Interprocess communication in a multiprocessor environment; specification of a communication controller

Vos, H.J.M.

Award date:
1990

EINDHOVEN UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF ELECTRICAL ENGINEERING
DIGITAL SYSTEMS GROUP (EB)

INTERPROCESS COMMUNICATION IN A MULTIPROCESSOR ENVIRONMENT; SPECIFICATION OF A COMMUNICATION CONTROLLER

by Herman J.M. Vos

Master Thesis
January 1987 - December 1987

supervisor : ir. A.C. Verschueren
supervising professor: Prof. ir. M.P.J. Stevens

Eindhoven, Netherlands

ABSTRACT

The report contains the results of a study of Interprocess Communication systems. Relations to the Open Systems Interconnection Reference Model of the International Standards Organisation are given. As appears from this model, an Interprocess Communication system has to implement the lowest four layers of this model. The user, which is here a multitasking co-processor, has to realise the highest three layers.

The lowest layers of the Reference Model define the network type and its protocol. From the available Local Area Network specifications, the Token Ring Medium Access Control mechanism turned out to be the most efficient one. This mechanism therefore was adopted and modified as far as possible with the international standard.

Also the higher layers (LLC through Transport layer) are discussed as well as the network management. It appeared that the Data Link layer should be connection-less (Class I LLC) and the Transport layer should be a Class 4 Transport layer.

For the purpose of communication between LAN's, bridges are introduced. The extensions to the communication controller are evaluated and turn out to be minor.

Finally, the use of the available (on-chip) hardware resources is discussed and described in an abstract way.

ACKNOWLEDGEMENTS

At this place I would like to thank everyone who helped me during my master thesis project for their ideas and their patience. Special thanks go to:

Prof. ir. M.P.J. Stevens
ir. A.C. Verschueren
ir. J.M.V. Daanen
ir. P.L.H.M. Nissink
Frank Westgeest

CONTENTS

ILLUSTRATIONS AND TABLES

Figure

tables

LIST OF ABBREVIATIONS

AC	Access Control
ANSI	American National Standards Institute
BER	Bit Error Rate
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
DA	Destination Address
DSAP	Destination Service Access Point
ED	Ending Delimiter
FC	Frame Control
FCS	Frame Check Sequence
FS	Frame Status
IEEE	the Institute of Electrical and Electronics Engineers
IPC	Interprocess Communication
ISO	International Standards Organisation
LAN	Local Area Network
LLC	Logical Link Control
MA	My Address
MAC	Medium Access Control
MAP	Manufacturing Automation Protocol
MMTCP	Multiprocessor Multitasking Co-processor
MMU	Memory Management Unit
NM	Network Manager
NMT	Network Management
OSI	Open Systems Interconnection
P	Priority
Pm	PDU Priority
Pr	Received Priority
PCPL	Poll Complete
PDU	Protocol Data Unit
PHY	Physical
R	Reservation
Rr	Received Reservation
REM	Ring Error Monitor
RPS	Ring Parameter Server
RUA	Received Upstream neighbour's Address
Sr	Highest Stacked Received Priority
Sx	Highest Stacked Transmitted Priority
SA	Source Address
SAP	Service Access Point
SD	Starting Delimiter
SDU	Service Data Unit
SFS	Start of Frame Sequence
SSAP	Source Service Access Point
SUA	Stored Upstream neighbour's Address
SV	Subvector
SVI	Subvector Identifier
SVL	Subvector Length
SVV	Subvector Value
TAM	Timer, Active Monitor
THT	Timer, Holding Token

TMU	Timer Management Unit
TNT	Timer, No Token
TOP	Technical and Office Protocols
TRR	Timer, Return to Repeat
TSM	Timer, Standby Monitor
TQP	Timer, Queue PDU
TVX	Timer, Valid Transmission
UNA	Upstream Neighbour's Address
VI	Vector Identifier
VL	Vector Length

1. INTRODUCTION

At the moment a co-processor for multitasking support in hardware is being developed at the Digital Systems group. This co-processor supports some functions for multiprocessor systems too. Someone who is interested in the background of this project should read reference [1]. Applications of an Interprocess Communication (IPC) system are given in [2].

First we will investigate how processes, running on different systems, communicate and how this communication is standardised by the International Standards Organisation (ISO). An indication will be given how their Open Systems Interconnection (OSI) Model can be used in our IPC system. Also, advanced networking properties (like bridges) will be discussed.

The next topic to be discussed is the structure of the IPC system in a system with a Multiprocessor Multitasking Co-processor (MMTCP). Once this is accomplished, we will choose from a few suitable network types the one best suited to our IPC system.

After we have chosen the network type a specification of (the lowest layers of) the system should be given. This can be done in two ways:

- a specification in a verbal way; we use a standard, where some additions and explanations will be given. The purpose of this specification is to have a readable specification of the system, from which we can extract a formal specification.

- a specification in a formal language will be given after the 'informal' specification is completed. This specification has two purposes: to simulate the system and to prove that the specification is complete and correct. A formal specification can be given in some computer language.

Only the verbal specification of the Token Ring mechanism will be given here. Furthermore we will discuss which higher layer(s) we will adopt.

Finally, we will investigate how the communication controller behaves in the MMTCP design; the interactions with the other building blocks need to be examined.

2. RELATION OF THE MMTCP TO THE ISO/OSI REFERENCE MODEL

2.1 The OSI Reference Model

The OSI Reference model is a conceptual network structure defined by the ISO. The purpose of the OSI model is to promote the development of worldwide data-communications standardisation ([3]).

Networks are partitioned into a series of layers to reduce their design complexity. These layers are hierarchical with each layer built upon its predecessor, thus shielding each layer from the details of how the services from the other layers are implemented. The OSI model is partitioned into seven layers. The bottom four layers of the model define the network and how it functions, and the top three layers define how the network is used. The seven-layer OSI model is shown in figure 2.1.

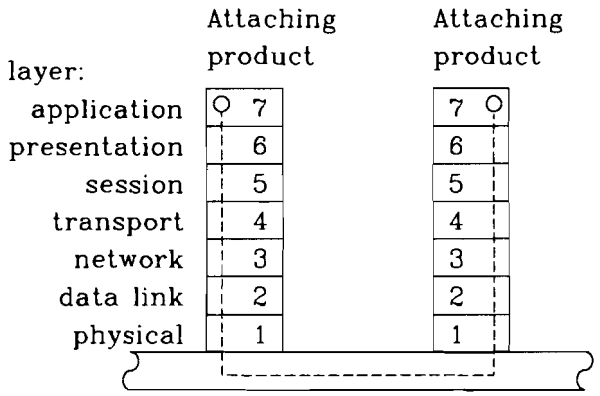


figure 2.1 : The OSI Reference Model

The illustration shows the relationship of the seven layers to one another and the path taken by communication between two attaching products.

The function of the layers are described here in order, starting with the highest.

* The **application** layer (layer 7) provides network-based services to an end user's application programs. It defines how processes communicate with one another and how they make use of these services. The boundary between the presentation layer and the application layer separates the domain of the network designers from the domain of the network users.

* The **presentation** layer (layer 6) provides the user's process with certain useful but not always essential services. Among these services are

cryptographic transformations, text compression, terminal handling, and file transfer.

* The **session** layer (layer 5) is the user's interface into the network. With this layer the user must negotiate to establish connection with a process on another machine. Once the connection has been established, the session layer can manage the dialogue in an orderly manner, if the user has requested that service.

* The **transport** layer (layer 4) accepts data from the session layer, splits it up in smaller units, if required, passes these to the network layer, and ensures that all pieces arrive correctly at the other end. Furthermore, it must be done in the most efficient way.

* The **network** layer (layer 3) controls the operation of the subnet. Among other things, it determines how packets (the units of information exchanged between network layers) are routed within the subnet. The layers task is basically to accept messages from the source host, convert them to packets, and see to it that the packets are directed towards its destination.

* The **data link** layer (layer 2) defines the way data is formatted for transmission and how access to the network is controlled. Since layer 1 merely accepts and transmits a stream of bits without any regard to meaning or structure, it is up to the data link layer to create and recognise frame boundaries.

* The **physical** layer (layer 1) provides mechanical, electrical, functional, and procedural characteristics to establish, maintain, and release physical connections between data link entries.

2.2 Network Services Provided to the MMTCP

In this section, we will indicate the contents of each layer of the ISO/OSI model and its relation to the MMTCP. As in the previous section, we will start with the highest layer.

* The **application** layer receives a command from a MMTCP subsystem, controlling a process. This layer will describe the communication protocol between two processes, running on (different?) machines. It passes the data transmitted by the process to the presentation layer. In the same way data shall be accepted from the presentation layer and passed to the user's process, if required.

* The main function of the **presentation** layer is the handling of file transfer actions. Its protocol can be invoked in two different ways. The process invoking the file transfer wishes to move a file between the local host and a remote one, or it wishes to direct a transfer from a remote host to another. Commands telling what to transfer and where to, may be sent over different connections than those used by the data. Other possible presentation layer services previously mentioned (cryptography, text compression, and terminal handling) are less important for use in the MMTCP.

* The **session** layer establishes the connection between two processes. It checks the identification of processes and their right to engage in the session. Another function of the session layer is management of the session once it has been set up. For example, if transfer becomes unreliable, the session layer may be required to attempt to recover from broken transport connections.

* The **transport** layer creates a distinct network connection for each transport connection required by the session layer. Together with the underlying layers, it provides an error free point-to-point channel that delivering messages in the order in which they were sent. Depending on the quality of the data link service provided to the network layer, a more or less complex transport protocol may be needed. For example, a connection-less data link service requires a very complex transport protocol, because no acknowledge or retransmission is provided in the data link protocol. If a high quality data link service is provided however (connection-oriented data link service), the transport protocol can be very simple.

* The **network** layer shall mainly be responsible for the routing of messages (packets) between nodes on different networks (similar or dissimilar), connected to each other.

* The **data link** layer and the **physical** layer perform the transmission of packets on a data network. The type of data link and physical services provided to the network layer will be discussed later in this report.

Observing each layer, we see (as stated earlier) that a separation in the OSI model has to be between the transport layer and the session layer. This separates the model into a network part (layer 1-4) and a part using the network (layer 5-7). It is obvious that the MMTCP should take the responsibility for the upper three layers, since it is the user of the network. In most cases, these layers are implemented as a piece of software that is embedded in the operating system of the computer. It should however be possible to implement these layers within the (hardware) structure of the MMTCP.

The lowest four layers implement the communication system that is used by the MMTCP. The only layer seen by the MMTCP is the transport layer. Of course, this layer is only being seen by the implemented session layer of the MMTCP. The other layers of the communication system are invisible to the MMTCP, since they only provide a service to the transport layer.

In the next chapters, we are only engaged in the communication system that was mentioned above.

3. CONFIGURATION CONSIDERATIONS

First the structure of the communication controller in the entire system was investigated. Attention was paid to the following subjects:

- speed of the system
- overhead for the host-processor with handling communication-interrupts
- ease of integration
- price
- flexibility
- complexity

3.1 Architecture of the System

The communication system can be connected to the MMTCP in two ways (figure 3.1). Both systems have a MMTCP connected to the host bus. The connection differs in both cases; the communication controller is coupled to the local host bus in figure 3.1.a and to the MMTCP in figure 3.1.b.

Note that this communication system will be used primarily for inter-process communication. For Local Area Network (LAN) facilities normally a special network will be needed. Possibly some (low-priority) LAN facilities will be provided by the interprocess network. However, it will not be the starting-point to this work.

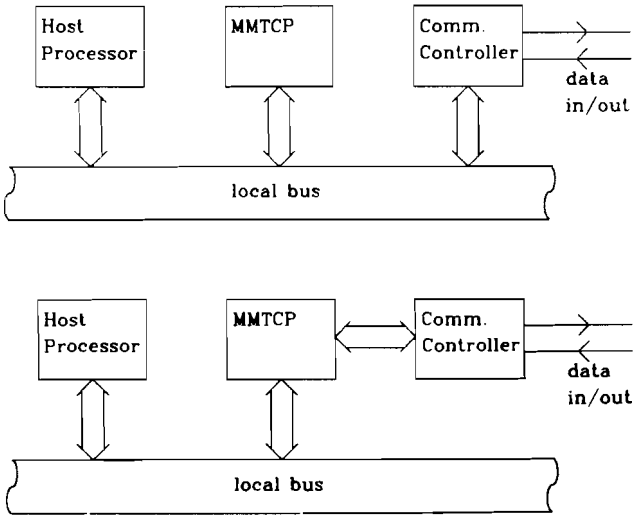


figure 3.1 : Two ways to link a communication controller to a system

Before we discuss the advantages and disadvantages of both versions, (which we will call system 1 and system 2 for simplicity) we will set our requirements on the communication system:

- it must be fast (at the network side and the host system side!)

- the complexity must not be too high
- the price of the communication controller must not be too high; the controller adds a value to the MMTCP but nothing more!
- it should be possible to use the controller in different systems (e.g. Motorola and Intel systems)

Let us test the two systems with the requirements listed above, starting with system 1 (figure 3.1.a).

The communication link controller consists of a (multi)protocol-controller with intelligence. This intelligence is a piece of hard- and/or software inside the controller. It has the advantage that the host processor does not have to manage the flow of data through a network. It suffices to send simple commands to the controller (e.g. SEND MESSAGE_XXX).

In this system the local bus is used to manage the controller and to transport data from the MMTCP to the controller and vice versa. The host system will use a part of the time for interprocess communication, which is a waste of the processor-time.

The realisation of an intelligent controller is rather easy: for each computer system many different 'communication adapters' are available. The software to operate these can be bought together with the adapters. Depending on the intelligence of the systems (adapters plus software) the price increases exponentially.

In system 2 (figure 3.1.b) the same(?) controller is directly coupled to the MMTCP. Advantageous is that no interaction with the host processor is needed. The time can be used more effective now. This system can be realised in (at least) two ways:

- couple a 'communication adapter' to the MMTCP
- integrate a communication link controller on the MMTCP-chip

The second option is attractive; The price for a multiprocessor system decreases, because no (expensive) adapter is needed. The complexity of the controller remains the same (not the design time!). If a certain network type is chosen, is it impossible to switch to an other network. This is not a disadvantage, because every MMTCP would have the same network.

comparing system 1 and system 2 with each other, we come to the following differences:

3.1.1 System Speed

The data rate on the communication link depends on the type of network that is implemented in the controller. The processes in system 1 will be slower than in system 2 because the host processor is involved in every transmission over the data link in system 1.

The co-processor can automatically redirect messages from local processes to other local processes, or to remote processes using the LAN. This redirection will be completely invisible to the communicating process, and will not give a higher load to the host processor either.

3.1.2 Complexity/Price

The complexity of both systems is comparable: both systems can be equipped with a complex communication adapter or (system 2) with an on-chip communication controller, which results in a more expensive IC. The price of a more complex MMTCP-chip will be less than the price of a MMTCP-chip and an adapter together. The price of a Texas Instruments TMS380 Token Ring Controller chipset for instance is about $450.- (without any software) to give an indication.

The hardware, however, will be much simpler when an on-chip communication controller would be used.

3.1.3 Multiprocessor Support

The MMTCP will be designed in a way, that the network does not have to be present. During initialisation the MMTCP is told whether a network is available or not. If not, the MMTCP shall support multitasking functions only.

3.1.4 System Dependance

In system 1 the system dependance is very high; for every computer system a different communication adapter will be needed. In the worst case the software has to be rewritten for a new system.

In system 2 this problem is not present because the network is only 'seen' by the MMTCP. Even for the smallest host system it is possible to use the LAN with very low software overhead. A single chip processor can talk directly to a mainframe!

3.2 Conclusions

From the previous section one may conclude that system 1 has a lot disadvantages; the host is involved in every network action of the MMTCP, the price is rather high and the system dependance is very high. System 2 has only one important disadvantage;

either
- the price is high when a communication adapter is used

or
- including an on-chip controller lengthens the development time.

We decided to go on with the development of a MMTCP with an on-chip communication controller (figure 3.2). If we want to have control over the entire chip design, we must have a strong separation between the MMTCP and the communication controller (like INTEL's RUPI for instance: a microcontroller with an on-chip communication controller).

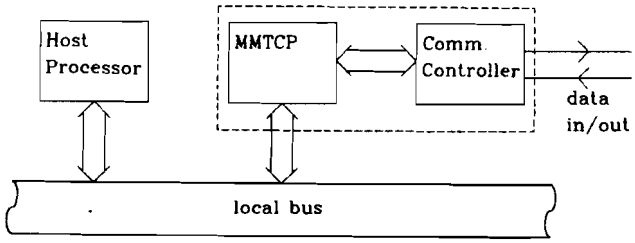


figure 3.2 : Structure of the MMTCP with the communication controller

3.3 ISO/OSI Protocol Layering within the MMTCP Architecture

Connecting the communication controller to the MMTCP, we have to assure that the design remains testable and that the communication between the MMTCP's session layer and the transport layer of the controller is optimal.

The realisation of the controller can be done in two ways, figure 3.3 gives two extremals. The first illustration gives an idea of the way we should realise the controller, when we think of an complete isolation of the lowest layers to the MMTCP. The internal structure of the MMTCP, however, makes it possible to exchange pointers to packets (stored in the working memory of the MMTCP) between two subsystems, using the MMTCP's messaging bus. The system proposed by the second illustration makes this very easy.

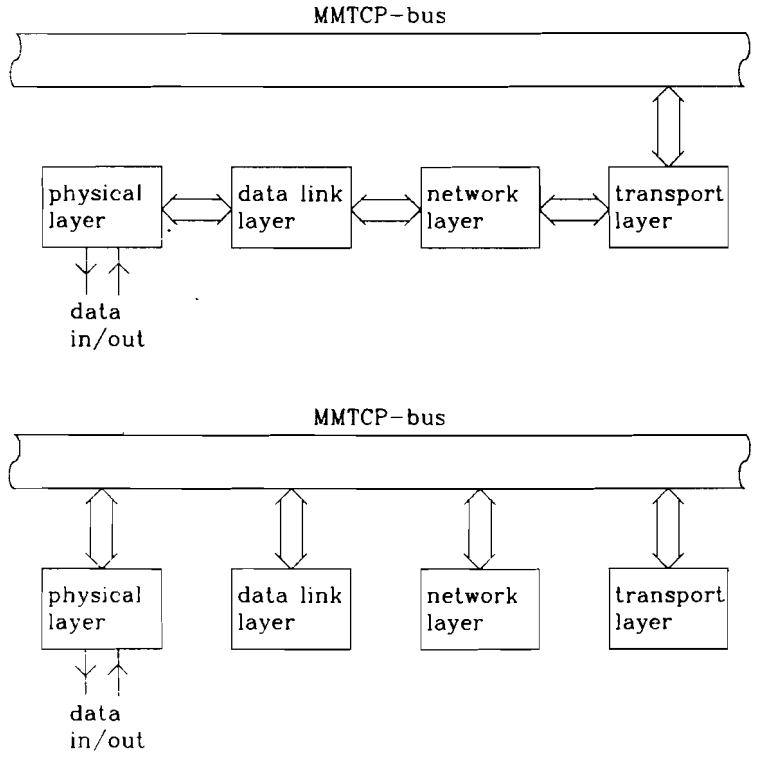


figure 3.3 : Possible configurations of the protocol layers

Let us investigate what happens when the MMTCP wants to transmit a message to an other MMTCP. It gives the transport layer a command to transmit a message, by sending this layer a pointer to the message. The transport layer appends an 'envelope' to the message and a new pointer is given to the network layer. This goes on until a pointer is given to the data link layer. This layer appends an envelope to the packet too, and gives a stream of symbols (e.g. bits or bytes) to the physical layer, who sends the message to the destination MMTCP through the network. Thus indicating that a direct connection between physical and data link layer is more efficient and therefore preferable. An efficient structure is given in figure 3.4.

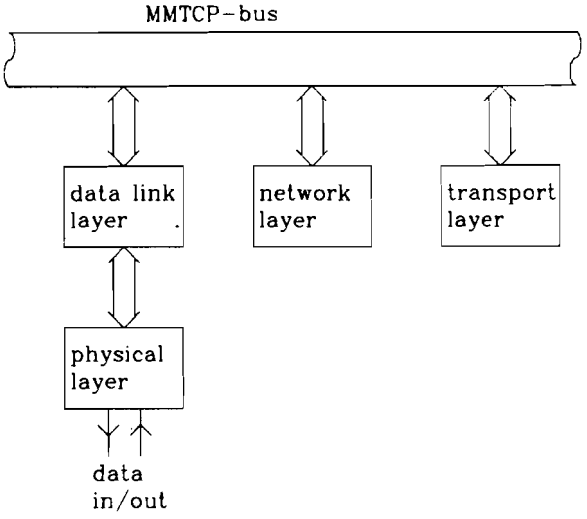


figure 3.4 : An efficient communication structure within the MMTCP

The same procedure is followed, when a stream of symbols is received. The physical layer will pass this stream to the data link layer, who puts the data in the MMTCP's local memory and passes a pointer to this memory part to the network layer. This communication structure has some advantages, mentioned below:

- it is possible to test all layers separately, using the bus of the MMTCP. This saves a lot of additional hardware.

- all layers can use the local memory of the MMTCP to store their data. This can be data that is received or has to be transmitted, but also a large address comparison table, for instance.

- all layers can use some features provided by the MMTCP's functional blocks; e.g. the memory (de)allocation block and the timer block.

Since the layers do not have to communicate with the building blocks, it is no disadvantage that every layer directly is connected to the MMTCP. We arrange that only the transport layer block can communicate with a block of the MMTCP, that implements the session layer. One exception shall be made however, every layer of the communication system may need some network management functions. These functions are controlled inside the layers or by the user (a program or a network operator). This means that there has to be some kind of interface between every layer and the host processor, which should be one or more special building blocks within the MMTCP's structure.

4. SELECTION OF THE LOCAL NETWORK TYPE

4.1 Comparison of Available Network Types.

As the title of this section already indicates, we only want to use a standard network type. The reason for this is very simple; we do not want to reinvent the wheel again. It is, of course, possible to define a completely new network standard, customised for the Interprocess Communication System. The obtained results probably would not justify the effort that was put into that project. Besides, there are enough network standards available to select a suitable one for our project.

In general, LAN standards are less complex than the standards specially developed for large (public) networks. For this reason we will concentrate us on three well documented LAN standards. These are standardised by the American National Standards Institute (ANSI) after being developed by the Standards Board of the Institute of Electrical and Electronics Engineers (IEEE). The LAN standards belong to the so called 802-family for Local Area Networks. The relationship between the members of the family is shown below ([4]).

