted or received by an SIO requires on average 35 μs to service. Also, packet transfers to and from the interchannel unit are performed under CPU block-mode control, which results in a transfer rate of 5 μs per byte. Secondly, it is assumed that the traffic in each serial link is balanced corresponding to the most demanding processing overheads and the most efficient use of the links. Thirdly, since the maximum length of user data supported is 128 bytes, it is assumed that all frames/packets have an average length in excess of 50 bytes.

Based on these assumptions, a single link access unit can support a number of full-duplex links which have a collective baud rate of up to 50 Kbaud. Hence, it could support either single 48 Kbaud link or a typical mix of links of, say, two at 9·6 Kbaud, four at 4·8 Kbaud and eight at 1·2 Kbaud.

Also, assuming negligible queue delays, a packet experiences a time delay of approximately 2 ms while within the exchange, i.e. the time delay between the last byte of a packet being received on an incoming link and the first byte of the packet being transmitted on the required outgoing link is approximately 2 ms. This includes the total processing time within each link access unit and also the necessary interunit transfer times. Hence, assuming an average packet length of 50 bytes, a link access unit is able to process 100 packets per second in each direction which, for an exchange comprising ten link access units, allows for a total input rate of 1000 packets per second.

5 Conclusions

Although market forces are likely to result in an establishment installing one or more currently available network types, the lack of any agreed software standards for use with these networks will mean that their usefulness and growth potential will be limited. The topology proposed in this paper, therefore, is intended to alleviate this situation by providing a higher-level network which operates using internationally agreed standards. Thus, a range of computers from a number of different manufacturers can be readily connected to it, and, by means of simple internetwork gateways which adhere to the standard, users of one network type can access and use resources on another.

The design and implementation details of the packet-switching exchange that has been described demonstrate that an inexpensive modular design based on microprocessor technology can be used for this function. The performance figures show that a single link access unit can handle a number of full-duplex links which are probably sufficient to meet the requirements of a small network. The use of an interchannel unit of the type described, however, coupled with the exchange management unit, provides a modular design with good diagnostic facilities.

6 Acknowledgments

J.A. Ruela holds a scholarship awarded by the Fundação Calouste Gulbenkian, Lisbon, Portugal, and the authors wish to acknowledge this financial support.

7 References

1 'CCITT recommendations X.3, X.25, X.28 and X.29'. Geneva, Switzerland, 1981

2 'Ethernet: a local area network – data link layer and physical layer specifications'. Digital, Intel and Xerox Corporations, Versions 1·0, Sept. 1980

3 WILKES, M.V., and NEEDHAM, R.M.: 'The Cambridge model distributed system', *Oper. Syst. Rev.*, 1980, 14, pp. 21-29

4 'A network independent transport service'. Yellow Book PSS Study Group Three, Feb. 1980

5 HALSALL, F., RUELA, J.A., and AL-JAFF, A.: 'An X.25 based local area data network'. Euromicro 80, London, pp. 271-277

BIBLIOTECA F.E. PORTO

FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO
BIBLIOTECA
0000006206