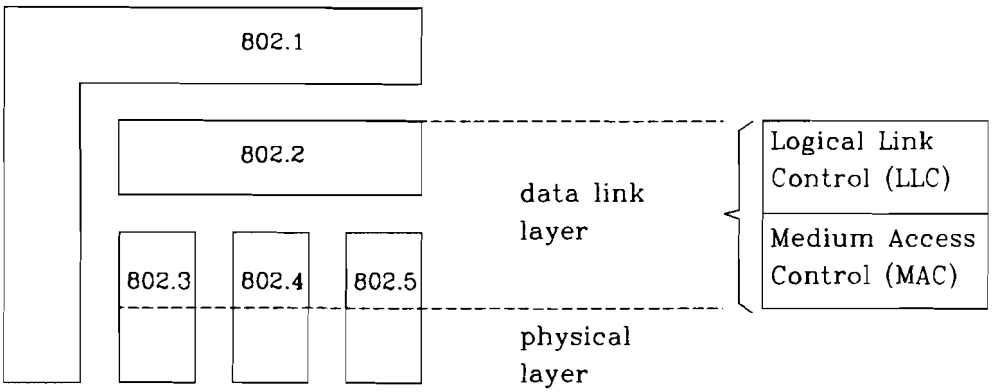


figure 4.1 : The family of LAN standards

This family of standards deals with the physical and data link layers as defined by the ISO/OSI Reference Model. The access standards define three types of medium access technologies and associated physical media, each appropriate for particular applications. The standards defining these technologies are:

1. ANSI/IEEE Standard 802.3-1985, a bus utilising CSMA/CD as the access method.

2. ANSI/IEEE Standard 802.4-1985, a bus utilising token passing as the access method.

3. ANSI/IEEE Standard 802.5-1985, a ring utilising token passing as the access method.

ANSI/IEEE Standard 802.2-1985, Logical Link Control (LLC) protocol, is used in conjunction with the medium access standards.

A companion document, IEEE 802.1 which still is in preparation, describes the relation among these standards and their relationship to the ISO/OSI Reference Model in more detail. This companion document will contain internetworking (a part of the network layer) and network management issues.

We will now give a brief description of the three Medium Access Control (MAC) mechanisms mentioned above:

1. The **CSMA/CD** MAC standard (802.3), which means Carrier Sense Multiple Access with Collision Detection. Each station listens to the medium. When it hears a silence on it, permission is given to transmit a message on the medium. Because this method does not guarantee that the station is the only one that is transmitting on the ring, the stations have to test during transmission if another station is transmitting too (collision detection). When a collision is detected, both stations withdraw their transmission from the bus and wait for a random time before they start to listen again to the medium. A priority mechanism is provided by letting the stations wait a certain time after the silence on the medium, before they gain the right to transmit on it; a high priority message has to wait a shorter time before a transmission than a low priority message.

2. The **Token Bus** MAC standard (802.4) has the 'same' bus topology as the standard mentioned above. The control mechanism differs significantly. All stations are directly coupled to each other via a bus (a coaxial cable or an optical fiber for instance). The stations are logical coupled towards each other as if they were in a ring configuration (coming next). A token, which is some message format, is transmitted on the logical ring from one station to the other. In fact, the message is continuously transmitted on the same medium. When a station wants to transmit a message, it changes the token format into a format that identifies the head of a frame and starts transmission to the addressed station. Note, that the transmission of this frame is not done over the logical ring, but directly to the addressed station. A priority mechanism is provided here too; a part of the token is used to identify the priority of the token and to identify the reservation field of the token. A token may only be used to begin the start of a transmission if the priority of the queued message is equal to or greater than the priority of the token.

3. The **Token Ring** MAC standard (802.5) makes use of a ring network topology. The logical ring connection of the Token Bus standard is replaced by a real ring connection of the stations towards each other. Whenever a station wants to transmit a message, it simply waits for the token. If the token can be used for transmission, it changes the token in a start of frame sequence and starts transmission on the ring. A token can be used if the requested priority of the enqueued frame is equal to or greater than the token priority. Each station on the ring will repeat the frame that is transmitted on the ring, adding an (at

least) one-bit delay to the message. If the token cannot be used for transmission however, the reservation bits in the token will be adjusted to the desired priority of the enqueued message. The addressed station copies the message from the ring and sets a few bits in the trailing sequence of the passing frame (the 'address recognised' and the 'frame copied' bits) to one. So, the priority mechanism is the same as the one used in the Token Bus system.

Together with the MAC layer, the LLC (802.2) layer implements the data link control layer of the ISO/OSI Reference Model. This standard describes the use of connection-oriented and connection-less services to the network layer.

A connection-less service is useful, when a high quality medium and physical access service is provided (which is true for a normal LAN; the BER usually does not exceed 10^{-8}). This service can be given without the establishment of a connection between stations, so the data rate on such a channel can be improved a little because less packets need to be send between stations. A draw-back of the connection-less service is, that a more complex transport protocol shall be needed to guarantee an error-free network service to the session layer of the MMTCP.

A connection-oriented service can be provided by the LLC layer. This service provides an error-free connection between two stations. In this case the transport layer can be much simpler.

The architecture of the system using the LAN standards of the ANSI/IEEE standards will be :

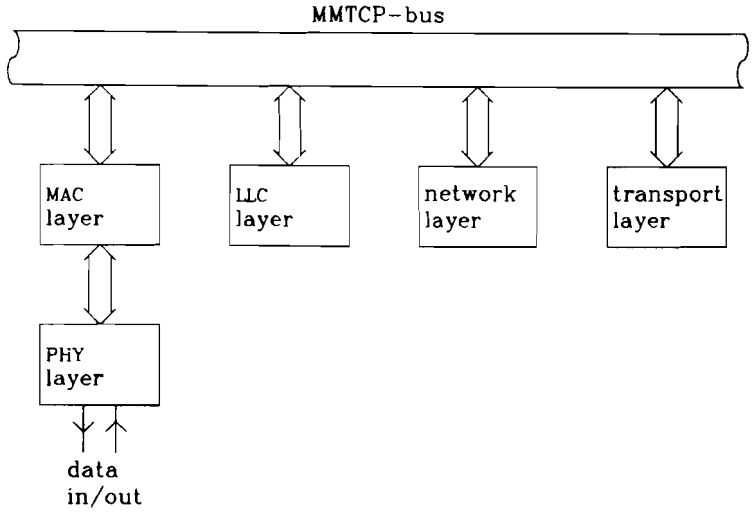


figure 4.2 : Architecture of the communication system within the MMTCP

In the next section we will not decide which type of data link service we are going to provide to the network layer. We are only going to decide on the type of medium access control we are implementing on the MMTCP-chip. The LLC and higher layers remain under study.

4.2 Selection of the Most Suitable Network for the MMTCP

For the selection of a suitable network, we pose some requirements on it. The most important are:

- it must be an existing standard.
- it must be possible to use a priority control scheme on the network.
- it must be possible to implement the network controller on a chip.
- high system capacity.
- high data rate.
- minimal system access time.

The three medium access control mechanisms mentioned in the previous section will be compared in the next subsections.

4.2.1 Priority Control Scheme

Between processes different kinds of messages are exchanged. Some of these messages are more important than the others and need a priority control scheme. In the previous section we already indicated how such a scheme is used (all three MAC mechanisms possess one).

The CSMA/CD protocol forces a station that wants to transmit a low priority message to wait a relatively long time after a silence is detected. A high priority message is transmitted immediately after a silence is detected. Eight priority levels are provided in this system.

The token access protocols use a field in the token to indicate which service level can be provided to a station. The priority at which a message can be transmitted is always lower than or equal to the requested priority, thus preventing a low priority frame to occupy the network instead of the high priority message. In these protocols eight priority levels are provided to the system too.

The scheme in which a priority is assigned to a message is under study. The MMTCP shall be responsible for this assignment (for the greatest part; some network management messages on the system shall be transmitted at a priority level, which is predefined by the network standards).

We can state that all three systems meet the requirements 1 and 2.

4.2.2 Hardware Implementation Considerations

The CSMA/CD and Token Ring protocols have already been implemented on a chip (set) by Intel and Texas Instruments, respectively. Comparing the three access methods, we observe that the complexity of the Token Ring is somewhat higher than the complexity of the CSMA/CD system. The Token Bus method is much more complex than the Token Ring access method, which might be the reason that it is never implemented in hardware. (In the future this method will probably be implemented, since General Motors Manufacturing Automation Protocol is based upon the Token Bus Medium Access Control standard).

The three systems need some subsystems outside the chip. These subsystems are:

- a line driver to put the electrical signals on the medium. This prevents the (expensive) MMTCP-chip from damage caused by spikes and consuming too much power (increasing the life-time of the chip).

- a receiver filter to shape the (distorted) input signals.

- a clock recovery network to extract the data clock from the received signal.

These subsystems can be integrated on an analogue chip or can be built with discrete components.

For our Interprocess Communication system, the CSMA/CD and Token Ring access method are probably best fit. Regarding the complexity, the Token Bus access method is less preferable.

4.2.3 System Capacity

With system capacity is meant here: the ability to process much and large data blocks in the medium access control and physical layer. The system capacity depends only on the use of MMTCP-buses and can be as high as needed for all three systems.

Remark that a bit rate that is too high for the MMTCP, may result in a frame handling error (this effect is called a 'receiver overrun'). In spite of the fact that the MAC layer has a top priority access to the memory, it sometimes has to wait before access to the memory bus is granted. This effect can be minimised using a receive FIFO buffer, thus using more hardware on-chip.

Another problem may be the processing capacity of the higher layers. These might be too slow to process the frames, delivered by the lower layer(s). An efficient implementation of the LLC Layer, Network Layer and Transport Layer probably shall eliminate this problem.

4.2.4 Data Rate

A performance comparison between the three mentioned systems is given in [5]. This paper presents the results of the calculation of the maximum mean data rate for LAN's. The next illustrations give an idea of some relevant results from this report. They show what the actual data rate on a system can be under a given system configuration (see figure 4.3).

Assume that, in a systems with many stations, only one station is active. The actual data rate of the CSMA/CD system then is maximal, because no messages can collide[1]. When more stations become active, the probability of a collision to occur increases. The actual data rate of the system thus decreases. When the number of active stations becomes high, network operation becomes impossible due to the many collisions. We can conclude

---

[1]For proper operation, CSMA/CD requires a minimum framelength. This minimum depends on the implementation. In most cases, the minimum number of bytes in the INFO field is 46. If the supplied number of bytes is less than 46, an extra field called PAD is added to the INFO field. The Token Bus and Token Ring standards do not require a minimum INFO field length.

that the CSMA/CD system performs well as long as the network load is not too high.

The Token Bus system, on the other hand, performs well if the load is high. If only one station is active, the actual data rate is much lower than the transmission rate. The reason for this is the long circulation time of a token on the (logical) ring. Since a token has to be transmitted from station to station, every transmission occupies the bus completely. A system with many stations on it, from which only a few are active, will need a lot of time to let the token circulate on the ring (e.g. 100 stations, each transmitting a token of 96 bits long!). When all stations are active, most of the tokens are used for the transmission of a frame. We see, that the efficiency of the system increases, when the number of active stations increases.

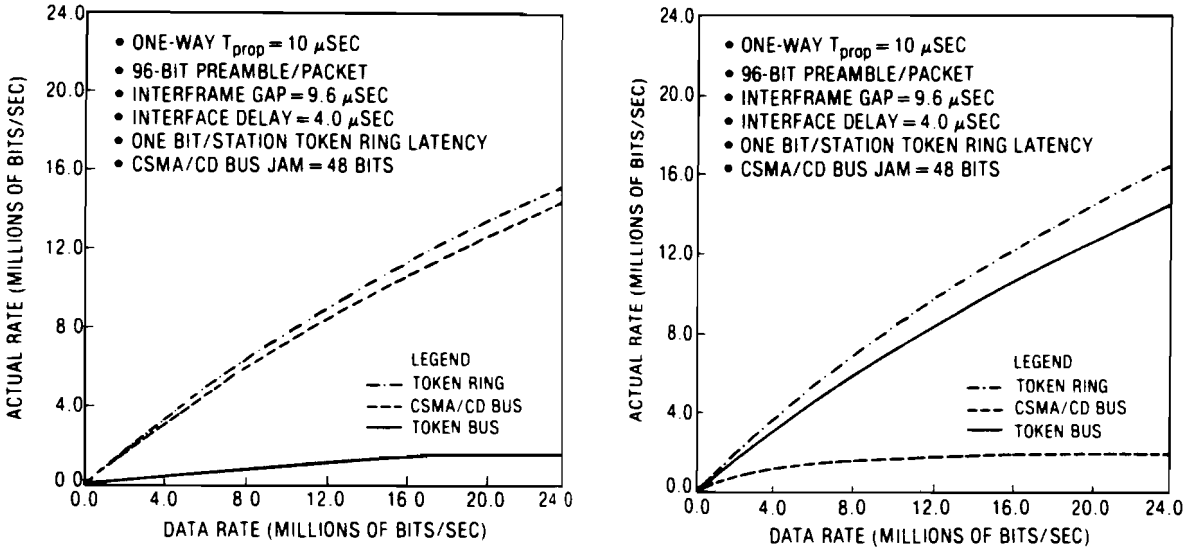


figure 4.3 : Maximum mean carried data rate versus actual transmission rate when one station is active out of 100 and when all 100 stations are active, respectively: 500 bits per packet.

The Token Ring system is the most efficient system of the three mentioned. Although a token has to circulate on the system, the total delay is only one bit per station instead of 96 bits per station in the Token Bus system. Therefore the efficiency with only one station is high, roughly as high as the efficiency of the CSMA/CD system. When the load of one station increases and/or the number of active stations increases, the actual data rate does not decrease; the system remains very efficient.

4.2.5 Access Time

The access time of a system will now be defined. The access time is the average time that is required to get permission to transmit on the system after a message is enqueued in the medium access control system.

The access time of a station is low in a CSMA/CD system, when not too much stations are active. As soon as a silence is detected, a station can transmit a message. The Token Ring system has a somewhat longer access time, because a station has to wait for the next free token. The access time of the Token Bus system is rather long; the token has a relatively long circulating time on the logical ring.

4.3 Conclusions

From the six requirements stated in the previous section, only three can be used to select the network. The other three requirements are automatically accomplished:

- a standard system will be used
- a priority scheme is already provided in the standards
- the system capacity is a designers business

The selection will be done on the following subjects:

- the possibility to implement the controller on a chip
- the data rate must be high
- the access time must be low

We have already seen that the Token Bus system is more complex than the other two systems (Token Ring and CSMA/CD) and thus is less preferable. The CSMA/CD system has not a very good performance, when the number of active stations becomes high. The Token Bus system when the number of active stations becomes low. This makes the Token Ring system the most preferable system of the three mentioned, since we require a system that has an optimal performance both when the number of active stations is low and when the number of active stations is high. The access time of the Token Ring system is somewhat higher than the access time of the CSMA/CD system, but this is acceptable. The actual transmission rate is standardised too. However, the importance of it for our project is minor. The actual transmission rate in the MMTCP network shall depend upon the integration technology and not on the standard. We require at least the standard transmission rates, which are:

CSMA/CD:	10 Mbit/second
Token Bus:	10 Mbit/second
Token Ring:	4 Mbit/second (16 Mbit/s under study)

We can conclude that the Token Ring access mechanism is the best suitable network type for the MMTCP. This method will be the subject of study during the rest of this report.

5. SPECIFICATION OF THE TOKEN RING CONTROLLER

A comprehensive description of the Token Ring access method can be found in [4.5] and [6]. The ANSI/IEEE Standard 802.5 [4.5], gives a detailed description of the system. [6] was one of the first papers contributed to the IEEE project. It gives a clear insight into the Token Ring system.

This chapter will give a specification of the MAC and PHY layer of the communication controller. Basis for this specification will be the standard 802.5. Also some ideas will be adopted from [7], which is the user's guide to the (TMS380) Token Ring chipset developed by Texas Instruments.

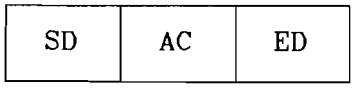
5.1 Specification of the Frame Format

There are two basic formats used in Token Rings: tokens and frames. In the following discussion, the figures depict the formats of the fields in the sequence as they are transmitted on the medium, with the left-most bit or symbol transmitted first.

Processes, which require comparison of fields or bits, perform that comparison upon those fields or bits as depicted, with the left-most bit or symbol compared first, and for the purpose of comparison, considered most significant.

5.1.1 Token Format

The token shall be the means by which the right to transmit (as opposed to the normal process of repeating) is passed from one station to another.

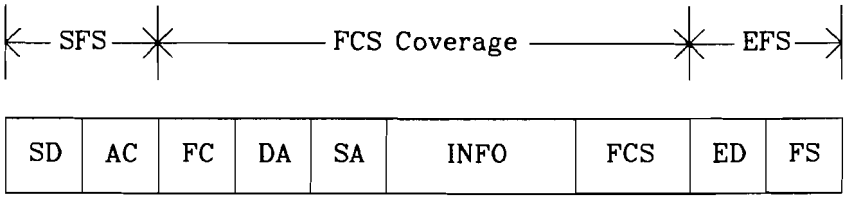


SD = Starting Delimiter (1 byte)
AC = Access Control (1 byte)
ED = Ending Delimiter (1 byte)

figure 5.1 : The token format

5.1.2 Frame Format

The frame format shall be used for transmitting both MAC and LLC messages to the destination station(s). It may or may not have an information (INFO) field.



SFS = Start-of-Frame Sequence
SD = Starting Delimiter (1 byte)
AC = Access Control (1 byte)
FC = Frame Control (1 byte)
DA = Destination Address (2 or 6 bytes)
SA = Source Address (2 or 6 bytes)
INFO = Information (0 or more bytes)
FCS = Frame-Check-Sequence (4 bytes)
EFS = End-of-Frame Sequence
ED = Ending Delimiter (1 byte)
FS = Frame Status (1 byte)

figure 5.2 : The frame format

5.1.3 Abort Sequence

This sequence shall be used for the purpose of terminating the transmission of a frame prematurely. The abort sequence may occur anywhere in the bit stream; that is, receiving stations shall be able to detect an abort sequence even if it does not occur on byte boundaries.

figure 5.3 : The abort sequence

5.1.4 Fill

When a station is transmitting (as opposed to repeating), it shall transmit fill preceding or following frames, tokens, or abort sequences to avoid what would otherwise be an inactive or indeterminate transmitter state.

Fill may be either 0 or 1 bits or any combination thereof and may be any number of bits in length, within the constraints of the Token Holding Timer (THT).

5.1.5 Starting Delimiter (SD)

A frame or a token shall be started with these eight symbols[2]. If otherwise, it shall not be considered valid.

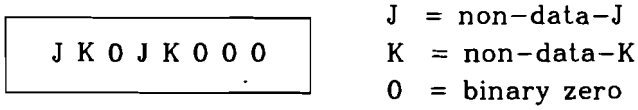


figure 5.4 : The starting delimiter

5.1.6 Access Control (AC)

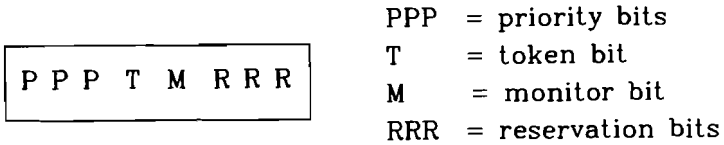


figure 5.5 : The access control sequence

5.1.6.1 priority bits

The priority bits shall indicate the priority of a token, and which stations are allowed to use the token. In a multiple-priority system, stations use different priorities depending on the priority of the Protocol Data Unit (PDU) to be transmitted.

The eight levels of priority increase from the lowest (000) to the highest (111) priority. For purposes of comparing priority values, the priority shall be transmitted most significant bit first; for example, 110 has higher priority than 011 (left-most bit transmitted first).

5.1.6.2 token bit

The token bit is a 0 in a token and a 1 in a frame. When a station with a PDU to transmit detects a token which has a priority equal to or less than the PDU to be transmitted, it may change the token to a start-of-frame sequence and transmit the PDU.

5.1.6.3 monitor bit

The monitor bit is used to prevent a token whose priority is greater than 0 or any frame from continuously circulating on the ring. If an active monitor detects a frame or a high priority token with the monitor bit equal to 1, the frame or token is aborted.

---

[2]For a discussion of non-data symbols, see appendix B.3.

This bit shall be transmitted as 0 in all frames and tokens. The active monitor inspects and modifies this bit. All other stations shall repeat this bit as received.

5.1.6.4 reservation bits

The reservation bits allow stations with high priority PDU's to request (in frames or tokens as they are repeated) that the next token is issued at the requested priority. The precise protocol for setting these bits is described in appendix B.1.

The eight levels of reservation increase from 000 to 111. For purposes of comparing reservation values, the reservation shall be transmitted most significant bit first.

5.1.7 Frame Control (FC)

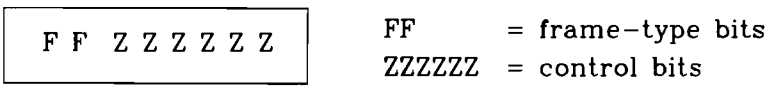


figure 5.6 : The frame control sequence

The FC field defines the type of the frame and certain MAC and information frame functions.

5.1.7.1 frame-type bits

The frame-type bits shall indicate the type of the frame as follows:

00 = MAC frame (contains an MAC PDU)
01 = LLC frame (contains an LLC PDU)
1x = undefined format (reserved for future use)

Medium Access Control Frames

If the frame-type bits indicate a MAC frame, all stations on the ring shall interpret and, based on the finite state of the station, act on the ZZZZZZ control bits.

Logical Link Control Frames

If the frame-type bits indicate an LLC frame, the ZZZZZZ bits are designated as rrrYYY. The rrr bits are reserved and shall be transmitted as 0's in all transmitted frames and ignored upon reception. The YYY bits shall be used to carry the priority (Pm) of the PDU from the source LLC entity to

the target LLC entity or entities. Note that P (the priority in the AC field of a frame) is less than or equal to Pm when the frame is transmitted onto the ring.

Undefined Format

The value 1x is reserved for frame types that may be defined in the future. Frames in this format shall be ignored upon reception.

5.1.8 Destination and Source Address (DA and SA)

Each frame shall contain two address fields: the destination (station) address and the source (station) address, in that order. According to the 802.5 standard addresses may be either 2 or 6 bytes in length. However, all stations of a specific LAN shall have addresses of equal length.

Since the average data packet length probably will be less than 64 bytes (depending on the application of the communication controller; a high speed system is assumed), the average frame length probably will be (less than) 77 or 85 bytes. When a address length of 2 bytes is chosen, the possible data rate on the network increases by more than 10%. The maximum number of stations in the network then will be 32,767, which should be sufficient for an interprocess network. However, if a large network is used, the address resolution of a 2 byte station address might become too low. Then a 6 byte address resolution is desired. Both options shall be available (this option is software switchable when the MMTCP is initialised).

5.1.8.1 the destination address

The destination address (DA) identifies the station(s) for which the information field of the frame is intended. Included in the destination address is a bit to indicate whether the destination address is an individual or group address. The format of the destination address is shown below.

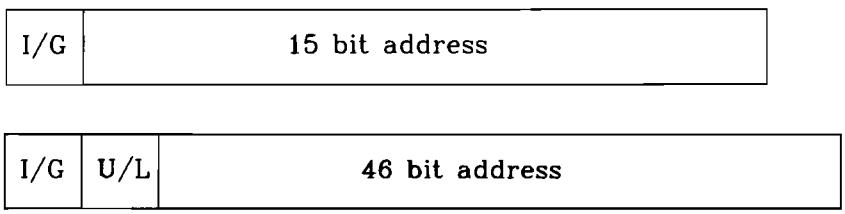


figure 5.7 : The destination address format (2 and 6 byte version)

The first bit transmitted of the destination address distinguishes individual from group addresses:

0 = individual address
1 = group address

Individual addresses identify a particular station on the LAN and shall be distinct from all other individual stations on the same LAN. Individual station addresses are administered by a local (to the LAN) authority.

A group address shall be used to address a frame to multiple destination stations. Group addresses may be associated with zero or more stations on a given LAN. A group address is an address associated by convention with a group of logically related stations.

There are two methods of administrating the set of 6 byte addresses: locally or through a universal authority (e.g. a PTT). The second bit of the destination address indicates whether the address has been assigned by a universal or local administrator:

- 0 = universally administered
- 1 = locally administered

The following structure provides for a Token Ring LAN divided into multiple rings, with one or more (MAC-level) relay stations interconnecting the rings. This structure is only allowed for locally administered addresses.

A ring is defined as the collection of all stations of a LAN that have the same ring number and that can exchange frames without any intermediary MAC-level relay entity. Stations on a ring can communicate with stations with different ring numbers only through a MAC-level relay or some other intermediary.

A hierarchical address permits a relay station to recognise frames that require forwarding to other rings by applying a straightforward algorithm to the frames to be forwarded.

The destination address partitioning recommended for this purpose is:

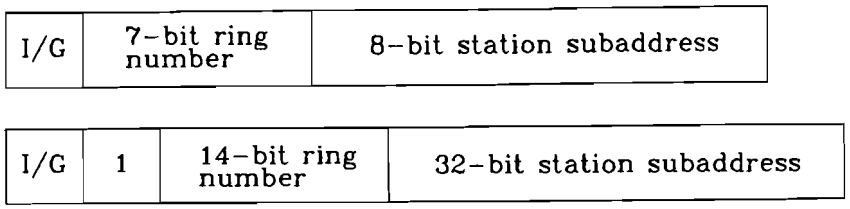


figure 5.8 : Hierarchical form destination address

This partitioning makes it possible to interconnect 126 rings, with each ring having a maximum of 254 stations with a 2 byte address or 16k rings, with each ring having a maximum of 4G stations with a 6 byte address. The all zeros and all ones fields shall not be used in an 'normal' address for neither a ring number nor a subaddress. The following addressing conventions are recommended (see table 5.1) :

Broadcast:

all bits are set to ones.

Null address:

all bits set to zero in the destination address shall be considered a null address. It will mean the frame is not addressed to any particular station. The use of this address type is very doubtful.

This ring:

the ring number field is set to all zeros or to the ring number of this ring, if known.

All stations, this ring:

the ring number field is set to all zeros or to the ring number of this ring, if known; the station subaddress field is set to all ones.

All rings:

all bits of the ring number set to all ones (it does not imply that the frame is destined for all stations on all rings).

table 5.1 : addressing conventions (only given here for the 2 byte address, the 6 byte address is obviously almost the same).

address type	I/G-bit	ring number	station subaddress
broadcast	1	1111111	11111111
null address	0	0000000	00000000
this ring	x	0000000	xxxxxxxx
	x	this ring's nr	xxxxxxxx
all stations	x	0000000	11111111
on this ring	x	this ring's nr	11111111
all rings	x	1111111	xxxxxxxx

x = don't care, depends on the application

Two formats for group addressing are defined within the structure of hierarchical addressing (as described above), using the first bit of the station subaddress field:

0 = bit-significant (functional) mode
1 = conventional group mode

The bit-significant mode specifies that each bit in the station subaddress field represents a single group address. For 16-bit addresses, 7 bit-significant addresses may be defined in this mode. Stations that are to copy frames destined for many different functions may implement a bit- significant mask

to facilitate the copying of frames with bit- significant destination addresses. Such a mask would have a bit set for each bit-significant address for which the station wishes to copy frames. Four bit-significant addresses will be defined now (see table 5.2).

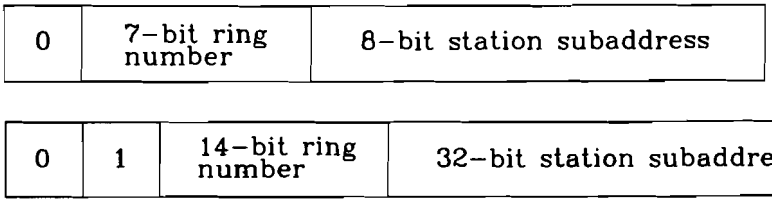
table 5.2 : defined bit-significant (functional) addresses

function	address (last 7 bits)
active monitor	0000001
ring parameter server	0000010
reserved	0000100
ring error monitor	0001000
network manager	0010000
reserved	0100000
reserved	1000000

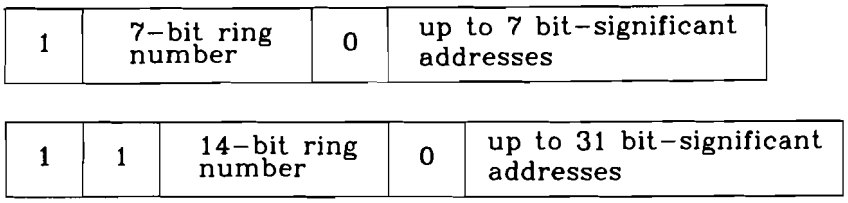
When station 'Y' has a bit-significant mask 0011000, for example, it can copy frames for the ring error monitor and the network manager.

The conventional group mode specifies the that the remaining bits in the station subaddress field represent a single group address. For 16-bit addresses, this allows 127 group addresses in conventional group mode (2G group addresses with 48-bit addresses). In this mode the group number '0000000' will be accepted, because this shall not be recognised as a broadcast address. The three addressing modes are illustrated in figure 5.9.

1. **16-bit hierarchical form, individual address**



2. **16-bit hierarchical form, bit significant mode**



3. **16-bit hierarchical form, conventional group mode**

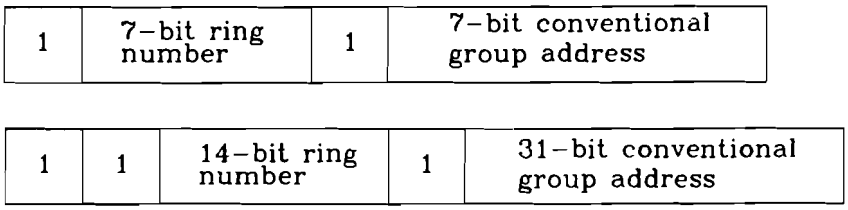


figure 5.9 : The three possible addressing modes

5.1.8.2 the source address

The source address (SA) shall identify the station originating the frame and shall have the same format as the destination address in a given frame. The individual/group bit shall be zero (see illustration below) and the universal/local bit shall be one (we assume that the addresses always will be administered locally).

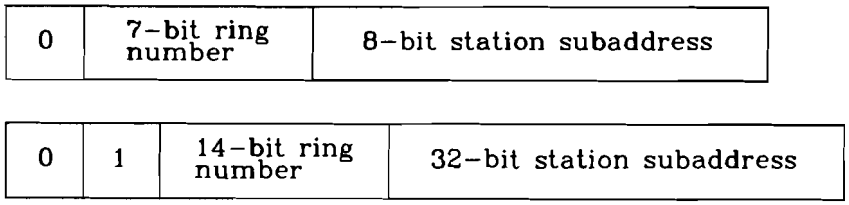


figure 5.10 : Source address format

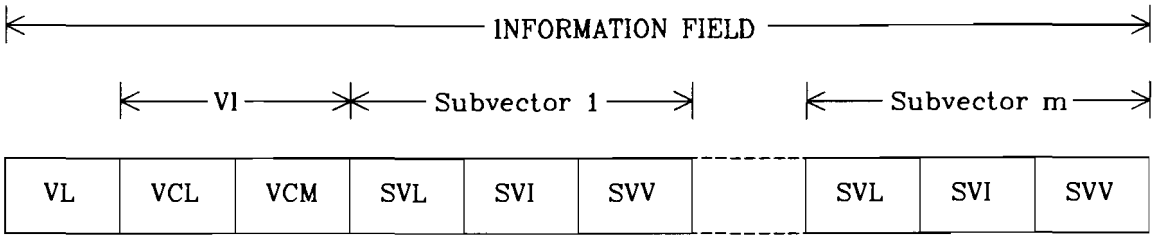
5.1.9 Information (INFO) Field

The information field contains 0, 1, or more bytes that are intended for MAC, NMT, or LLC. Although there is no maximum length specified for the information field, the time required to transmit a frame may be no greater than the token holding period that has been established for the station.

The format of the information field is indicated in the frame-type bits of the FC field. The frame types defined are the MAC frame and the LLC frame.

5.1.9.1 MAC frame format

Figure 5.11 defines the format of the information field, when present, for MAC frames.



VL (2 bytes)	=	Vector Length
VI (2 bytes)	=	Vector Identifier
VCL (1 byte)	=	Vector-class identifier
VCM (1 byte)	=	Vector-command identifier
SVL (1 or 3 bytes)	=	Subvector Length
SVI (1 byte)	=	Subvector Identifier
SVV (n bytes)	=	Subvector Value

figure 5.11 : The MAC frame information field structure

Vector

A vector is the fundamental unit of the MAC and NMT information. A vector contains its length, an identifier of its function, and zero or more subvectors. Only one vector is permitted per MAC frame.

Vector Length (VL)

The VL is a 16-bit binary number that gives the length, in bytes, of the vector. The length includes the VL field and, depending on the length of the Token Holding Timer, can have values such that:

0004H <= VL <= FFFFH.

Vector Identifier (VI)

The VI is a 2- or 4-byte code point that identifies the vector. The first byte, the vector-class identifier (VCL) defines the origin and destination class of the vector. The high-order four bits are the destination class (DC) and the low order four bits are the source class (SC). The defined class types are as follows:

0H: Ring Station
4H: Network Manager
5H: Ring Parameter Server
6H: Ring Error Monitor

The second byte, the vector command (VCM), defines the function the receiver has to perform. The value FFH indicates that an expanded identifier is being used and the command is contained in the next two bytes. In most cases an expanded identifier is not used. The communication controller does not use them at all. Therefore frames carrying an expanded identifier shall be ignored.

Subvector (SV)

Vectors require all data or modifiers to be contained within subvectors. One subvector is required to contain each piece of data or modifier that is being transported. A subvector is not position-dependant within a vector, but rather, each subvector must be identified by its subvector identifier.

Subvector Length (SVL)

The SVL is an 8-bit binary number that gives the length, in bytes, of the subvector. The length includes the length of the SVL field. A subvector length of FFH means that the subvector is longer than 254 bytes and the actual length is included in the next two bytes.

Subvector Identifier (SVI)

The SVI is a 1-byte code point that identifies the subvector. The code point of FFH indicates that an expanded identifier is being used and is contained in the next two bytes

The subvectors are of two types. The subvectors with code points from 00H through 7FH are used so that certain specific, common (to many vectors) strings of MAC and NMT data can be formatted and labeled in a standard

manner. This standardisation is intended to facilitate sharing of data between MAC and NMT applications and make the data as application-independent as possible.

The subvectors with code points from 80H through FEH are for specific definition within a particular vector by vector identifier. For example, the subvector 90H can have an entirely different definition in every different vector. The subvector 40H has only one definition across all vectors and applications.

Subvectors themselves may contain other subvectors and other types of vectors and optional fields which are unique only to the particular subvector to which they belong.

5.1.9.2 LLC frame format

The format of the information field for LLC frames is not specified here. However, all stations at least must be capable to receive frames whose information field is up to and including 133 bytes in length.

5.1.9.3 order of bit transmission

Each byte of the information field shall be transmitted most significant bit first.

5.1.10 Frame-Check Sequence (FCS)

The FCS shall be a 32-bit sequence based on the following standard generator polynomial of degree 32.

$$G(X) = X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$$

The FCS shall be the 1's complement of the sum (modulo 2) of the following:

(1) The remainder of $X^k(X^{31} + X^{30} + X^{29} + \ldots + X^2 + X^1 + 1)$ divided (modulo 2) by G(X), where k is the number of bits in the FC, DA, SA, and INFO fields;

(2) The remainder after multiplication by X^{32} and then division (modulo 2) by G(X) of the content (treated as a polynomial) of the FC, DA, SA, and INFO fields.

The FCS shall be transmitted commencing with the coefficient of the highest term.

As a typical implementation, at the transmitter, the initial remainder of the division is preset to all 1's and then modified by division of the FC, DA, SA, and INFO fields by the generator polynomial, G(X). The 1's complement of this remainder is transmitted with the most significant bit first as the FCS.

At the receiver, the initial remainder is preset to all 1's and the serial incoming bits of FC, DA, SA, INFO, and FCS will be divided by G(X). In the

absence of transmission errors, a unique non-zero remainder value will result. This unique remainder value is the polynomial:

$X^{31} + X^{30} + X^{26} + X^{25} + X^{24} + X^{18} + X^{15} + X^{14} + X^{12} + X^{11} + X^{10} + X^8 + X^6 + X^5 + X^4 + X^3 + X + 1$

5.1.11 Ending Delimiter (ED)

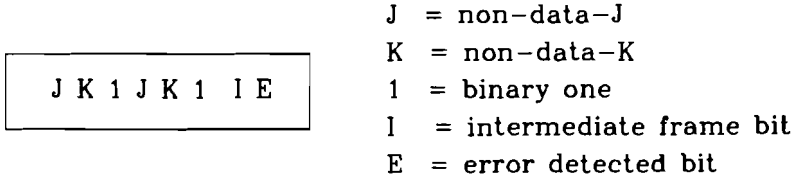


figure 5.12 : The ending delimiter sequence

The transmitting station shall transmit the delimiter as shown. Receiving stations shall consider the ending delimiter valid if the first six symbols JK1JK1 are received correctly.

5.1.11.1 intermediate frame bit (I bit)

To indicate that this is an intermediate (or first) frame of a multiple frame transmission, the I bit shall be transmitted as 1. An I bit of 0 indicates the last or only frame of the transmission.

5.1.11.2 error-detected bit (E bit)

The E bit shall be transmitted as 0 by the station that originates the token, abort sequence, or frame. All stations on the ring check tokens and frames for errors (for example, FCS error, non-data symbols). The E bit of tokens and frames that are repeated shall be set to 1 when a frame with error is detected; otherwise the E bit is repeated as received.

5.1.12 Frame Status (FS)

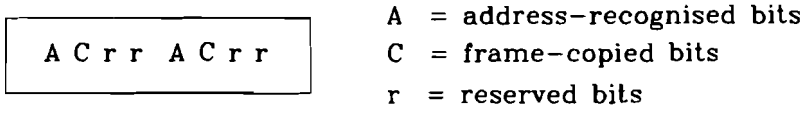


figure 5.13 : The frame status sequence

The r-bits are reserved for future standardisation. They shall be transmitted as 0's; however, their value shall be ignored by the receivers.

5.1.12.1 address-recognised (A) bits and frame-copied (C) bits

The A and C bits shall be transmitted as 0 by the station originating the frame. If another station recognises the destination address as its own address or relevant group address, it shall set the A bits to 1. If it copies the frame (into its receive buffer), it shall also set the C bits to 1. This allows the originating station to differentiate among three conditions:

(1) Station non-existent/non-active on this ring
(2) Station exists but frame not copied
(3) Frame copied

The A and C bits shall be set without regard to the value of the E bit and only if the frame is good as defined in appendix B.1. Only the values that are 00rr00rr, 10rr10rr, and 11rr11rr shall be considered valid. All other values are invalid and ignored by the receiver.

5.2 Specification of the MAC Frames

The following are descriptions of various MAC frames that are used in the management of the Token Ring. Values for PDU priority (Pm), FC, DA, and INFO field content (VI, SVI, and SVV) associated with the particular MAC Supervisory Frame, are indicated. All frames are essential for proper operation of the Standby and Active Monitors. Frames with the following FC values are to be handled as listed:

- If the value of the FC of the frame is 00H and it is addressed to the station, it will be copied only if there is sufficient free buffer available for copying.

- If the value of the FC of the frame is 01H and it is addressed to the station, every effort will be made to copy the frame including over-writing previously received information.

- If the value of the FC of the (MAC) frame is greater than 01H (and of course smaller than 80H), it will be addressed to all stations on the ring. It will be copied only if there is sufficient free buffer available for copying. If the frame is not copied, action will be based on the value of the FC field.

Values, other than those defined below, shall not be processed within the MAC layer. These shall be passed to the Network Management Layer, who will take the correct receiving actions. The general format of the information field of MAC frames is described under 5.1.9.1.

5.2.1 Claim Token MAC frame (CL_TK)

When a station that is in standby state determines that there is no active monitor operating on the ring, it shall enter a claiming token state. While in this state the station shall send claim token frames and inspect the source address of the claim token MAC frames it receives. If the SA matches its own (MA) address and subvector 1 matches the stored upstream neighbour's address (SUA), it has claimed the token and shall enter active monitor mode and generate a new token. (For a more detailed description, see section 5.4). The CL_TK values are as follows:

Pm:	Zero	
FC:	03H	
DA:	All stations, this ring	
SC:	0H	
DC:	0H	
VCM:	03H	(Claim Token)
SVI-1:	02H	(Received Upstream neighbour's Address)
SVV-1:	.....	(2- or 6-byte address)

5.2.2 Duplicate Address Test MAC frame (DAT)

This frame is transmitted with DA = MA as part of the initialisation process. If the frame returns with the A bits set to 1, it indicates that there is another station with the same address. If such an event occurs, the station's network manager is notified and the station returns to bypass state. A station that copies a DAT frame will ignore it. The DAT values are:

Pm:	Zero	
FC:	00H	
DA:	MA	(This station's address
SC:	0H	
DC:	0H	
VCM:	07H	(Duplicate Address Test)

5.2.3 Active Monitor Present MAC frame (AMP)

This frame is transmitted by the active monitor. It shall be queued for transmission following the successful purging of the ring or following the expiration of the Timer Active Monitor (TAM). Any station in standby state that receives this frame shall reset its Timer Standby Monitor (TSM). The AMP values are:

Pm:	Pr[3]	
FC:	05H	
DA:	All stations, this ring	
SC:	0H	
DC:	0H	
VCM:	05H	(Active Monitor Present)
SVI-1:	02H	(Received Upstream neighbour's Address)
SVV-1:	.....	(2- or 6-byte address)

5.2.4 Standby Monitor Present MAC frame (SMP)

This frame is transmitted by the standby monitor(s). After receipt of an AMP or SMP frame whose A and C bits equal 0[4], the Timer Queue PDU (TQP) is reset. When timer TQP expires, an SMP PDU shall be queued for transmission.

The queuing of a SMP PDU is delayed for a period of TQP to assure that the transmission of SMP frames do not use more than 1% of the bandwidth of the ring in any TQP period of the time. The SMP values are:

Pm:	Zero	
FC:	06H	
DA:	All stations, this ring	
SC:	0H	
DC:	0H	
VCM:	06H	(Standby Monitor Present)
SVI-1:	02H	(Received Upstream neighbour's Address)
SVV-1:	.....	(2- or 6-byte address)

5.2.5 Beacon MAC frame (BCN)

This frame shall be sent as a result of serious ring failure (for example, broken cable, jabbering station, etc.). It is useful for localising the fault. The transmission of beacon frames is covered in the Standby Monitor Finite-State Machine (section 5.4).

The immediate upstream station is part of the failure domain about which the beacon is reporting. Therefore, as noted above, the address of the upstream station that was previously recorded is included in the MAC INFO field. The BCN values are as follows:

---

[3]An AMP is transmitted at the ring service priority (Pr) that exists at the time a token is received after an AMP PDU is queued. The default value for Pm for this frame is seven; see appendix B.2 to change this value.

[4]Stations that receive an AMP or SMP frame in which the value of the A and C bits are 0 will regard the frame as having originated from their upstream neighbour's station. Therefore, a station that copies such a frame shall record the source address contained in the frame as the SUA for later transmission as a subvector in certain MAC frames as well as performing a comparison with certain MAC frames.

Pm:	Zero	
FC:	02H	
DA:	All stations, this ring	
SC:	0H	
DC:	0H	
VCM:	02H	(Beacon)
SVI-1:	02H	(Received Upstream neighbour's Address)
SVV-1:	.....	(2- or 6-byte address)
SVI-2:	01H	(Beacon Type)
SVV-2:	0001H	Issued by station during reconfiguration (reserved for future use).
	0002H	Continuous J symbols received; signal loss.
	0003H	Timer TNT expired during claiming token; no FR_CL_TK received.
	0004H	Timer TNT expired during claiming token; FR_CL_TK (SA<MA) received.

5.2.6 Purge MAC frame (PRG)

This frame is transmitted by the active monitor. It shall be transmitted following claiming the token or to perform re-initialisation of the ring following the detection of an M bit set to 1 or the expiration of Timer Valid Transmission (TVX). The PRG values are as follows:

Pm:	Zero	
FC:	04H	
DA:	All stations, this ring	
SC:	0H	
DC:	0H	
VCM:	04H	(Purge)
SVI-1:	02H	(Received Upstream neighbour's Address)
SVV-1:	.....	(2- or 6-byte address)

5.2.7 Report New Monitor MAC frame (REP_NM)

This frame is sent by the active monitor, after winning contention, to the network manager that the station is now the new active monitor. The REP_NM values are as follows:

Pm:	Zero	
FC:	00H	
DA:	F(NM)	(functional address of network manager)
SC:	0H	
DC:	4H	
VCM:	25H	(Report New Monitor)
SVI-1:	02H	(Upstream Neighbour's Address)
SVV-1:	.....	(2- or 6-byte address)
SVI-2:	22H	(Product Identification)
SVV-2:	.....	(18-byte ID number)

5.2.8 Report Ring Poll Failure MAC frame (REP_RPF)

This frame is sent by the active monitor to the ring error monitor to report a failure in the neighbour notification process. This frame contains the addresses of the last station that responded in the neighbour notification process before the active monitor detected the failure. The REP_RPF values are as follows:

Pm:	Zero	
FC:	01H	
DA:	F(REM)	(functional address of ring error monitor)
SC:	0H	
DC:	6H	
VCM:	27H	(Report Ring Poll Failure)
SVI-1:	0AH	(Address of last ring poll[5])
SVV-1:	.....	(2- or 6-byte address)

5.2.9 Report SUA Change MAC frame (REP_SUA_CH)

This frame is used in the neighbour notification process to report a change in the stored upstream address of the station upstream from the station generating the Report SUA Change MAC frame. This frame is sent to the functional address of the network manager. The REP_SUA_CH values are as follows:

Pm:	Zero	
FC:	00H	
DA:	F(NM)	(functional address of network manager)
SC:	0H	
DC:	4H	
VCM:	26H	(Report SUA Change)
SVI-1:	02H	(Received Upstream neighbour's Address)
SVV-1:	.....	(2- or 6-byte address)

---

[5]Source address of the last AMP or SMP MAC frame before the poll cycle failed.

5.2.10 Report Monitor Error MAC frame (REP_MON_ERR)

This frame is used to report a problem with the active monitor or the possibility of a duplicate address of stations contending for active monitor. This frame is sent to the functional address of the ring error monitor. The REP_MON_ERR values are as follows:

Pm:	Zero	
FC:	00H	
DA:	F(REM)	(functional address of ring error monitor)
SC:	0H	
DC:	6H	
VCM:	28H	(Report Monitor Error)
SVI-1:	02H	(Received Upstream neighbour's Address)
SVV-1:	.....	(2- or 6-byte address)
SVI-2:	30H	(Error Code)
SVV-2:	0001H	Monitor Error
	0002H	Duplicate Monitor
	0003H	Duplicate Address

5.2.11 Used MAC Vectors and Subvectors

In the previous subsections several MAC vectors and subvectors were defined. Some (sub)vectors will be defined in the next sections. In the following tables these are given in numerical order.

table 5.3 : used MAC vectors

Vector ID	Description
00H	Response (RESP)
02H	Beacon (BCN)
03H	Claim Token (CL_TK)
04H	Purge (PRG)
05H	Active Monitor Present (AMP)
06H	Standby Monitor Present (SMP)
07H	Duplicate Address Test (DAT)
08H	Lobe Media Test (LMT)
0BH	Remove Ring Station (REM_RS)
0CH	Change Parameters (CH_PAR)
0DH	Initialise Ring Station (INIT_RS)
0EH	Request Station Address (REQ_SA)
10H	Request Station Attachment (REQ_AT)
20H	Request Initialisation (REQ_INIT)
22H	Report Station Address (REP_SA)
24H	Report Station Attachment (REP_AT)
25H	Report New Monitor (REP_NM)
26H	Report SUA Change (REP_SUA_CH)
27H	Report Ring Poll Failure (REP_RPF)
28H	Report Monitor Error (REP_MON_ERR)
2FH	Report New Station (REP_NS)

table 5.4 : used MAC subvectors

Subvector ID	Description
01H	Beacon Type
02H	RUA-Received Upstream neighbour's Address
03H	Local Ring Number
06H	Enabled Function Classes
07H	Allowed Access Priority
0AH	Address of last ring poll
20H	Response Code
22H	Product Identification
26H	Wrap data
2BH	Group Address
2CH	Functional Address
30H	Error Code
3FH	Timer Values

5.3 Specification of the Timers, Flags, Registers and, Stacks

As long as a station is active, it has to administrate events in order to prevent the network from any malfunction. The protocol for the MAC layer defines several timers, registers and stacks to operate the network. These will be described in the next subsections.

5.3.1 Timers

The value of these timers shall be established by the mutual agreement among the users of the LAN. The values can be given by several parties using the LAN; By the user during initialisation of the MMTCP, by the ring parameter server (INIT_RS MAC frame) during insertion into the ring network, and by the network manager during operation of the MMTCP (CH_PAR MAC frame). The calculation of the default timer values is given in appendix A.

Timer, return to repeat (TRR):

Each station shall have a timer TRR to ensure that the station shall return to Repeat State. TRR shall have a value greater than the maximum ring latency. The maximum ring latency consists of the signal propagation delay around a maximum length ring plus the sum of all station latencies. The operation of TRR is described in the operational finite-state machine. The default time-out value of TRR shall be 150 microsec.

Timer, holding token (THT):

Each station shall have a timer THT to control the maximum period of time the station may transmit frames after capturing a token. A station may initiate transmission of a frame if such transmission can be completed before timer THT expires. The operation of THT is described in the operational finite-state machine. The default time-out value of THT shall be 150 microsec.

Timer, queue PDU (TQP):

Each station shall have a timer TQP for the purpose of timing the enqueuing of an SMP PDU after reception of an AMP or SMP frame in which the A and C bits were equal to 0. The default time-out value of TQP shall be 150 microsec.

Timer, valid transmission (TVX):

Each station shall have a timer TVX which is used by the active monitor to detect the absence of valid transmissions. The operation of TVX is described in the monitor finite-state machine. The time-out value of TVX shall be 750 microsec.

Timer, no token (TNT):

Each station shall have a timer TNT to recover from various token-related error situations. TNT shall have a time-out value equal to TRR plus n times THT (where n is the maximum number of stations on the ring). The operation of TNT is described in the monitor finite-state machines. The default time-out value of TNT shall be 150 millisec.

Timer, active monitor (TAM):

Each station shall have a timer TAM which is used by the active monitor to stimulate the enqueuing of an AMP PDU for transmission. The default time-out value of timer TAM shall be 750 millisec.

Timer, standby monitor (TSM):

Each station shall have a timer TSM which is used by the stand-by monitor(s) to assure that there is an active monitor on the ring and to detect a continuous stream of tokens. The default time-out value of timer TSM shall be 2 sec.

5.3.2 Flags

Flags are used to remember the occurrence of an particular event. They shall be set when the event occurs. The flags used are:

I flag: A flag which is set upon receiving an ED with the I bit equal to 0 (zero).

SFS flag: A flag which is set upon receiving an SFS sequence.

MA flag: A flag which is set upon receiving an SA which is equal to the station's address.

PCPL flag: A flag, used by the active monitor to determine whether or not it was able to complete a poll cycle (receive its own AMP MAC frame and/or one or more SMP MAC frames).

5.3.3 Registers and Stacks

Timer registers:

The previously defined timers each shall have a register which contains their values.

Address Registers:

Several registers exist to compare the destination address of the received frames with their contents. The result of this comparison is used to decide whether or not a frame is copied. The used registers are:

- Source Address register
- The Group Address registers
- Functional Address register
- The Ring Number register

RUA register:

The value of the received upstream neighbour's address is stored in the RUA register.

Last Ring Poll Address register:

The source address of the last received AMP or SMP MAC frame in stored in the Last Ring Poll Address register.

ID register:

The product identification number is stored in the ID register, which shall be a piece of Read Only Memory (ROM).

Enabled Function Classes:

The class designator is stored in the Enabled Function Classes register.

Allowed Access Priority:

The Allowed Access Priority register contains the number of the maximum service level that may be provided by the network.

Pr and Rr register:

The value of the priority (P) and reservation (R) of the most recently received AC field are stored in registers Pr and Rr.

Sr and Sx Stacks:

If at the time of transmission of a token the value of Rr or Pm (the priority of a queued PDU) is greater than Pr, a token with a priority of the higher of Rr or Pm shall be transmitted. At the same time the station shall store the value of Pr in a stack as Sr and shall store the value of the priority of the token that was transmitted in a stack as Sx

The use of the Pr and Rr registers and the Sr and Sx stacks in performing the priority function is described in detail in the finite-state machines, presented in the next section.

5.4 Specification of the Token Ring Protocols

5.4.1 Standby Monitor Finite-State Machine

The standardised version of this machine is given in appendix B.1. Some states have been modified or added to the machine (see figure 5.14). These states will be described in this subsection in numerical order:

(13) If AMP, SMP, or PRG has been received, a DAT pdu is enqueued for transmission awaiting the receipt of a usable token. Timer TSM is reset, and transition made to Initialise state (state 2).

(33) If the station receives a FR_CL_TK with a source address equal to the station's address and an RUA equal to the SUA, the bid for active monitor has been won. The latency buffer shall be inserted in the ring, the functional address (active monitor) is set, timer TNT reset, and transition made to ACTIVE MONITOR Purge state (state 2).

(34) If a FR_CL_TK MAC frame is received in which the source address is equal to the station's address and the RUA does not equal the SUA, a duplicate address in detected during the Claiming Token State. Timers TNT and TSM are reset, MA_STATUS is indicated to

NMT, and a REP_MON_ERR MAC frame is enqueued for transmission (with subvector value 0003H: duplicate address detected).

(42c) If a FR_SMP whose A and C bits equal zero is received, the SA of the MAC frame is compared to the SUA and a REP_SUA_CH MAC frame is enqueued for transmission if they are not equal. The SA is stored as SUA and is stored as the 'last ring poll address'. The timer TQP is reset.

(42d) If a FR_AMP whose A and C bits equal zero is received, the SA of the MAC frame is compared to the SUA and a REP_SUA_CH MAC frame is enqueued for transmission if they are not equal. The SA is stored as SUA and is stored as the 'last ring poll address'. The timer TQP is reset.

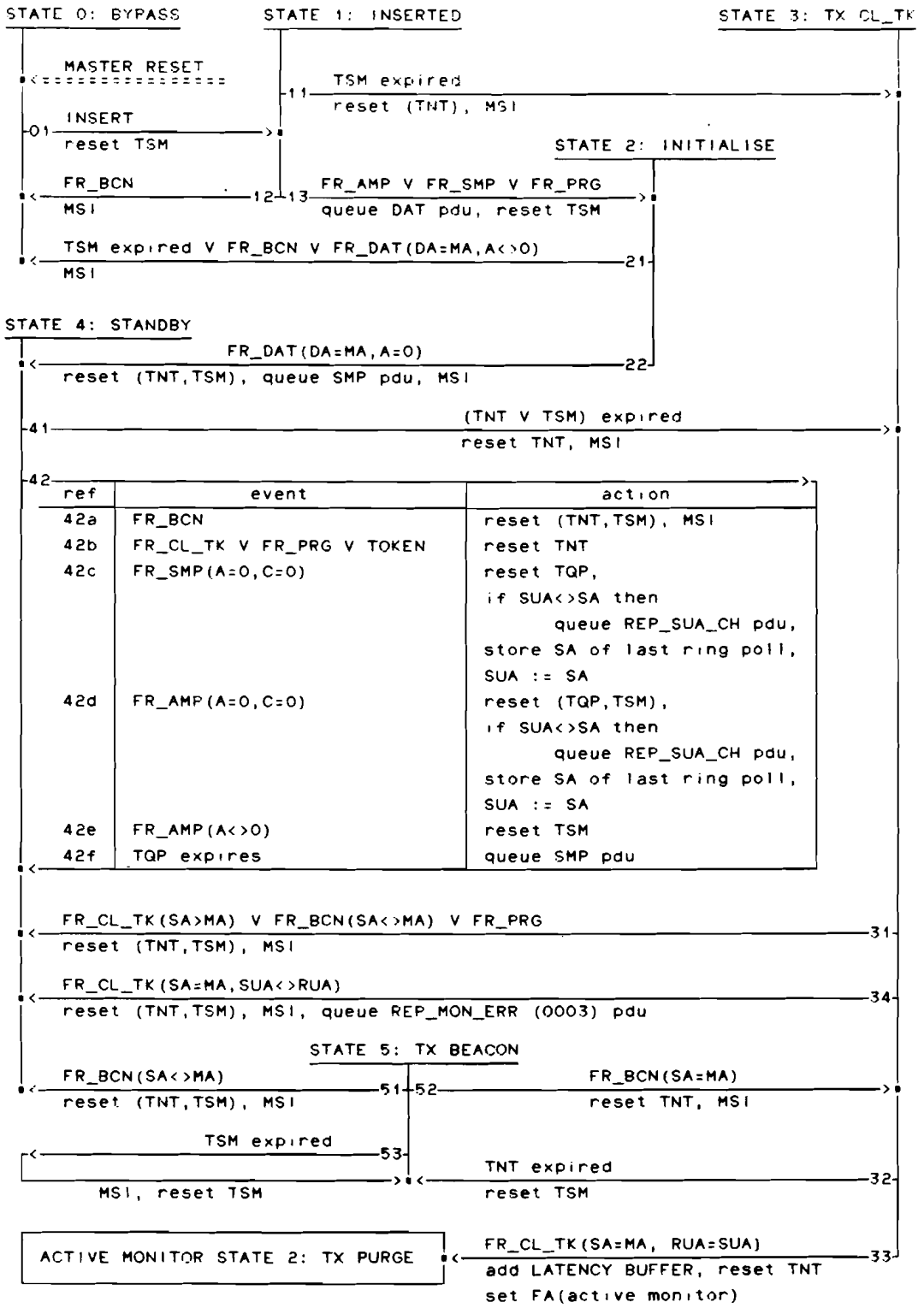


figure 5.14 : **Standby monitor finite-state machine diagram**

5.4.2 Active Monitor Finite-State Machine

The standardised version of this machine has been modified completely (see figure 5.15). Therefore the new machine is described in this subsection in its entirety.

The function of the active monitor is to recover from various error situations such as absence of validly formed frames or tokens on the ring, a persistently circulating priority token or frame. In normal operation there is only one active monitor in a ring at any point in time. Timers TVX, TNT, TAM, and TRR are used by the active monitor.

The active monitor shall utilise its own crystal oscillator to provide timing for all symbols repeated or transmitted on the ring. It also supplies the latency buffer for the ring.

5.4.2.1 state 0: ACTIVE

The active monitor is in this state when the ring is operating normally.

(01a) The M bit is set to one on a token whose M bit is zero and whose priority is greater than zero or a frame whose M bit is zero. Timer TVX is reset.

(01b) Receipt of a token whose M bit and priority are zero will cause timer TVX to be reset.

(01c) If an FR_SMP whose A and C bits equal zero is received, the SA of the MAC frame is compared to the SUA and a REP_SUA_CH MAC frame is enqueued for transmission if they are not equal. The SA is stored as SUA and is stored as the 'last ring poll address'. The PCPL flag is set to indicate the completion of a poll cycle.

(01d) If a FR_SMP whose A bit equals one is received, or if a FR_AMP MAC frame is received in which the source address is equal to the station's address, the PCPL flag is set to indicate the completion of a poll cycle.

(01e) If timer TAM expires and the PCPL flag is set, an AMP pdu is enqueued for transmission, and timer TAM is reset without changing state. The PCPL flag is reset.

(01f) If timer TAM expires and the PCPL flag is reset, an AMP pdu and an REP_RPF (to indicate that no SMP or AMP frame was received) pdu are enqueued for transmission, and timer TAM is reset without changing state. The PCPL flag is reset.

(02) If a frame or token that is being repeated has its M bit equal to one, the frame or token is aborted, timer TNT is reset, and transition made to Transmit Purge State (state 2).

(03) If timer TVX expires, timer TNT is reset, and transition made to Transmit Purge State (state 2).

(04) If the monitor station receives an AMP or PRG frame with a source address that does not equal the station's address, the latency buffer shall be deleted, timers TNT and TSM reset, MA_STATUS indicated to NMT, the functional address (active monitor) reset, and a REP_MON_ERR MAC frame (with subvector value 0002H: duplicate monitor detected) shall be enqueued for transmission, and transition made to STANDBY MONITOR Standby State (state 4).

(05) If the monitor station receives a FR_CL_TK or a FR_PRG with a source address that equals the station's address, the latency buffer shall be deleted, timers TNT and TSM reset, MA_STATUS indicated to NMT, the functional address (active monitor) reset, and a REP_MON_ERR MAC frame (with subvector value 0001H: monitor error detected) shall be enqueued for transmission, and transition made to STANDBY MONITOR Standby State (state 4).

(06) If the monitor station receives a FR_BCN, the latency buffer shall be deleted, timers TNT and TSM reset, MA_STATUS indicated to NMT, the functional address (active monitor) reset, and transition made to STANDBY MONITOR Standby State (state 4).

5.4.2.2 state 1: TRANSMIT FILL

This state exists to assure that all purge frames have been stripped from the ring before transmitting a new token.

(11) When timer TRR expires, a token is transmitted with P equal to Rr, and M and R equal to zero. P is stacked as Sx and a zero is stacked as Sr, timers TVX and TAM are reset, MA_STATUS indicated to NMT, AMP and REP_NM MAC frames enqueued for transmission, and transition is made to State 0.

5.4.2.3 state 2: TRANSMIT PURGE

In this state, PRG MAC frames are continuously transmitted to purge the ring before transmitting a token.

(21) If the station receives an FR_PRG whose source address is equal to the station's address and with a subvector equal to UNA, timer TRR is reset and transition is made to Transmit Fill State (state 1).

(22) If timer TNT expires while waiting for receipt of the station's source address, the latency buffer is deleted, timers TNT and TSM reset, MA_STATUS indicated to NMT, the functional address (active monitor) reset, and transition made to STANDBY MONITOR Standby State (state 4).

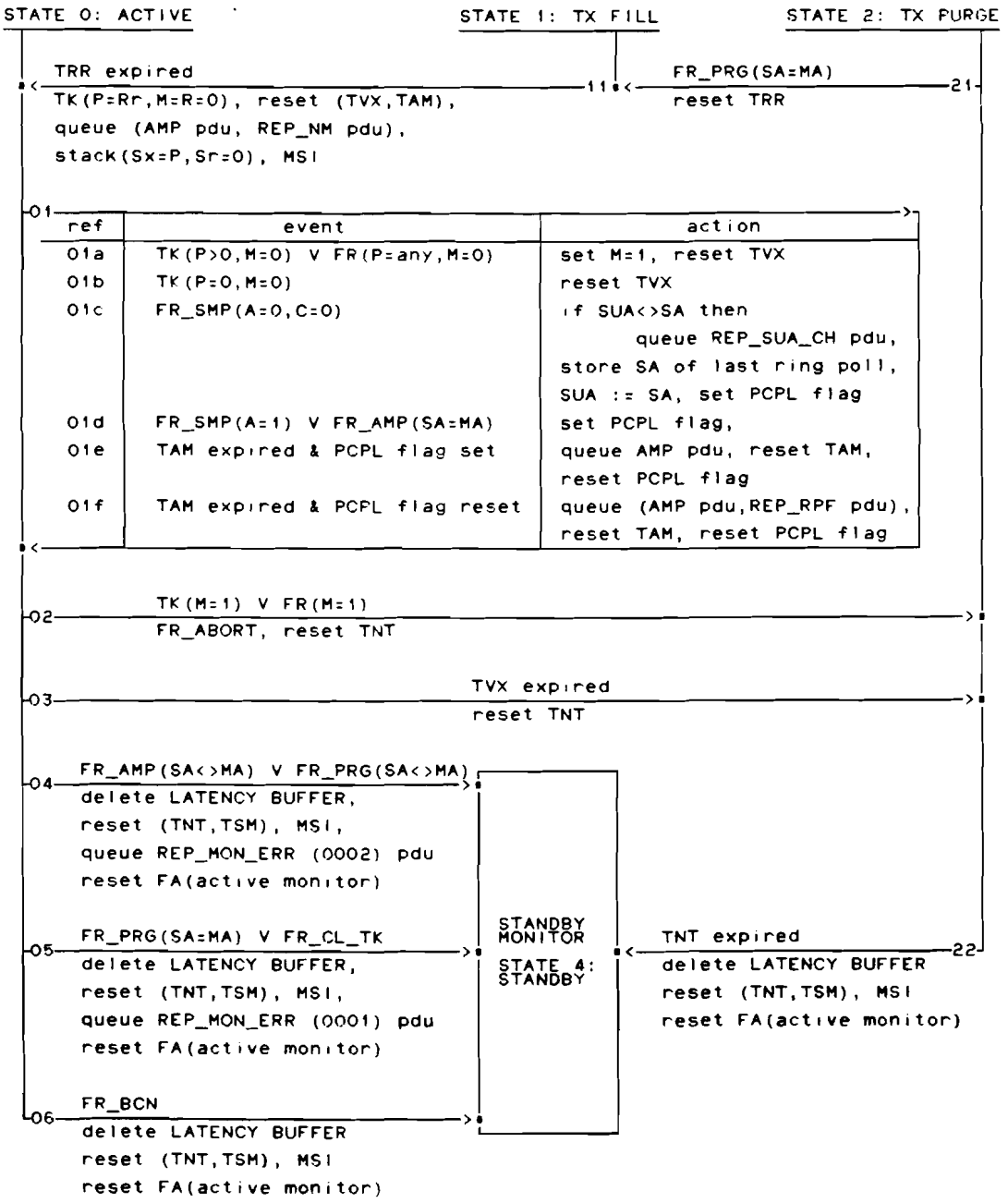


figure 5.15 : Active monitor finite-state diagram

5.5 Service Specifications

The services provided by the lowest layers of the OSI model (in relation to the Token Ring Protocol) are defined in the ANSI/IEEE standard 802.5. This standard specifies the services provided

- by the MAC layer to the LLC layer
- by the PHY layer to the MAC layer
- by the MAC layer to the Network Management
- by the PHY layer to the Network Management

They remarked on the service specifications that these are described in an **formal** way and do **not** imply any particular implementation. The specification of the service primitives (the reprint of chapter 5 of the 802.5 standard) is given in appendix B.2. A few remarks on them are given below.

5.5.1 Remarks on the Standardised Service Specifications

MA_DATA.request: none

MA_DATA.indication: The requested_service_class parameter will be added to the parameter list. The value of this parameter equals the priority (Pm) that was desired for the data unit transfer (at its MA_DATA.request).

MA_DATA.confirmation: The transmission_status parameter value has to be defined.

PH_DATA.request: none

PH_DATA.indication: none

PH_DATA.confirmation: The transmission_status parameter value has to be defined.

MA_INITIALISE_PROTOCOL.request:

The address length will be initialised by this primitive and may have the value 2 or 6:
- ADDRESS_LENGTH(value)

Several group_MAC_addresses may exist; one bit-significant and more conventional group addresses.

The indicate_for_rcv_only_good_frames parameter always will be true; frames with an error never will be passed to an other layer, since this is only a waste of memory bus bandwidth.

The ring number checking options will be initialised by this primitive:
- CHECK_NOT_EQUAL_RING
- CHECK_RING_TABLE

The use of these options is discussed in the next chapter about "bridges".

MA_INITIALISE_PROTOCOL.confirmation:

The status parameter has to be defined.

MA_CONTROL.request: none

MA_STATUS.indication: none

MA_NMT_DATA.request:none

MA_NMT_DATA.indication:

The requested_service_class parameter will be added to the parameter list. The value of this parameter equals the priority (Pm) that was desired for the data unit transfer (at its MA_NMT_DATA.request).

MA_NMT_DATA.confirmation:

The transmission_status parameter value has to be defined.

5.6 Specification of the Physical Layer

The physical layer specification will be adopted from the 802.5 standard (chapter 6). This section defines the data symbol encoding and decoding, symbol timing, and reliability. Chapter 7 of the standard describes the functional, electrical, and mechanical characteristics of balanced, baseband, shielded twisted pair attachment to the trunk cable of a Token Ring. This chapter will be fully adopted too. Both chapters are contained in this report in appendix B.3.

5.7 Medium Interface Testing

Before inserting the station into the ring, we sometimes want to test the medium between the station and the Trunk Coupling Unit (if available). This is done by transmitting a test frame, which is defined below. This frame is transmitted 64k times. To complete the testphase, a DAT MAC frame is transmitted to itself. This ensures, that the receive functions of the station are working correct. If this frame is not received correctly, the station will retry once. The Medium Interface Test shall be optional.

5.7.1 Lobe Media Test MAC frame (LMT)

This frame is used by the station in the insertion process to test the continuity of a wire in a loop back path. This occurs prior to physical insertion in the ring. This frame is ignored by the station when it is received. The LMT values are as follows:

Pm:	Zero	
FC:	00H	
DA:	Null address	
SC:	0H	
DC:	0H	
VCM:	08H	(Lobe Media Test)
SVI-1:	26H	(Wrap data)
SVV-1:	.....	(about 250 data bytes)

5.8 Conclusions and Remarks

This chapter, together with the appendices B.1 through B.3, replaces the ANSI/IEEE standard 802.5. Some parts of the standard that were not specified completely or had to be specified, are specified now. Also some modifications are made to improve the quality of the network during start-up and during the operation of the network.

Some MAC frames that were introduced are adopted from the Texas Instruments implementation of the 802.5 standard, including the vector and subvector identifiers. We did not check whether or not the newly defined MAC frame identifiers were consistent with identifiers of other manufacturers of a Token Ring controller (if present), except with the implementation of Texas Instruments.

5.8.1 Implementation notes on the Token Ring system

The concept of the layering of a communication system is a very powerful one. It gives the possibility of making a well-structured, hierarchical design. If a system is standardised (or designed) in a proper way, it will fit perfectly into a layered model of a communication system. Unfortunately this is not the case with the Token Ring system. It is obvious that the Token Ring system is designed as a whole first and then poured into the LAN model (figure 4.1) by force. This will be illustrated by the following examples.

- The latency buffer lies in the PHY layer. Its control however, is done by the Active Monitor and Standby Monitor Finite State Machines, who lie in the MAC layer. Extra control lines between the two layers are needed to control this buffer.

- The PHY layer decodes symbols out of the (Manchester encoded) data stream. To do this properly, The decoder has to be synchronised with the data stream with respect to Starting and Ending Delimiters. This synchronisation however, is done in the MAC layer who collects the decoded symbols and waits for a synchronisation event.

- When the PHY layer detects a signal loss event (more than four half bit-times with the same polarity in succession), a signal loss is indicated to Network Management via a PH_STATUS primitive. The MAC layer needs this information too (in the Standby Monitor F.S.M. in Beacon State).

Therefore, we suggest that the boundary between the MAC layer and the PHY layer is not kept too strictly. Besides, the exact position of the MAC

layer in the OSI Reference Model is in discussion at the moment; now it is a part of the Data Link Layer, in future it probably will be a part of the Physical Layer. This would be a logical step, since the service provided by the MAC layer is to transmit frames on and to receive frames from the medium; the service provided by a Physical Layer.

The presented Operational F.S.M. (appendix B.1) can be realized using a simple (eventually coupled) state-machine. The speed of this machine must be high, because its reaction time must be in the order of the bit rate on the communication channel. Therefore this F.S.M. cannot be realised as a dedicated microprocessor. On the opposite, the presented Monitor F.S.M.'s should be realised with some kind of dedicated processor. Most of the F.S.M. outputs trigger a 'hardware subroutine', who can be implemented as a piece of a program running on this processor. This is possible because the actions of the Monitor F.S.M.'s are not as time critical as the Operational F.S.M.

6. THE NETWORK MANAGEMENT LAYER

6.1 Relation of the Network Management Layer to the ISO/OSI Model

Observing the ISO/OSI Reference Model, we see that a Network Management Layer (NMT) can be thought 'parallel' to the 7 layer model as illustrated in figure 6.1:

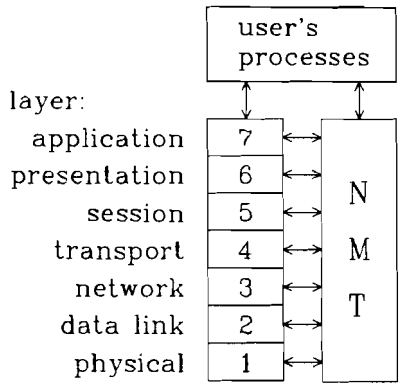


figure 6.1 : Relation of the Network Management Layer to the ISO/OSI Reference Model

In most cases, processes make use of this model by requesting a service of the (highest) layer of this model. However, some processes can control the communication protocols, running in the different layers. This is done via the NMT. This control concerns timer values, priority levels, address numbers, etc. Besides processes on the users machine these control processes can be running on a machine somewhere in the ring or the network.

Because we already have decided that the upper three layers will be handled by the MMTCP, we only will be interested in a network management system for the transport layer and below.

In most cases the NMT actions involve more than one layer of the OSI model at each time. Thus when we would implement a NMT building block for each adjacent layer, the MMTCP bus would be used inefficiently. The interaction between those building blocks again would be a waste of bus bandwidth. To prevent this, the NMT will be implemented in only one building block, as illustrated in figure 6.2. Note that the MAC layer must be transparent for the Network Management Control of the PHY layer.

In the next sections we will discuss the NMT actions that are related to the control of the MAC and PHY layers. The involved NMT protocol will not be specified here. It is reasonable to wait for this specification until the standard 802.1 is available. [9] will give an idea of this standard that is expected to come out in 1988.

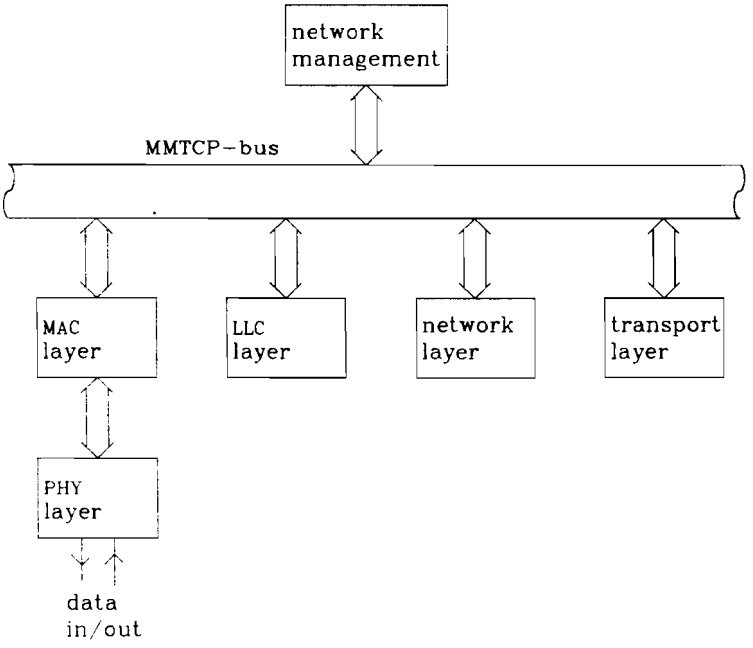


figure 6.2 : Architecture of the MMTCP with the NMT building block

6.2 Management of the MAC and PHY Layers

Observing the PHY layer, no special management functions for this layer are known. The interaction of the NMT to the PHY layer only concerns passing of commands to the PHY layer and indicating events to the NMT.

Besides these two interactions, the MAC layer provides the same service to the NMT as to the LLC layer; transmission of frames using the same type of service primitives (appendix B.2). Every frame that cannot be processed by the MAC layer is indicated to the NMT too. Also initialisation of the MAC layer is done via the NMT layer.

6.2.1 Network Management Processes

Several processes are designated to the NMT to increase the performance of the network. These processes are:

- Network Manager (NM), this process of the network management monitors and modifies the state of individual stations and that of the LAN as a whole.
- Ring Parameter Server (RPS), this process can assign operating parameters of individual stations and of the LAN during the time a station is inserting on the ring.
- Ring Error Monitor (REM), this process serves as a collection point of error reports for network management.

These processes, which are always running on one or more systems (on each ring!) must guarantee that every station can use the system in a way, such that the messages with the highest priority always are transmitted in the shortest time possible. This is done by a dynamic control of the timer values of all MAC layers in the ring(s). Errors must be localised and serviced as soon as possible. Note that it is not necessary to know the exact location(s) of these processes, since they can be addressed by a functional MAC address. As already indicated, these processes (and their relations) have not been defined at the moment. This will be done at a later phase.

6.2.2 Start-up Process

Immediately after power-up, the MAC layer shall perform some kind of self-test. After this, the station can insert in the ring and begin operation. Immediately after inserting in the ring, the NMT will request the RPS and NM to initialise the station with the correct parameters to operate on the ring. To do this, some MAC frames are defined. As soon as the new parameters are received, they are passed to the MAC layer with the MA_INITIALISE_PROTOCOL.request. The MAC layer responds to this request with a confirm primitive. During operation the station may request new parameters or may get these without asking for them.

6.2.3 User Processes

If a station receives a MAC frame that is not known by the MAC layer, the frame is passed to the NMT. If the NMT does not know how to process this frame, the frame is passed to the user (or to a designated user process). Otherwise the frame is processed by the NMT. The frames that can be processed by the NMT are defined in the next section.

6.3 Specification of the MAC frames

The following are descriptions of various MAC frames that are used in the management of the Token Ring. Values for PDU priority (Pm), FC, DA, and INFO field content (VI, SVI, and SVV) associated with the particular MAC Supervisory Frame, are indicated.

6.3.1 Response MAC frame (RESP)

This frame is used to send positive responses to frames that require acknowledgement, or to report errors in syntax in a MAC frame sent to the station. The structure of the SVV format is given in figure 6.3. The source class, destination class and vector command are generated from the contents of the vector in the received MAC frame that caused the station to send the Response MAC frame. The RESP values are as follows:

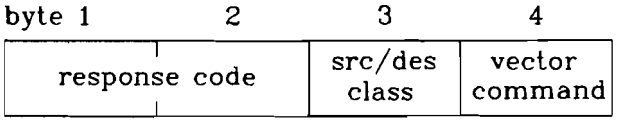


figure 6.3 : The subvector value format

Pm: Zero
FC: 00H
DA: source address of received MAC frame
SC: 0H
DC: source class of received MAC frame
VCM: 00H (Response)
SVI-1: 20H (Response Code)
Response Code:

0001H — Positive Response (ACK); Sent in response to MAC frames requiring positive acknowledgement of receipt.

8001H — MAC Frame Data Field Incomplete; The MAC frame was too short to contain the VI and VL (less than 4 bytes).

8002H — VL Invalid; The VL did not agree with the length of the frame or a subvector was found that did not fit within the vector.

8003H — VCM Not Supported; The VI was not recognised by the station.

8004H — Inappropriate Source Class; The source class in the VI is not valid for the vector.

8005H — SVL Invalid; The length of a recognised subvector is longer than the maximum allowed.

8007H — Required Subvector Missing; A subvector required by the station is not in the frame.

8008H — Required Subvector Unknown; A subvector received in the MAC frame that is marked required is not known by the station. This response is also generated if a required subvector is duplicated in the vector.

8009H — MAC Frame Exceeds Maximum Length; The received frame is rejected because it did not fit in one buffer.

800AH	Function Disabled; The received MAC frame is not executed because the function requested is disabled.

6.3.2 Remove Ring Station MAC frame (REM_RS)

This frame is sent by the network manager to request the station to de-insert itself from the ring. If the station is inserted, the station de-inserts from the ring and reports 'Remove Received' to the MMTCP system. The REM_RS values are as follows:

Pm:	Zero	
FC:	00H	
DA:	target station	
SC:	4H	
DC:	0H	
VCM:	0BH	(Remove Ring Station)

6.3.3 Report New Station MAC frame (REP_NS)

This frame is sent by the station as soon as it has been inserted on the ring. It is sent to the network manager, which responds with a Change Parameters MAC frame. The REP_NS values are as follows:

Pm:	Zero	
FC:	00H	
DA:	F(NM)	(functional address of network manager)
SC:	0H	
DC:	4H	
VCM:	2FH	(Report New Station)
SVI-1:	22H	(Product Identification)
SVV-1:	.....	(18-byte ID number)

6.3.4 Change Parameters MAC frame (CH_PAR)

A CH_PAR MAC frame is sent by the network manager to change certain parameters in the station. This frame will be accepted in response to a Report New Station MAC frame transmitted by a station in the insertion process. The CH_PAR values are as follows:

Pm:	Zero	
FC:	00H	
DA:	source address of received REP_NS frame	
SC:	4H	
DC:	0H	
VCM:	0CH	(Change Parameters)
SVI-1:	03H	(Local Ring Number)
SVV-1:	0....	
SVI-2:	06H	(Enabled Function Classes)
SVV-2:	0....	
SVI-3:	07H	(Allowed Access Priority)
SVV-3:	00000xxx	
SVI-4:	2BH	(Group Address)
SVV-4:	1....	
SVI-5:	3FH	(Timer Values)
SVV-5:	word-1:	Timer, Return to Repeat (TRR)
	word-2:	Timer, Holding Token (THT)
	word-3:	Timer, Queue PDU (TQP)
	word-4:	Timer, Valid Transmission (TVX)
	word-5:	Timer, No Token (TNT)
	word-6:	Timer, Active Monitor (TAM)
	word-7:	Timer, Standby Monitor (TSM)

6.3.5 Request Initialisation MAC frame (REQ_INIT)

This frame is sent by the station to request operational parameters from the ring parameter server. The values of REQ_INIT are as follows:

Pm:	Zero	
FC:	00H	
DA:	F(RPS)	(functional address of ring parameter server)
SC:	0H	
DC:	5H	
VCM:	20H	(Request Initialisation)
SVI-1:	02H	(Upstream Neighbour's Address)
SVV-1:	.....	(2- or 6-byte address)
SVI-2:	22H	(Product Identification)
SVV-2:	.....	(18-byte ID number)

6.3.6 Initialise Ring Station MAC frame (INIT_RS)

INIT_RS MAC frames are sent by the ring parameter server to set parameters in the station when the station is in the insertion process and transmits a Request Initialisation MAC frame. When received, this frame can set some of the parameters within the station. The INIT_RS values are as follows:

Pm:	Zero	
FC:	00H	
DA:	target station	
SC:	5H	
DC:	0H	
VCM:	0DH	(Initialise Ring Station)
SVI-1:	03H	(Local Ring Number)
SVV-1:	0....	
SVI-2:	2BH	(Group Address)
SVV-2:	1....	
SVI-3:	3FH	(Timer Values)
SVV-3:	word-1:	Timer, Return to Repeat (TRR)
	word-2:	Timer, Holding Token (THT)
	word-3:	Timer, Queue PDU (TQP)
	word-4:	Timer, Valid Transmission (TVX)
	word-5:	Timer, No Token (TNT)
	word-6:	Timer, Active Monitor (TAM)
	word-7:	Timer, Standby Monitor (TSM)

6.3.7 Request Station Attachment MAC frame (REQ_AT)

This frame is sent by the network manager to respond with a Report Station Attachment frame. The purpose of this frame is to get more (or an update of the) information on a station. The REQ_AT values are as follows:

Pm:	Zero	
FC:	00H	
DA:	target station	
SC:	4H	
DC:	0H	
VCM:	10H	(Request Station Attachment)

6.3.8 Report Station Attachment MAC frame (REP_AT)

This frame is sent by the station in response to the Request Station Attachment MAC frame, which was sent by the network manager. The REP_AT values are as follows:

Pm:	Zero	
FC:	00H	
DA:	source address of received REQ_AT frame	
SC:	0H	
DC:	4H	
VCM:	24H	(Report Station Attachment)
SVI-1:	06H	(Enabled Function Classes)
SVV-1:	0....	
SVI-2:	07H	(Allowed Access Priority)
SVV-2:	00000xxx	
SVI-3:	22H	(Product Identification)
SVV-3:	.....	(18-byte ID number)
SVI-4:	2BH	(Group Address)
SVV-4:	1....	
SVI-5:	2CH	(Functional Address)
SVV-5:	0....	

6.3.9 Request Station Address MAC frame (REQ_SA)

This frame is sent by the network manager to request the station to respond with a Report Station Address MAC frame. The purpose of this frame is to administrate how a station can be addressed. The REQ_SA values are as follows:

Pm:	Zero	
FC:	00H	
DA:	target station	
SC:	4H	
DC:	0H	
VCM:	0EH	(Request Station Address)

6.3.10 Report Station Address MAC frame (REP_SA)

This frame is sent by the station in response to the Request Station Address MAC frame, which was sent by the network manager. The REP_SA values are as follows:

Pm:	Zero	
FC:	00H	
DA:	source address of received REQ_SA frame	
SC:	0H	
DC:	4H	
VCM:	22H	(Report Station Address)
SVI-1:	02H	(Upstream Neighbour's Address)
SVV-1:	.....	(2- or 6-byte address)
SVI-2:	2BH	(Group Address
SVV-2:	1....	
SVI-3:	2CH	(Functional Address)
SVV-3:	0....	

6.4 Remarks

In our opinion, all presented network management MAC frames are essential to achieve a proper management of a Token Ring network. This does not mean that all presented frames have to be processed within the NMT building block. Much of the network management can be done in a process, running on the host processor. Maybe the Network Management standard 802.1 comes up with a set of (slightly?) different MAC frames. The presented set of frames then again should be subject of discussion.

The proposed timer-value parameters are valid for the MMTCP Token Ring controller only. Therefore, controllers always identify themselves before they get a (new) set of parameters. Unfortunately, these parameters have not been subject of standardisation and probably never will be, since each manufacturer has its own idea of implementing a Token Ring controller; Texas Instruments for instance did not use the standardised timers.

7. BRIDGING BETWEEN IEEE 802 LOCAL AREA NETWORKS

7.1 Introduction

definition: a bridge is a device that is used to interconnect any of the LAN technologies, defined by the IEEE 802 standard committee, transparently.

The committee is currently concentrating effort on the standardisation of such a MAC layer bridge. A bridge is seen as a vital piece of equipment to extend a LAN. It acts as a station on the medium to which it is attached. It can buffer frames and perform transmission speed changes in switching data from one LAN to another.

Why do we want to interconnect (different) LAN technologies towards each other? There are numerous reasons, the most important are the following:

- When many processes on a LAN communicate intensively, the LAN gets congested. In this case it is better to install one or more extra LAN's that are interconnected.
- It is possible, that a CSMA/CD LAN (e.g. Ethernet) is already available at a plant and we want to connect a Token Ring LAN with it. For example, we have two Token Rings on a distance from each other that are connected with each other through an existing link.

The result of a network planning thus can be a network with an arbitrary topology, which is called an 'extended LAN' (an example is given in figure 7.1). As we can see, there are 16 paths between station A and station B, so a routing strategy becomes necessary.

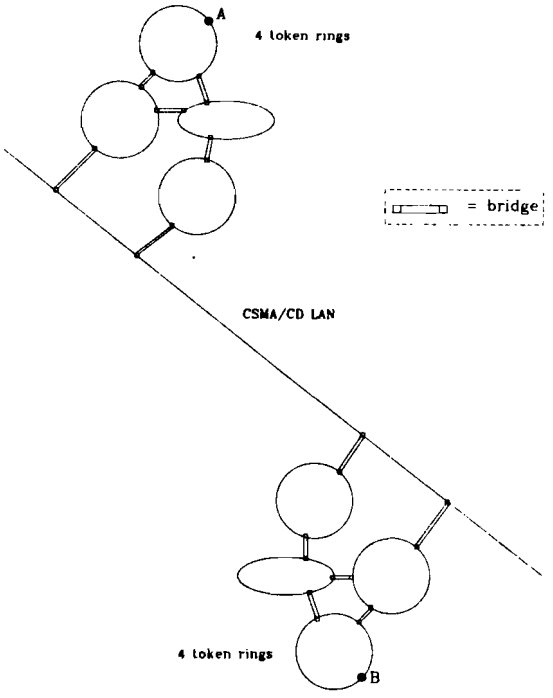


figure 7.1 : Example of an extended LAN

7.2 Bridging Techniques

In the previous paragraph, we mentioned that the interconnection of the LAN's will be done at the MAC level. The reason for this is the following:

The extended LAN is seen as one, big network. The communication between networks is handled by the network layer, who does not handle the routing within a network. Therefore this has to be done by one of the lower layers. Because the IEEE 802 LAN's have the same interface to the surrounding layers (MAC layer to the LLC layer and to the NMT layer) and the address formats are the same (a 16-bit address with a 7-bit segment number or a 48-bit address with a 14-bit segment number, the segment number designates the LAN), the MAC layers can be interconnected transparently [10].

How the bridges can be seen in the ISO/OSI model is illustrated in figure 7.2. Note that this technique provides independence for all higher layer protocols running either in connection-less or connection-oriented mode.

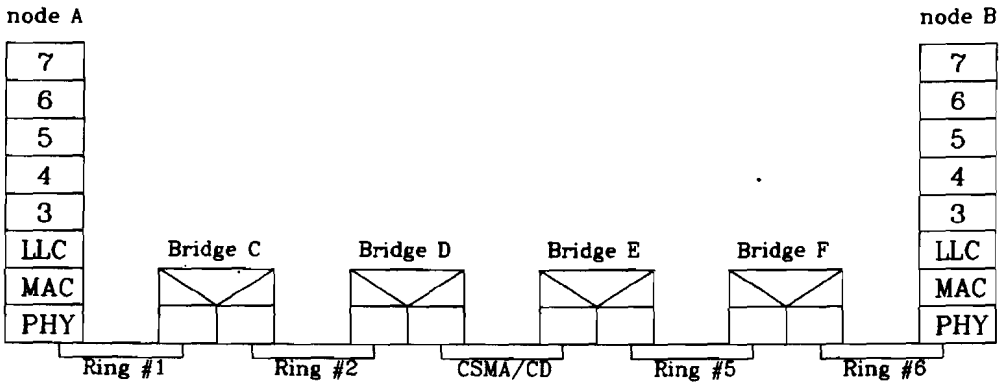


figure 7.2 : Bridges within the ISO/OSI model

Three bridging techniques are under discussion at the moment. These are:

- source routing
- spanning tree algorithms
- self-learning bridges

Besides these, numerous other techniques exist. In most cases, these were not as powerful as the techniques mentioned above, or were too complex. In the following subsections the three techniques will be explained.

7.2.1 Source Routing

The destination address of a frame contains only enough information to route a frame to a node on the same LAN as the source node. Additional information, used by bridges to decide which frames to transport across LAN boundaries, is contained in a field called the routing information field. When this additional information is supplied by the originator of the frame, the routing control is termed 'source routing'. This source routing information is contained within the Routing Information Indicator field of a frame.

The routing information field is variable, up to 32 bytes in length, and its presence is indicated by the most significant bit in the source address field of a frame (the individual/group bit) as indicated in figure 7.3. If this MSB is set to one, a routing information field may be found immediately following the source address and prior to the information field.

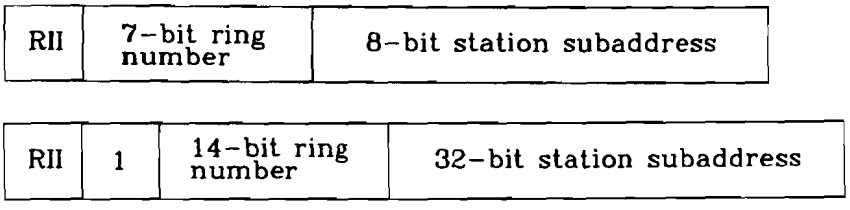


figure 7.3 : Source address format including routing information indicator

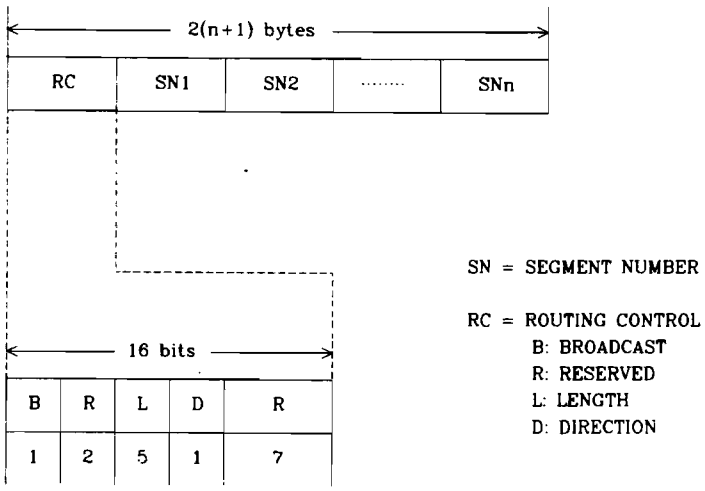


figure 7.4 : Routing information field format[6]

The format of the routing information field is shown in figure 7.4. Its contents will be explained below.

B Broadcast. This bit, when set to one, indicates that the frame is destined for all LAN's. It does not imply that the frame is destined for all stations on all LAN's.

R Reserved. These bits are reserved and set to zero for transmission and ignored on reception.

L Length. This field indicates the length of the routing information field in bytes including the routing control field.

D Direction. This indicates to a bridge whether a frame is travelling from the originating station to the target or the other way around. This bit allows the segment numbers to appear in the same order regardless of the direction of transmission.

SNx Segment Numbers. These 16-bit fields indicate the path between nodes on different LAN's. These will contain the segment number of the next LAN to be addressed.

Not included in the previous discussion on source routing is how the source station obtains knowledge on how to route a packet. In most cases, all routes are given to the communication controller in the form of a table

[6]Adopted from Texas Instruments Token Ring Adapter Chipset User's Guide TMS380.

(which probably will be a large database) during initialisation. Once the initial routes are given, they can be updated by a network manager.

7.2.2 Spanning Tree Algorithms

definition: a spanning tree of a network is a subgraph containing all the nodes of the graph and some collection of arcs chosen so, that there is exactly one path between each pair of nodes.

Because there exist 16 paths between station A and station B of figure 7.1, it is not a spanning tree network. By making some bridges inactive, a spanning tree network is obtained as shown in figure 7.5.

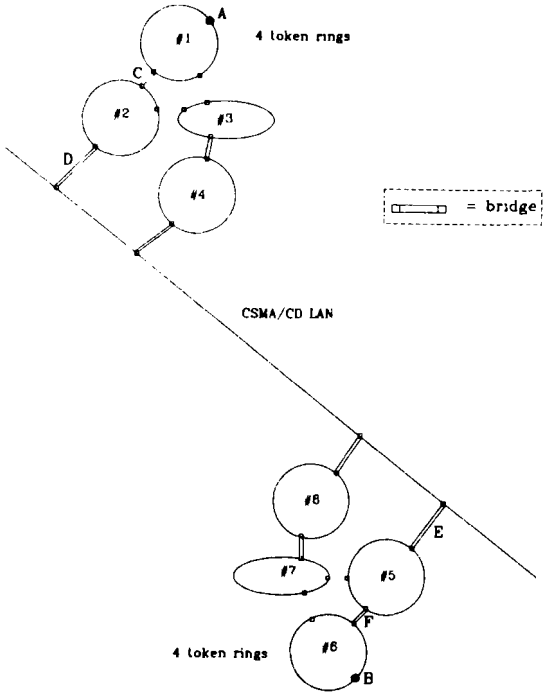


figure 7.5 : Example of a spanning tree network

Depending on the performance of the network, the topology is updated regularly. Therefore all bridges are equipped with some kind of inter-bridge protocol.

7.2.3 Self-learning Schemes

Within these systems a technique is used to optimise the bridge-behaviour, by observing the data stream on both sides of it. Usually complex algorithms are needed for these type of bridges. In the following paragraph the behaviour is simplified enormous.

In the beginning, all frames with a destination segment number different from the actual segment number are copied. When the bridge detects that an

acknowledge on the frame(s) never is received from the other side, its stops copying frames for that segment number for a while, because probably another bridge was used to route the frame(s).

The characteristic property of these bridges is, that they can be removed from a network or inserted in a network at any time without the network going down and that they do not communicate with other bridges.

7.2.4 Standardisation of a Bridge

At the moment it is not known, which of the three techniques is going to be standardised. Since the Routing Information Indicator bit is not a part of the LAN standards, these standards would have to be adjusted. Therefore the source routing technique probably will be dropped first. Besides this, it is not a positive aspect, that the source needs to transmit extra data on the LAN's and needs a routing database.

The spanning tree algorithms have the disadvantage, that only one path is available between each pair of nodes. Besides this disadvantage, the bridges communicate intensively to keep the paths as optimal as possible. Thus the hardware resources (the 'disabled' bridges) are not used optimally, which can result in a significantly longer path between two nodes than necessary. However, if communication between two LAN's becomes more intensive, a new path between these LAN's may be chosen, since the bridges always try to use the most optimum path.

The major advantage of the spanning tree algorithms is the simplicity of the routing control; the algorithms to determine the optimum path are (relatively) simple and, since there is only one path between each pair of nodes, the routing itself becomes simple.

The major disadvantage of the self-learning bridges is their inability to recover quickly from changes in the network topology and network load. The reason for this is the (relatively) slow learning process, compared to the spanning tree algorithms. Their advantage is, that they do not need any inter-bridge communication and thus waste less bandwidth. Because of the slow learning process this advantage is of minor importance. The major advantage is the better use of the hardware resources since they optimise paths between all nodes on the shortest distance.

7.3 Impact on the implementation of the MMTCP

At the moment it is of no interest which routing strategy becomes a standard. The reason for this is that the MMTCP only will perform a basic address check on a frame that is sent on the ring. The rest of the bridge will be implemented as a piece of software, running on the host processor.

We implement the bridge in software and not in hardware because the received frames are never used within the MMTCP, but always on another LAN. The frames have to be copied to the host anyway.

During initialisation the following MMTCP switches are given.

- BRIDGE_STATION
- ACCESSIBLE_SEGMENT_TABLE

If the first switch value is 'true', the MMTCP acts as a bridge on the Token Ring LAN. If the second switch is 'true', a table is stored in memory

to compare the received destination addresses with. The check is done on the segment number of the address; if the number is in the list, the frame is copied. During operation the following address check is done:

```
if (destination_segment_number <> this_ring_number)
then if ((not accessible_segment_table) or (destination_segment_in_table))
     then (copy_frame_to _bridge_process);
```

As soon as a frame is copied from the ring, a message is passed to the Task Restarter of the MMTCP (to be placed in the mailbox buffer). Since the bridge process is continuously waiting for a received frame, this process is restarted. Besides this, a message is given to the Bridge Data DMA Controller (of the MMTCP) to order the copying of the frame to the bridge process.

The transmission of a frame by a bridge process is very simple; the Host Command Interpreter receives the command, orders the Bridge Data DMA Controller to copy the data into the MMTCP working memory, and orders the transmission to the MAC layer if the frame was not intended for this MMTCP. Otherwise the Host Command Interpreter passes the frame to another MMTCP building block.

7.4 IBM's Bridge Backbones

Articles of IBM research on Token Ring networks [6,8] describe the possibility to connect all bridges together with a high-speed backbone (not necessarily an IEEE 802 LAN). This backbone can be a high-speed Token Ring or a broadband cable system, for example.

These methods do not have any impact on the MMTCP design, since only the address decoding is done within its MAC layer.

7.5 Conclusions

The additions to the MAC layer for the implementation of a bridge are minor, while the implementation flexibility is major.

Which bridging technique will be used is of no importance, the MMTCP can handle its most basic function; the address check. The rest of the bridge is implemented using one or more processes, running on the host processor.

8. THE HIGHER LAYERS OF THE COMMUNICATION CONTROLLER

In this chapter we want to suggest what the contents of the higher layers of the communication controller (LLC through Transport Layer) should be.

8.1 The LLC Layer

8.1.1 LLC Types and Classes

The LLC standard 802.2 defines two types of operations for data communication between service access points; Type 1 and Type 2.

With Type 1 operation, PDU's are exchanged between LLC's without the need for the establishment of a data link connection. In the LLC sublayer, these PDU's shall not be acknowledged, nor shall there be any flow control or error recovery in the Type 1 procedures. The service, given in Type 1 operation, is often referred to as a 'datagram or a connection-less' service, which means that the LLC layer simply accepts messages from the network layer and attempts to deliver each one as an isolated unit.

With Type 2 operation, a data link connection shall be established between two LLC's prior to any exchange of information-bearing PDU's. The normal cycle of communication between two Type 2 LLC's on a data link connection shall consist of the transfer of PDU's containing information from the source LLC to the destination LLC, acknowledged by a PDU in the opposite direction. With this operation type, traffic control between the source and destination LLC is executed. This is done by the means of a numbering scheme, which is cyclic within a modulus of 128. An independent numbering scheme shall be used for each source/destination pair, which shall be defined to be a logical point-to-point data link connection between two Data Link Layer service access points. The service, given in Type 2 operation, is often referred to as a 'virtual circuit or a connection-oriented' service, which means that the LLC layer provides the network layer with a perfect channel.

Two classes of LLC's are defined. A Class I LLC shall support Type 1 operation only, a Class II LLC shall support both Type 1 and Type 2 operations. This means that all LLC's on a LAN shall have Type 1 operation in common.

8.1.2 LLC Service Access Points

Each LLC PDU shall contain two address fields: the Destination Service Access Point (DSAP) and the Source Service Access Point (SSAP). The DSAP field shall identify the SAP(s) for which the LLC information field is intended. A number of specific SAP codes have been identified for particular uses. These are listed in table 8.1.

table 8.1 : LLC SAP code points

code point	usage
00H	LLC layer
08H	network management facility
0CH	user tasks
10H	
14H	
.	
.	
.	
F8H	
FCH	
FEH	network layer
FFH	broadcast (= all SAP's)

8.1.3 Implementation of the LLC layer

The LLC layer must be able to deliver messages with a high priority faster than messages with a low priority, thus not necessarily in the order requested by the network layer. This is already the case for the Token Ring controller, who only delivers messages with a priority higher than or equal to the ring priority, with the highest priority first. We claim that the LLC layer in that case must be a Class I LLC layer, who offers a connection-less data link service.

Between two LLC SAP's there may only be one data link connection in a Class II LLC. This means that between two network layers there is only one perfect data link, on which data is delivered in the order as it was sent. Thus the Class II LLC may not be used in a LAN where the priority option is used. The Class I LLC layer therefore must be implemented for the communication controller. A rather complex Transport Layer is the consequence of this.

8.2 The Network Layer

The service provided to the Transport Layer is described by the ISO standard IS8348. This standard only deals with the connection-oriented network service . However, this service does not cover all applications. Since LAN's demand very fast gateways at points of interconnection, the need became for much simpler protocols and systems than those designed for connection-mode data transfer. It became apparent that the connection concept had to be joined with the (complementary) concept connection-less data transmission. Therefore an addendum, Addendum 1, was written. This addendum covers the connection-less network service.

The ISO Internetwork Protocol (IS8473) is a subnetwork-independent protocol of the Network Layer through which a connection-less network service is provided to the Transport Layer. Thus the Internetwork Protocol provides the capability to transmit data without requiring that any logical relationship is maintained among PDU transmissions. The assumption is, that

some higher layer protocols are expected to provide the logical relationship between the PDU's, submitted for transmission, if required.

This protocol has become very famous in the LAN-world, due to its 'simplicity'. However, the Network Layer is so complex that it has a sub-architecture of its own; the international standard IS8648, which is the Internal Organisation of the Network Layer. Many examples of possible internetworking scenarios are explained in this standard.

8.3 The Transport Layer

The purpose of the Transport Layer is to provide an error free end-to-end protocol to its user(s). It optimises use of the available network services to provide at minimum cost the performance required by session entities. The Transport Layer user (the MMTCP, or better: its session layer implementation) can rely on a perfect connection-oriented transport service and, eventually on a connection-less transport service. These services are standardised by ISO, they are described in IS8072.

Dedicated links will be set up for each MMTCP to MMTCP connection and, **for each priority level.** They will be removed by the session layer when they have not been used for a while.

Note that (conform the standard 8072) the Transport Layer may not exceed a certain (predetermined) number of frames, transmitted per time interval. If we want to exceed this limit, the Transport Layer has to open one or more extra communication channels to satisfy its transmission needs.

8.3.1 Transport Protocol Types and Classes

The ISO has developed a family of transport protocol standards tailored to various levels of service and communication facilities (IS8073). They define three network types:

Type A: Network connection with acceptable residual error rate and acceptable rate of signalled failures
Type B: Network connection with acceptable residual error rate but unacceptable rate of signalled failures
Type C: Network connection with residual error rate not acceptable to the transport service user.

In this context, an error is defined as a lost or duplicated Network PDU.

In order to handle a variety of user service requirements and available network services, the ISO has defined five classes of Transport Protocols:

Class 0: Simple
Class 1: Basic error recovery
Class 2: Multiplexing
Class 3: Error recovery and multiplexing
Class 4: Error detection and recovery.

These classes are related to the three types of network services as follows: Class 0 and Class 2 are used with Type A networks; Class 1 and

Class 3 are used with Type B networks; and Class 4 is used with Type C networks.

8.3.2 Implementation of the Transport Layer

It is obvious that the implemented Transport Layer should be of Class 4, because the underlying layers do not handle any error detection and recovery (except a cyclic redundancy check at the MAC layer). The service provided to the MMTCP mostly will be connection-oriented, because this service is error-free which is not the case with a connection-less transport service.

9. USAGE OF THE HARDWARE RESOURCES

The MMTCP bus in fact consists of two buses that eventually share the same hardware on the chip, but can be used as two independent buses. The MMTCP bus consists of a memory access bus and a messaging bus. The use of the memory bus is very obvious and will not be discussed here. The messaging bus will be used as an internal bus that connects all building blocks to each other. The use of this bus is described in [1]. This is done in terms of addressing a register of a building block and writing a word to that block.

In the next sections we will discuss what service we expect from the available resources (timer management unit, host interface, memory (de-)allocation handler, memory interface et al.) and how we want to use this service. We must remark that the service is presented in a very abstract way and thus do not imply <u>any</u> implementation. The proposed transferred information is <u>not</u> definitive yet.

9.1 Communication with other Building Blocks

The information that is transferred and how this information will be transferred is under study. The transfer probably will be done on a single- or double cycle access basis. This means that in the write register of the building block a few bits (of the 16 bits available) will be used to encode the most important functions and the remaining bits will be used to point to a register which can be written in the second write cycle. This method is very efficient if some commands to a building block, that are activated frequently, are coded in the primary register and the other commands, that are less frequent activated, are encoded in the secondary register(s). In the next subsections we will describe the service needed by the different building blocks.

9.1.1 Timer Management Unit (TMU)

We will assume that a TMU is available somewhere in the MMTCP. Otherwise the implementation of a timer has to be done within a building block that needs a timer, which is a trivial case. The timer interface makes use of the messaging bus. The service provided by the TMU will be described with service primitives.

9.1.1.1 TIMER.request

This primitive defines the transfer of data from a protocol entity to the TMU.

TIMER.request (
command,
timer_id,
user_id,
timer_value
)

The command parameter specifies the requested action and shall be one of the following:

- restart (set timer to initial value and start again; this command is called 'reset' in the MAC protocol)
- stop
- time_left (referred to in the Operational F.S.M. of the MAC protocol as TEST_THT)
- new_timer (including this command the time-basis of the timer should be given somehow)
- dispose_timer

The timer_id parameter specifies the timer on which the command is applicable. If the timer_id is unknown it shall have number zero. If the command is applicable to all timers, the timer_id shall be 'all ones'. The user_id parameter specifies which entity requested the service from the TMU. This number can be equal to the register number of the calling protocol entity. The timer value specifies the time-out time.

9.1.1.2 TIMER.confirmation

This primitive defines the appropriate response to the TIMER.request primitive, signifying the success or failure of the request.

TIMER.confirmation (
status,
timer_id,
user_id,
timer_value
)

The status parameter specifies the status information on the requested action and shall be one of the following:

- timer_stopped
- timer_restarted
- time_left
- new_timer
- timer_disposed
- timer_unknown
- invalid_command

The timer_id parameter specifies the timer on which the command is applicable. If the timer_id is unknown it shall have number zero. If the status information is applicable to all timers, the timer_id shall be 'all ones'. The user_id parameter specifies which entity requested the service from the TMU.

This number can be equal to the register number of the calling protocol entity. The timer value specifies the time-out time.

9.1.1.3 TIMER.indication

This primitive defines the transfer of data from the TMU to a protocol entity.

```
TIMER.indication    (
                    command,
                    timer_id,
                    user_id,
                    )
```

The command parameter specifies the indicated action and shall be one of the following:

- timer_expired
- error_in_TMU

The timer_id parameter specifies the timer on which the command is applicable. If the timer_id is unknown it shall have number zero. If the command is applicable to all timers, the timer_id shall be 'all ones'. The user_id parameter specifies which protocol entity is the owner of the timer. This number can be equal to the register number of the calling protocol entity.

9.1.2 Memory Management Unit (MMU)

The MMU manages (as its name already says) the memory available for the building blocks. This is done to share the memory in a economical way between the blocks, since each block sometimes can need a lot of it for a short time. The MMU consists of a memory allocation handler and a memory de-allocation handler, who do not necessarily have to be located within the same block. The memory allocation handler assigns memory blocks (with a size of probably 32 words) to a building block, who needs memory space to store a piece of data or who wants to create a pool of buffer space. The memory de-allocation handler is used to throw away the buffers not needed anymore. The MMU interface makes use of the messaging bus. The service provided will be described using service primitives. Since the allocation handler and the de-allocation handler are assumed not to be on the same location, they do not share the same primitives.

9.1.2.1 MEMORY_ALLOCATE.request

This primitive defines the transfer of data from a building block to the allocation handler of the MMU.

```
MEMORY_ALLOCATE.request(
                                number_of_blocks,
                                use_code,
                                user_id
                                )
```

The number_of_blocks parameter specifies the number of memory blocks the requesting entity needs to operate correctly. The use_code parameter identifies the purpose of the requested entity. The user_id parameter specifies which entity requested the service from the MMU.

9.1.2.2 MEMORY_ALLOCATE.confirmation

This primitive specifies the appropriate response to the request primitive, signifying the success or failure of the request.

```
MEMORY_ALLOCATE.confirmation(
                                status,
                                block_ptr
                                )
```

The status parameter specifies the status information on the requested action and shall be one of the following:

- request_accepted
- request_denied
- invalid_use_code

The block_ptr specifies the start-address of the requested memory block. If more than one block was requested, it specifies the start-address of the first memory block. Within the memory block a pointer to the start-address of the next memory block is provided.

9.1.2.3 MEMORY_DE_ALLOCATE.request

This primitive defines the transfer of data from a building block to the de-allocation handler of the MMU. This primitive shall not be confirmed. The memory blocks are always de-allocated.

```
MEMORY_DE_ALLOCATE.request (
                                block_ptr
                                )
```

The block_ptr specifies the start-address of the memory block that must be de-allocated by the MMU.

9.1.2.4 Ownership of the Memory Blocks

Memory blocks can be passed from one building block to another. An example for this is the following:

a MMTCP generates a message for another MMTCP. The message is passed to the Transport Layer building block, who processes the message. Then the (modified) message is passed to the Network Layer building block, and so on. The MAC Layer building block is in this example the last block that receives the message.

Now the MAC layer has to decide what to do with the memory block. Therefore, each memory block shall contain one byte, indicating the owner of it. If no owner is designated or, the MAC Layer building block is the owner, then the block may be de-allocated. The Transport Layer probably will be the owner of the block, because this layer has to buffer each message until it is confirmed by the peer Transport Layer entity.

9.1.3 MMTCP Building Blocks

These primitives are responsible for the communication between the communication controller and its environment.

9.1.3.1 MMTCP.request

This primitive defines the transfer of a command from a MMTCP building block to a protocol entity. This primitive shall not be confirmed.

```
MMTCP.request (
                building_block_id,
                protocol_entity_id,
                block_ptr
                )
```

The building_block_id parameter designates the building block that is receiving a command from the protocol entity,designated by the protocol_entity_id parameter. The block_ptr parameter specifies the start-address of the block that is passed from the building block to the protocol entity.

9.1.3.2 MMTCP.indication

This primitive defines the transfer of data from a protocol entity to a MMTCP building block. This primitive shall not be confirmed.

```
MMTCP.indication   (
                   source_id,
                   building_block_id,
                   block_ptr
                   )
```

The block_ptr specifies start-address of the block that needs to be passed to the building block, designated with the building_block_id parameter. The source_id parameter designates the transmitting entity.

9.2 Primitive Passing between the Layers of the ISO/OSI Model

The primitives, exchanged between the blocks that implement the layers of the ISO/OSI Model, shall be exactly the same as defined in their standards. Perhaps some slight modifications are necessary, but these will not be discussed now.

The protocol entities shall use the external working memory as a buffer to pass their primitives. How access to the external memory is achieved is discussed in [1]. Because this is done quite well, we have no comments on it.

A straightforward method to process data is the following. Data is read from memory, processed (adding or deleting protocol information to or from the data field), copied into a new buffer and, passed to the next layer. However, we must use the memory in a smart way to save memory access time. Because a few times almost the same information is passed to different building blocks, we can modify the information blocks instead of copying them into a new memory block. How this is done will be discussed in the next subsections.

9.2.1 Processing of Memory Blocks

We require that processing of memory blocks should be easy to implement, since it has to be realised on-chip more than once. The operations that can be performed on PDU's and therefore 'must' be implemented are the following (defined by the OSI model and to be used by the Transport Layer of the communication controller):

- segmentation (splitting one PDU into two or more smaller PDU's)
- reassembling (the reverse of segmenting)
- concatenation (the grouping of two or more PDU's into a larger PDU)
- separation (the reverse of concatenation)

For interprocess communication the last two functions are of interest, because two small PDU's can be delivered more efficient to a peer protocol entity. Only if we want to use the MMTCP as a communications co-processor the first two functions should be implemented, because the Transport Layer may accept large blocks of data (64 kbytes) from the Session Layer. To remain compatible with other systems, a communications co-processor should be able to process large data blocks.

Apart from these operations, the protocol adds information to a higher layer PDU, This means that a header and a trailer must be added to the PDU. When a received PDU from a lower layer is processed, the reverse operation must be performed; the header and the trailer must be removed and, the stripped PDU will be passed to the next higher layer.

The external working memory is organised in such a way, that messages are stored in a list of linked memory blocks with a size of 64 bytes (or better: 32 words). Within these blocks two elements are (pre)defined: one word to designate the next block in the list and one byte to designate the so called 'use_code'. Besides these, one byte is reserved in the memory block. This is done because the writing to a memory block happens on a 'one-word-basis'.

Because these memory blocks are small, transmission of large frames (which will not occur often) will cost a lot of overhead in processing time and memory use, if each memory block contains extra information about its

contents. For smaller frames, the processing time probably increases if each memory block contains extra information. Thus we should think about a way to describe the contents of all the memory blocks (forming the 'main buffer') in one 'primitive description buffer'. Using such a structure, each layer could append its information to the higher layer PDU, just by modifying the description buffer.

Besides the description buffer, another extra buffer is needed; the 'transparent data buffer'. The use of this buffer is the following. Suppose the MMTCP wants to transmit a frame. Besides the information field, the priority and the destination addresses must be specified. For these parameters no space can be reserved in the main buffer, because this information has to be used by (one of) the lower layers. The mentioned parameters (priority and DA) for instance are used by the MAC layer and, therefore are passed transparently through the upper layers. After reception of a frame by the MAC layer the same happens; the received priority and DA are not put into the main reception buffer, but are passed transparently to the MMTCP in the 'transparent data buffer'.

The buffer structure we have introduced is described more formal in the next subsection.

The most important we always have to keep in mind when discussing a buffer implementation: keep the operations on buffers as simple and as fast as possible!

9.2.2 A Suggested Buffer Structure with the Memory Blocks

60 bytes are free to use in each memory block. Because the used memory blocks are very small, we cannot put much extra information in it (e.g. how many useful data is in it, where the data can be found in the buffer, etcetera). Perhaps it is more efficient to fill buffers in a 'standard' way. One memory block gives a complete description[7] of the message,the transparent data, the headers and, the trailers; thus the complete linked list. A possible description of such a buffer is given as a pseudo-PASCAL record.

---

[7]Note that this is only possible because the contents of the PDU's is standardised (besides some options during initialisation) and thus is known a priori.

```
type pointertype     =    ^memory_block;
     arraytype       =    array [1..60] of byte;
     statustype      =    (list of possible events);

memory_block         =    record
                          next_mem_blockpointertype;
                          use_code        : byte;
                          reserved_byte   : byte;
                          datafield       : arraytype
                          end;

description_field    =    record
                          ptr_first_mem_block          : pointertype;
                          offset_to_data_in_blocks     : 1..60;
                          offset_to_first_byte         : 1..60;
                          number_bytes_in_block        : 1..60;
                          number_bytes_in_last_block   : 1..60;
                          field_length                 : integer
                          end;

transparent_data_buffer =    record
                          priority                  : 0..7;
                          reception_status          : statustype;
                          transport_source_address  : 6_bytes;
                          transport_dest_address    : 6_bytes;
                          network_source_address    : byte;
                          network_dest_address      : byte;
                          LLC_SSAP                  : byte;
                          LLC_DSAP                  : byte;
                          MAC_SA                    : 6_bytes;
                          MAC_DA                    : 6_bytes
                          end;

description_buffer   =    record
                          total_length              : integer;
                          MAC_header                : description_field;
                          MAC_trailer               : description_field;
                          LLC_header                : description_field;
                          LLC_trailer               : description_field;
                          Layer_3_header            : description_field;
                          Layer_3_trailer           : description_field;
                          Layer_4_header            : description_field;
                          Layer_4_trailer           : description_field;
                          MMTCP_data                : description_field;
                          transparent_data_ptr : ^transparent_data_buffer
                          end;
```

The used parameters and variables will be explained now (although their use is very obvious).

The pointertype points at a memory block (the size of a pointer is one word). The memory block is described as a record containing the 'use_code', the pointer to the next block in a chain and one reserved byte. The data field is described as an array of 60 bytes.

A description field contains the description of one buffer. The ptr_first_mem_block parameter points at the first memory block of the buffer. The offset_to_data_in_blocks is the index that designates the first byte in every array in every memory block, except the first one. The blocks are assumed to be filled in the same way, but from the first memory block bytes can be deleted by the layers. The parameter offset_to_first_byte therefore designates the first byte of the array of the first memory block. The number_bytes_in_block parameter indicates how many relevant data bytes are stored in each array. The parameter number_bytes_in_last_block indicates the number of bytes in the last memory block, who does not have to be filled completely. The number of bytes in the first memory block can be calculated using the difference between the offset_to_data_in_blocks and the offset_to_first_byte parameters. At last, the field_length parameter shows the total number of bytes in the buffer.

The transparent data buffer (who can be stored easily in one memory block) contains the description of all parameters that can be passed transparently through the buffers. The given list perhaps may not be complete. The parameter names speak for themselves and are therefore not explained.

The description buffer contains a lot of description fields; the original data field that was supplied by the MMTCP or an external process and the headers and trailers supplied by the layers. Besides these, the description buffer contains a pointer to the transparent data buffer. The total_length parameter designates the total length of the current processed frame. This parameter is very important, although it is redundant. It is used within the MAC layer to determine the time that is needed to transmit the frame.

During transmission of a frame, a layer has to:
- prepare a description field for the description buffer, after having filled a buffer
- add the description field to the description buffer
- modify the total_length parameter
- (eventually) modify the transparent data buffer.

After reception of a frame, a layer has to:
- (eventually) modify the transparent data buffer
- modify the MMTCP_data description field (where the data is supposed to be stored by the MAC layer) of the description buffer
- modify the total_length parameter.

9.3 Conclusions and Remarks

The communication between all building blocks that are involved in the communication process has been discussed in this chapter. Because a lot of work still has to be done on the internal buses, this communication is described on a high level of abstraction.

The same holds for the description of the memory usage; by using an abstract notation we do not lose any flexibility. The given buffer description will not be the definitive one, this given description should only serve as food for thought.

During transmission, all layers append a header to the PDU. The trailers, who are only given for the sake of completeness, probably never will be used.

10. CONCLUSIONS AND REMARKS

1 The previous chapters have given an idea of how a communication controller for use in the MMTCP can be realised. Conformance to the ISO/OSI Reference Model was one of the goals of the study. Another goal was to have a system with a quick reaction time, which is most important with a real-time multitasking multiprocessor system. The result of this effort is a MMTCP that may have a communication system that is capable of communicating with many other systems. These other systems may be other Token Ring controllers or, using a (MMTCP as a) bridge, a MAP or a TOP system (using the Token Bus and CSMA/CD technology respectively).

2 The spin-off of this project might be that an extremely intelligent communication controller is developed in hardware (eventually supported with some special multitasking capabilities). As far as we know, a hardware implementation of the lowest four layers of the ISO/OSI mode is never made. It is possible, that the actual integration technology cannot make such a complex design. In the future this might be possible.

3 We should remark that some functions (realised in hardware) might already be present in the entire MMTCP design, so they would be duplicated. Sometimes it is useful to realise such a function as a 'hardware subroutine', to save chip space. An example of this is the Timer Management Unit. Another example is the address checking of a frame. This is done inside the MAC layer, inside the Host Command Interpreter after reception of the frame by the bridge process (to check whether or not the frame has to be forwarded to another MMTCP) and probably within other blocks too.

4 A possible use of the MMTCP's communication controller we did not investigate is the following. When we put two or more MMTCP's on one printed circuit board, we might couple these together without a difficult (external) physical layer implementation (no clock recovery, no line drivers).

5 The Token Ring controller can be specified in a formal way now; the verbal specification of the MAC layer and its hardware subroutines is complete. The LLC and higher layers need a further study before these can be specified in a more formal way. This specification for example can be done in a language called HHDL (High-level Hardware Description Language).

REFERENCES

[1] A.C. Verschueren, A Coprocessor for Hardware Multitasking Support, Master's Thesis, Eindhoven University of Technology, Department of Electrical Engineering, August 1987.

[2] B.H. Liebowitz, J.H. Carson, Multiple Processor Systems for Real-Time Applications, Prentice Hall, New Jersey, 1985.

[3] A.S. Tanenbaum, Computer Networks, Prentice/Hall International Editions, Englewood Cliffs, New Jersey, 1981.

[4] IEEE project 802 Local Area Network Standards, IEEE Computer Society:

[4.1] 802.1-draft : Internetworking and Network Management
[4.2] 802.2-1985 : Logical Link Control
[4.3] 802.3-1985 : CSMA/CD Medium Access Method
[4.4] 802.4-1985 : Token Bus Medium Access Method
[4.5] 802.5-1985 : Token Ring Medium Access Method

[5] B.W. Stuck, Calculating the Maximum Mean Data Rate in Local Area Networks, IEEE Computer Magazine, May 1983, pp. 72-76.

[6] D.W. Andrews, G.D. Schultz, A Token-Ring Architecture for Local-Area Networks: an Update, Proceedings of the IEEE COMPCON Fall 82 Conference, pp 615-624, 1982.

[7] Texas Instruments, TMS380 Adapter Chipset User's Guide, 1986.

[8] R.C. Dixon, N.C. Strole, J.D. Markov, A Token-Ring Network for Local Data Communications, IBM Systems Journal, Vol. 22, Nos 1/2, pp. 47-62, 1983.

[9] T. Jeffree, Management of Local Area Networks: IEEE 802.1 systems management, Proceedings of the International Conference on OPEN SYSTEMS '86, London, march 1986, pp. 183-195, Online Publications, New York.

[10] J.A. Bernsten, J.R. Davin, D.A. Pitt, N.G. Sullivan, MAC layer interconnection of IEEE 802 local area networks, Computer Networks and ISDN Systems 10, 1985, pp. 259-273.

APPENDIX A. CALCULATION OF THE TIMER VALUES

The calculation below is based upon the standardised data rate of 4 Mbit per second. As soon as this rate increases, the timer values shall be decremented.

The real-time clock we have in mind is based upon a crystal oscillator, originating from digital watches. This oscillator generates a 32 kHz signal. Thus the smallest time unit will be 31 microseconds. The maximum time unit will be 2 seconds (because the counters will be 16 bits wide).

A.1 Default Timer Values

In the following we assume that the default address length equals to 2 bytes; the values are not too critical so these can be used for a 6 byte address length too.

The Token Holding Timer shall have a time-out time, large enough to transmit small messages between stations (64 bytes). Since the envelope of a frame is 17 bytes in length, the total length of a frame is 616 bytes. The time needed to transmit this frame is 150 microseconds (5 tics of the timer).

The Return to Repeat Timer (TRR) depends on the length of the ring (the number of inserted stations). Suppose we have a delay per station of 2 bits, which is a realistic assumption. The total ring delay then is 542 bits; 256 stations and a 'latency buffer' of 30 bits that is provided by the active monitor. This makes the time of the ring delay 140 microseconds (5 tics of the timer).

The Queue PDU Timer (TQP) is not critical and shall have a time-out value of 150 microseconds (equal to the THT).

The Valid Transmission Timer (TVX) will have a time-out value equal to the time-out value of THT plus the time-out value of TRR. The THT value that will be used here is the longest THT on the ring, which will be about 600 microseconds (a frame with 256 information-bytes). Thus the TVX time-out time will be 750 microseconds.

The Timer No Token received (TNT) will have a time-out time equal to TRR plus 256 times (again the longest) THT value. The time-out time of TNT thus will be 150 milliseconds.

The default value of the Standby Monitor Timer (TSM) shall be the maximum time-out time: 2 seconds. This time is as high as possible to prevent that the station, when it inserts itself into the ring, immediately enters a claiming token state.

The Active Monitor Timer shall have a time-out time of 750 milliseconds. This allows an AMP frame to get lost (twice) before the TSM expires. This results in the following default timer parameters:

TRR	150 microsec.
THT	150 microsec.
TQP	150 microsec.
TVX	750 microsec.
TNT	150 millisec.
TAM	750 millisec.
TSM	2 sec.

APPENDIX B. PARTS OF THE STANDARD 802.5

B.1 Standardized Token Ring Protocols

4. Token Ring Protocols

This section specifies the procedures that shall be used in the medium access control (MAC) sublayer.

4.1 Overview. The subsections of 4.1 provide a descriptive overview of frame transmission and reception. The formal specification of the operation is given in 4.2.

4.1.1 Frame Transmission. Access to the physical medium (the ring) is controlled by passing a token around the ring. The token gives the downstream (receiving) station (relative to the station passing the token) the opportunity to transmit a frame or a sequence of frames. Upon request for transmission of an LLC PDU or NMT PDU, MAC prefixes the PDU with the appropriate FC, DA, and SA fields and enqueues it to await the reception of a token that may be used for transmission.

Such a token has a priority less than or equal to the priority of the PDU(s) that is to be sent. Upon queuing the PDU for transmission and prior to receiving a usable token, if a frame or an unusable token is repeated on the ring, the station requests a token of appropriate priority in the RRR bits of the repeated AC field. Upon receipt of a usable token, it is changed to a start-of-frame sequence by setting the token bit.

At this time, the station stops repeating the incoming signal and begins transmitting a frame. During transmission, the FCS for the frame is accumulated and appended to the end of the information field.

4.1.2 Token Transmission. After transmission of the frame(s) has been completed, the station checks to see if the station's address has returned in the SA field, as indicated by the MA_FLAG. If it has not been seen, the station transmits fill until the MA_FLAG is set, at which time the station transmits a token.

4.1.3 Stripping. After transmission of the token, the station will remain in transmit state until all of the frames that the station originated are removed from the ring. This is done to avoid unnecessary recovery action that would be caused if a frame were allowed to continuously circulate on the ring.

4.1.4 Frame Reception. Stations, while repeating the incoming signal stream, check it for frames they should copy or act upon. If the frame-type bits indicate a MAC frame, the control bits are interpreted by all stations on the ring. In addition, if the frame's DA field matches the station's individual address, relevant group address, or broadcast address, the FC, DA, SA, INFO, and FS fields are copied into a receive buffer and subsequently forwarded to the appropriate sublayer.

4.1.5 Priority Operation. The priority bits (PPP) and the reservation bits (RRR) contained in the access control (AC) field work together in an attempt to match the service priority of the ring to the highest priority PDU that is ready for transmission on the ring. As previously noted in 3.6, these values are stored in registers as Pr and Rr. The current ring service priority is indicated by the priority bits in the AC field, which is circulated on the ring.

The priority mechanism operates in such a way that *fairness* (equal access to the ring) is maintained for all stations within a priority level. This is accomplished by having the same station that raised the service priority level of the ring (the *stacking station)* return the ring to the original service priority. As previously noted in 3.6, the Sx and Sr stacks are used to perform this function.

The priority operation is explained as follows: When a station has a priority (a value greater than zero) PDU (or PDU's) ready to transmit, it requests a priority token. This is done by changing the reservation bits (RRR) as the station repeats the AC field. If the priority level (Pm) of the PDU that is ready for transmission is greater than the RRR bits, the station increases the value of RRR field to the value Pm. If the value of the RRR bits is equal to or greater than Pm, the reservation bits (RRR) are repeated unchanged.

After a station has claimed the token, the station transmits PDUs that are at or above the present ring service priority level until it has completed transmission of those PDUs or until the transmission of another frame could not be completed before timer THT expires (see 3.4.2). The priority of all of the PDUs that are transmitted should be at the present ring service priority value. The station will then generate a new token for transmission on the ring.

If the station does not have additional PDUs to transmit that have a priority (Pm) or does not have a reservation request (as contained in register Rr) neither of which is greater than the present ring service priority (as contained in register Pr), the token is transmitted with its priority at the present ring service priority and the reservation bits (RRR) at the greater of Rr or Pm and no further action taken.

However, if the station has a PDU ready for transmission or a reservation request (Rr), either of which is greater than the present ring service priority, the token is generated with its priority at the greater of Pm or Rr and its reservation bits (RRR) as 0. Since the station has raised the service priority level of the ring, the station becomes a stacking station and, as such, stores the value of the old ring service priority as Sr and the new ring service priority as Sx. (These values will be used later to lower the service priority of the ring when there are no PDU's ready to transmit on the ring whose Pm is equal to or greater than the stacked Sx.)

NOTE: Since a station may have raised the service priority of the ring more than once before the service priority is returned to a lower priority (for example, from 1 to 3 and then 5 to 6), it may have multiple Sx and Sr values stored and, hence, the term *stacked*. Also note that the terms *stack* and *stacked* are not to be confused with other usages of these same terms.

Having become a stacking station, the station claims every token that it receives that has a priority (PPP) equal to its highest stacked transmitted priority (Sx) in order to examine the RRR bits of the AC field for the purpose of raising, maintaining, or lowering the service priority of the ring. The new token is transmitted with its PPP bits equal to the value of the reservation bits (RRR) but no lower than the value of the highest stacked received priority (Sr), which was the original ring priority service level.

If the value of the new ring service priority (PPP equal to Rr) is greater than Sr, the RRR bits are transmitted as 0, the old ring service priority contained in Sx is replaced with a new value Sx equal to Rr, and the station continues its role as a stacking station.

However, if the Rr value is equal to or less than the value of the highest stacked received priority (Sr) the new token is transmitted at a priority value of the Sr, both Sx and Sr are removed *(popped)* from the stack, and if no other values of Sx and Sr are stacked, the station discontinues its role as a stacking station.

NOTE: A stacking station that has claimed the token may transmit PDUs as well as examining RRR bits, as described above. Of course only those PDUs which have a priority equal to or greater than the ring service priority may be transmitted.

The frames that are transmitted to initialize the ring have a PPP field that is equal to 0. The receipt of a PPP field whose value is less than a stacked Sx will cause any Sx or Sr values that may be stacked to be cleared in all stations on the ring.

The complete description of priority operating is contained in the Operational Finite-State Machine (see Fig 4-3).

4.1.6 Beaconing and Neighbor Notification. When a hard failure is detected in a token ring, its cause must be isolated to the proper failure domain so that recovery actions can take place. The failure domain consists of

(1) the station reporting the failure (the beaconing station)

(2) the station upstream of the beaconing station

(3) the ring medium between them

For example, if a failure occurred within the domain shown in Fig 4-1, station G would report upon it by transmitting beacon MAC frames.

A failure that causes bit disruption within the transmitter side of station F, in the medium between stations F and G, or within the receiver side of station G, will be detected and reported upon by station G using a beacon MAC frame. This alerts all other stations on the ring that the token protocol has been suspended until such a time that the disruption terminates or is removed.

To do accurate problem determination, all elements of the failure domain must be known at the time that the failure is detected. This implies that at any given time, each station should know the identity of its upstream neigh-

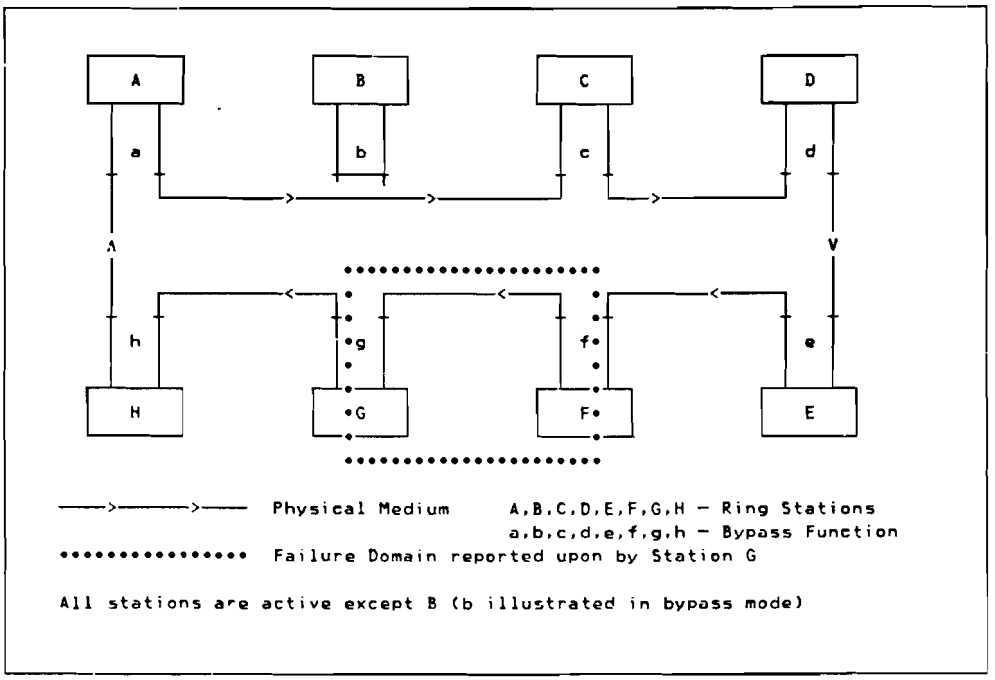


Fig 4-1
An Example of a Failure Domain

bor station. A process for obtaining this identity, known as Neighbor Notification, is described below.

Neighbor Notification has its basis in the address-recognized and frame-copied bits (the A and C bits) of the FS field. These bits are transmitted as 0's. If a station recognizes the destination address of the frame as one of its own, the station sets the A bits to 1 in the passing frame. If a station also copies the frame, then the C bits are also set to a 1.

When a frame is broadcast to all stations on a ring, the first station downstream of the broadcaster will see that the A and C bits are all 0's. Since a broadcast frame will have its destination address recognized by all of the stations on the ring, the first station downstream will, in particular, set the A bits to 1. All stations further downstream will, therefore, not see the A and C bits as all 0's. This process continues in a circular, daisy-chained fashion to let every station know the identity of its upstream neighbor (see the note under 3.3.4).

The monitor begins Neighbor Notification by broadcasting the active monitor present (AMP) MAC frame. The station immediately downstream from it takes the following actions:

(1) resets its timer TSM, based on seeing the AMP value in the FC field;

(2) if possible, copies the broadcast AMP MAC frame and stores the upstream station's identity in an upstream neighbor's address (UNA) memory location;

(3) sets the A bits (and C bits if the frame was copied) of the passing frame to 1's;

(4) at a suitable transmit opportunity, broadcasts a similar standby monitor present (SMP) MAC frame.

One by one, each station receives an SMP frame with the A and C bits set to 0's, stores its UNA, and continues the process by broadcasting such a frame itself.

Since the AMP frame must pass each station on a regular basis (the active monitor present MAC frame sent by the monitor), the continuous transmission of tokens onto a ring can be detected. In addition to the timer TAM in the active monitor, each standby station has a timer TSM that is reset each time an AMP MAC frame passes. If timer TSM expires, that standby monitor station begins transmitting claim token frames.

4.2 Specification. The operation of the ring is described in this section.

In the case of a discrepancy between the FSM diagrams/tables and the supporting text, the FSM diagrams/tables shall take precedence.

The MAC receives from the PHY layer a serial stream of symbols. Each symbol shall be one of the following:

0 = binary zero
1 = binary one
J = non-data-J
K = non-data-K

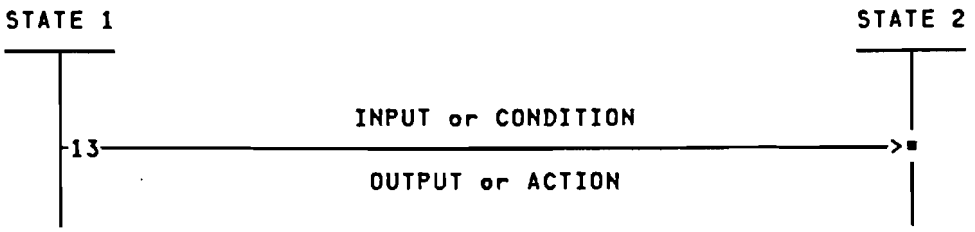
(See 6.1 for a detailed description of these symbols.)

From the received symbols MAC detects various types of input data, such as tokens, MAC frames, and LLC information frames.

In turn, MAC stores values, sets flags, and performs certain internal actions (as noted in Fig 4-2, Receive Action Table) as well as generating tokens, frames, or fill, or flipping bits and delivering them to the PHY layer in the form of a serial stream of the 0, 1, J, and K symbols.

For the purpose of accumulating the FCS and storing the contents of a frame, J and K symbols that are not part of the SD or ED shall be interpreted as 1 and 0 bits, respectively.

Finite-State Machine (FSM) Notation. The notation used in the FSM diagrams is as follows:



States are shown as vertical lines. Transitions are shown as horizontal lines with a number indicating the transition (for example, 13) and the arrow indicating the direction of transition.

The input or condition shown above the line is the requirement to make the transition. The output or action shown below the line occurs simultaneously with making the transition. The transition begins when the input occurs or the condition specified is met and is complete when the output or action has occurred. If the state transition is in progress, then no other FSM transition may be initiated.

If the exit conditions of a state are satisfied at the time the state is entered, no action is taken in that state and the state is immediately exited.

Abbreviations and Mnemonics (as used in FSM description)

A = Address-Recognized Bit
AMP = Active Monitor Present
BCN = Beacon
C = Frame-Copied Bit
CL = Claim
DA = Destination Address
DAT = Duplicate Address Test
E = Error Detected Bit
ED = Ending Delimiter
EFS = End-of-Frame Sequence
FR = Frame
FS = Frame Status (Field)
I = Intermediate Frame Bit
M = Monitor Bit
A = My (station's) Address
MSI = MA_STATUS.indication
NMT = Network Management
P = Priority (of the AC)
PDU = Protocol Data Unit
Pm = PDU Priority
Pr = Last Priority Value Received

PRG = Purge
R = Reservation (of the AC)
RR = Last Reservation Value Received
RUA = Received Upstream Neighbor's Address
SUA = Stored Upstream Neighbor's Address
SA = Source Address
SFS = Start-of-Frame Sequence
SMP = Standby Monitor Present
Sr = Highest Stacked Received Priority
Sx = Highest Stacked Transmitted Priority
TAM = Timer, Active Monitor
THT = Timer, Holding Token
TK = Token
TNT = Timer, No Token
TQP = Timer, Queue PDU
TRR = Timer, Return to Repeat
TSM = Timer, Standby Monitor
TVX = Timer, Valid Transmission
TX = Transmit

¬ = Boolean NOT
& = AND
V = OR
/ = the greater of

TK(P=x,M=y,R=z) = Token with P=x, M=y, and R=z
FR(P=x,M=y,R=z) = Frame with P=x, M=y, and R=z

4.2.1 Receive Actions. Three varieties of frame identification are used in the state transitions and at the service interfaces described in this standard: *good frame*, *validly formed frame*, and *frame with error*. These frame varieties are indicated by combinations of the following properties:

Properties of a Frame

(1) Is bounded by a valid SD and ED
(2) Has the E (error) bit equal to 0
(3) Is an integral number of octets in length
(4) Is composed of only 0 and 1 bits between the SD and ED
(5) Has the FF bits of the FC field equal to 00 or 01
(6) Has a valid FCS
(7) Has a minimum of 10 (2-octet addressing) or 18 (6-octet addressing) octets between SD and ED

The three frame varieties are defined below. This is not an inclusive list of all possible bit-sequence formats; for example, other format sequences known in this standard are the token and the abort sequence. Note that the value of the I, E, A, and C bits are not part of these definitions.

Good Frame (FR_GOOD). A bit sequence that satisfies the following condition, based on the properties of a frame listed above:

1 & 3 & 4 & 5 & 6 & 7

Validly Formed Frame. A bit sequence that satisfies the following condition:

1 & 3 & 5 & 7

Frame With Error (FR_WITH_ERROR). A bit sequence that satisfies the following condition:

1 & (¬3 V ¬4 V (5&¬6) V (5&¬7))

The various internal actions that are taken as a result of an input received from the ring are summarized in the Receive Action Table (Fig 4-2). They are explained as follows:

(R-A) Report Frame Condition. The reporting actions for received frames are dependent upon the properties of a frame. Whenever one of the following report conditions is satisfied, MA_STATUS is indicated to NMT:

(1) 1 & 2 & 3 & 4 & 5 & 6 & 7
(2) 1 & ¬2 & 3 & 4 & 5 & 6 & 7
(3) 1 & 2 & (¬3 V ¬4 V (5 & ¬6) V (5 & ¬7))

(R-B) Priority Level Error. If there is a highest stacked transmitted priority (Sx) stored and a token is received with a priority (P) less than the value of Sx, then an error has occurred. Therefore, the stacks shall be cleared.

(R-C) My Address Received. If the source address that is received is equal to the station's individual address, the MA flag shall be set. Note that the MA flag shall be set without regard to whether it is a good frame, a validly formed frame, or a frame with error.

(R-D) Access Control Field Received. Upon the receipt of an access control (AC) field in a token or a frame, the value of the priority bits shall be stored as Pr, the reservation bits shall be stored as Rr, and the previously stored Pr and Rr shall be discarded.

(R-E) I Bit Equal Zero Received. If an end-of-frame sequence with I=0 is received the I_FLAG shall be set.

Fig 4-2
Receive Action Table

REF	RECEIVE	ACTION
R-A	REPORT FRAME CONDITION	MSI
R-B	TK(P<Sx)	CLEAR STACKS
R-C	SA=MA	SET MA_FLAG
R-D	TOKEN V FRAME	STORE (Pr, Rr)
R-E	I=0	SET I_FLAG
R-F	SFS	SET SFS_FLAG
R-G	FR_(SA=MA,RUA≠SUA)	MSI

(R-F) Start-of-Frame Sequence Received. If a start-of-frame sequence is received the SFS_FLAG shall be set.

(R-G) SA = MA and RUA and SUA Not Equal. If a MAC frame is received in which the SA equals the station's address and it contains an RUA (that is, BCN, CL_TK, AMP, SMP, or PRG frame) not equal to the SUA, MA_STATUS is indicated to MNT.

4.2.2 Operational Finite-State Machine. The operational finite-state machine (see Figs 4-3 and 4-4) is explained as follows:

4.2.2.1 Resume (Operational FSM Activity). When the station is in monitor states of Bypass, Inserted, Transmit Claim Token, Transmit Beacon, Transmit Fill, or Transmit Purge (for example, not in Initialize, Standby, or Active states), activity of the Operational FSM is suspended. Upon reentry into Initialize, Standby, or Active Monitor states, activity of the operational FSM shall be resumed in Repeat state.

4.2.2.2 State 0: REPEAT (Repeat State). In Repeat state, the bits that are received are, in general, repeated on the line to the next station. Certain bits and fields in the repeated bit stream may be modified and certain actions taken without changing state. Transition shall be made to State 1: TX DATA_FR (Transmit Data Frame[s]) when there are one or more PDUs queued for transmission and the conditions for transmission are satisfied. Transition shall be made to State 4: TX ZEROS & MOD STACKS (Transmit Zeros and Modify Stacks) for the purpose of modifying the priority stacks.

(01) Usable Token Received. If a PDU is queued for transmission and a token is received whose priority (P) is equal to or less than the PDU priority (Pm), the station shall change the token to a start-of-frame sequence (by changing the token bit from 0 to 1) and transmit M and R as 0, initiate the transmission of the enqueued PDU, reset the THT and the MA flag, and make a transition to State 1.

(02) Bit Flipping Loop. A number of actions may be taken without changing state. These actions are shown in Fig 4-4 and are explained as follows:

(02A) Request Usable Token. If there is a PDU queued for transmission with priority Pm, the reservation (R) shall be set to Pm on frames in which the reservation is less than Pm, and on tokens in which the priority is greater than Pm and the reservation is less than Pm and the priority is not equal to the highest stacked transmitted priority.

(02B) Frame With Error. The E (error) bit shall be transmitted as 1 if a frame with error is detected. (See Reference R-B in 4.2.1.)

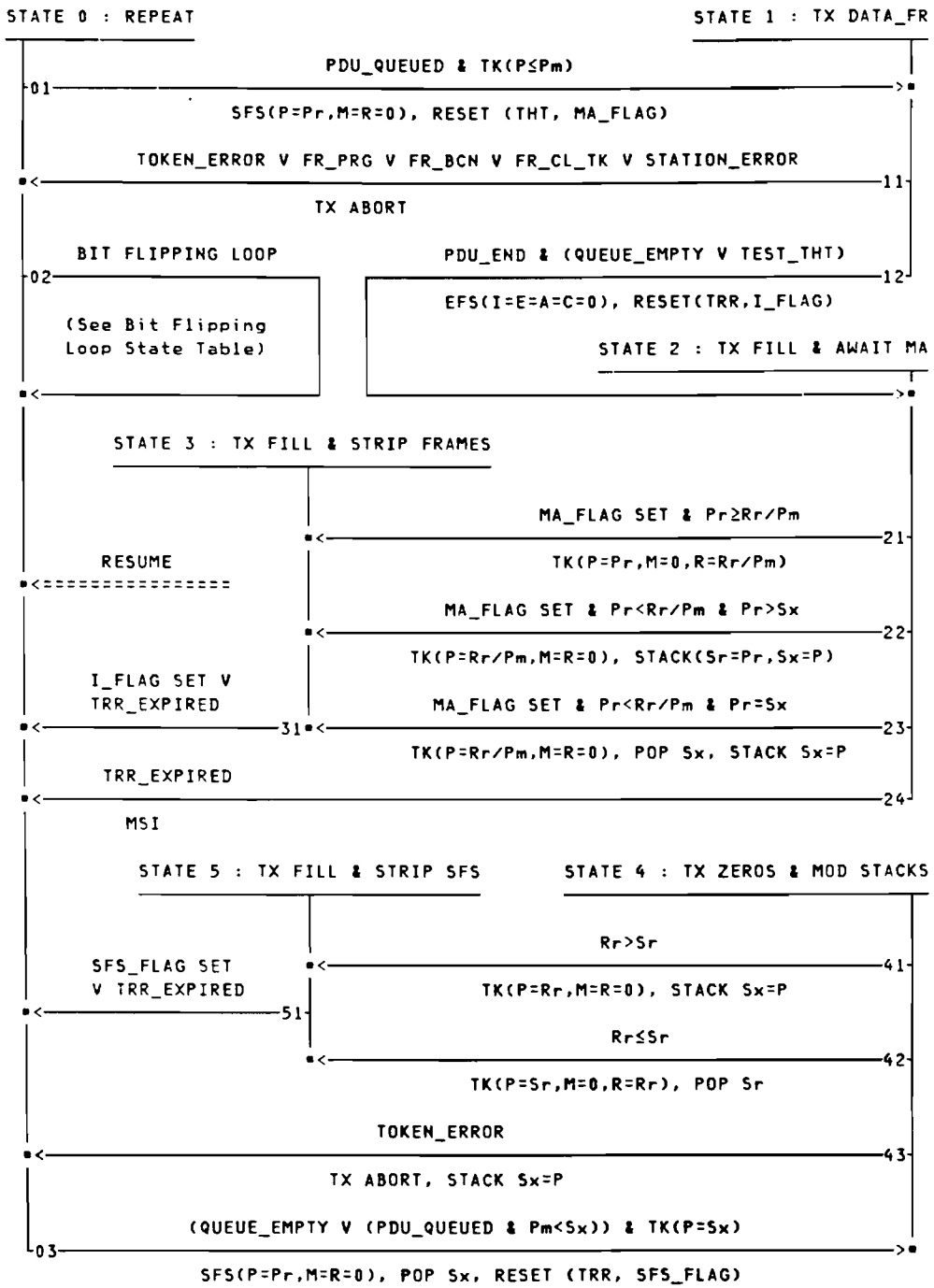


Fig 4-3
Operational Finite-State Machine Diagram

REF	INPUT	OUTPUT
02A	PDU_QUEUED &(FR(R<Pm) V TK(P>Pm>R,P≠Sx)	SET R=Pm
02B	FR_WITH_ERROR	SET E=1
02C	DA=MA (ADDRESS RECOGNIZED)	SET A=1
02D	FR_COPIED	SET C=1

Fig 4-4
Bit Flipping Loop State Table

(02C) Own Address Detected. If the station detected its own address or relevant group address in the DA field, the A bits in the FS field shall be transmitted as 1.

(02D) Frame Copied. If the station copies the frame from the ring, the C bits in the FS field shall be transmitted as 1.

(03) Re-stack Operation. If there are no frames enqueued with priority (Pm) equal to or greater than the highest stacked transmitted priority (Sx) and a token is received with priority (P) equal to the highest stacked transmitted priority (Sx), the following actions are taken. The token shall be changed to a start-of-frame sequence by changing the T bit from 0 to 1, popping the Sx from the stack, resetting timer TRR and the SFS flag, and making the transition to State 4. If there is no Sx value stacked, the test P=Sx shall be considered to be false.

4.2.2.3 State 1: TX DATA_FR (Transmit Data Frame[s]). While in this state, the station transmits one or more frames. The first and all subsequent PDU's that are transmitted shall have a Pm equal to or greater than the priority of the token that was used. All frames transmitted will have P equal to Pr and M and R equal to 0. On the receive side, as noted in Fig 4-2, the station shall monitor the receive data for the value of the priority and reservation bits, its station address, which has been transmitted in the source address field, and the ending delimiter.

(11) Abort State 1: Error Recovery Action. If after changing the token bit from a 0 to a 1, the station detects that the token did not end with an ED; or if a beacon, purge, or claim token frame is subsequently received; or if an error has occurred within the station, the transmission shall be terminated immediately with an abort sequence, the PDU dequeued, LLC notified of the event, and transition made to State 0.

(12) End-of-Frame Transmission. If the transmission of the PDU is completed (PDU_END) and there are no more PDU's to transmit at this priority or a higher priority (QUEUE_EMPTY), or if transmission of an additional frame could not be completed before THT expires (TEST_THT), an end-of-

frame sequence (EFS) shall be transmitted with the I, E, A, and C bits equal to 0; timer TRR and the I flag shall be reset; and transition shall be made to State 2.

4.2.2.4 State 2: TX FILL & AWAIT MA (Transmit Fill and Await My Address). If a source address equal to the station's address has not been received (that is, MA_FLAG reset) the station shall transmit fill until MA_FLAG is set or TRR expires. If upon entering State 2, MA_FLAG is already set, transition shall be made directly to State 3 via transitions 21, 22, or 23.

(21) Token Transmission, Same Priority. If both the stored value Rr and a queued PDU priority (Pm) are less than or equal to the stored value Pr, a token shall be transmitted with the P equal to Pr, M equal to 0, and R equal to the greater of Rr or Pm, and transition shall be made to State 3.

(22) Token Transmission, Higher Priority, and Pr > Sx (Push Ring Priority). If the Rr or an enqueued PDU priority (Pm) is greater than the Pr, and the highest stacked transmitted priority (Sx) is less than the last priority value received (Pr), a token shall be transmitted with the P equal to the greater of Rr or Pm, and M and R equal to 0. Pr shall be stacked as Sr, P shall be stacked as Sx, and a transition made to State 3. If there is no Sx value stacked, the test Pr>Sx shall be considered true.

(23) Token Transmission, Higher Priority, and Pr = Sx (Pop Ring Priority). If the Rr or an enqueued PDU priority (Pm) is greater than the Pr, and the highest stacked transmitted priority (Sx) is equal to the last priority value received (Pr), a token shall be transmitted with the P equal to the greater of Rr or Pm, and M and R equal to 0. Sx shall be popped from the stack and a new value P shall be stacked as Sx and transition made to State 3. If there is no Sx value stacked, the test Pr=Sx shall be considered false.

(24) TRR Expires. If, while waiting for the MA flag to be set, timer TRR expires, transition shall be made directly to Repeat state (State 0) and MA_STATUS indicated to NMT.

4.2.2.5 State 3: TX FILL & STRIP FRAMES (Transmit Fill and Strip Frames). If an EFS with I equal to 0 has not been received (that is, I_FLAG reset) the station shall transmit fill until the I_FLAG is set or TRR expires. If upon entering State 3 the I_FLAG is already set or TRR has already expired, transition shall be made directly to State 0.

(31) Strip Complete. In this state, fill shall be transmitted until an EFS with I equal to 0 is received or TRR expires whereupon transition shall be made to State 0.

4.2.2.6 State 4: TX ZEROS & MOD STACK (Transmit Zeros and Modify Stack). A continuous string of 0's shall be transmitted immediately follow-

ing the SFS until the internal logic of the station can perform the necessary functions to transmit a token.

Transmission of 0's may or may not terminate on an octet boundary. Note that this state shall cause consecutive SD's to exist on the ring without an intervening ED and that the SD of the transmitted token may not occur on an octet boundary relative to the transmitted 0's.

(41) Reservation Request (Rr) > Highest Stacked Received Priority (Sr). If Rr is greater than the highest stacked received priority Sr, a token with its priority (P) set to Rr and its M and R bits set to 0 shall be transmitted, P shall be stacked as Sx, and a transition shall be made to State 5.

(42) Reservation Request (Rr) ≤ Highest Stacked Received Priority (Sr). If Rr is equal to or less than the Sr, then a token with P equal to Sr, M equal to 0, and R equal to Rr shall be transmitted, Sr popped from the stack, and transition shall be made to State 5.

(43) Token Recognition Error. If after changing a token to a SFS, the station detects that the token did not end properly (with MRRR, JK1JK1), the transmission shall be terminated immediately with an abort sequence, Pr stacked as Sx, and transition shall be made to State 0.

4.2.2.7 State 5: TX FILL & STRIP SFS (Transmit Fill and Strip SFS). In this state, fill shall be transmitted until the transmitted SFS is received or TRR expires.

(51) Strip Complete. Upon receipt of the SFS or TRR expiring, transition shall be made to State 0.

4.2.3 Standby Monitor Finite-State Machine. (See Fig 4-5.) Upon coming on-line or after the station has been reset, (re)initialization is performed to assure that no other station on the ring has the same address as this station and that its (re)entry into the ring is known to its immediate downstream neighbor.

Upon completion of initialization, transition is made to Standby state where the ring is monitored to assure that there is a properly operating active monitor on the ring. It does so by observing the tokens and AMP frames as they are repeated on the ring. If tokens and AMP frames are not periodically detected, the standby monitor shall time-out and initiate claiming token.

The standby monitor utilizes timers TNT and TSM in its operation. When in Transmit Claim Token and Transmit Beacon states (States 3 and 5), the station shall utilize its own oscillator for transmission timing.

The standby monitor function is explained as follows:

4.2.3.1 Master Reset. If the station is reset, transition will be made from the current state of the monitor to Standby Monitor Bypass state (State 0). The latency buffer, if in use, will be deleted and all timers will be reset.

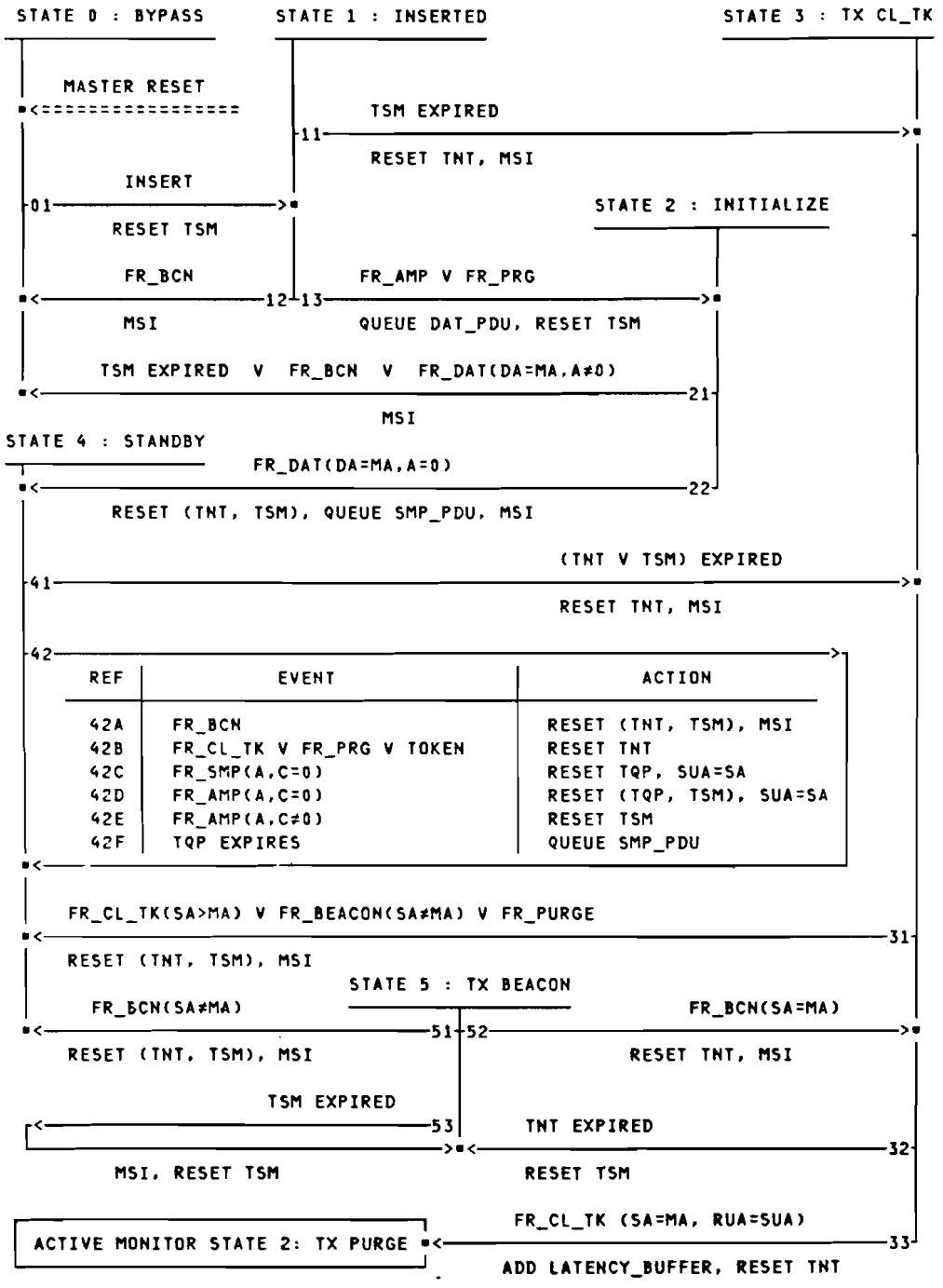


Fig 4-5
Standby Monitor Finite-State Machine Diagram

4.2.3.2 State 0: BYPASS. In this state the station is not inserted in the ring.

(01). Upon activation of the insertion logic (see 5.3.2.3), timer TSM is reset and transition made to State 1.

4.2.3.3 State 1: INSERTED. In this state the station synchronizes its receive clock with the receive signal and then, having achieved synchronization, repeats the received symbols on the line and awaits the receipt of an AMP or PRG.

(11). If an AMP or PRG is not received before timer TSM expires, it is assumed that there is no active monitor in the ring, timer TNT is reset, MA_STATUS is indicated, and transition is made to the Claiming Token state (State 3).

(12). If an FR_BCN is received, the station shall return to Bypass state (State 0) and the MA_STATUS shall be indicated.

(13). However, if AMP or PRG has been received, a Duplicate Address Test (DAT) PDU is enqueued for transmission awaiting the receipt of a usable token, timer TSM is reset, and transition made to Initialize state (State 2).

4.2.3.4 State 2: INITIALIZE. This state exists to detect the existence of a duplicate station address on the ring. This enhances the validity of later checks within the FSMs for SA=MA, etc. This is particularly useful in environments in which the station address assignments are not rigidly controlled. While in this state the station transmits the queued DAT_PDU when a usable token is received and repeats the received symbols on the line until one of the following events occur.

(21). If the DAT MAC frame that was transmitted by the station is not received before timer TSM has expired, or a beacon MAC frame is received, or a DAT MAC frame which the station originated (DA=MA) is received with the Address Recognized bits not set to 0, (A≠0) MA_STATUS is indicated to the NMT and the station returned to a Bypass state (State 0).

NOTE: NMT may determine if the station should retry insertion into the ring.

(22). However, if the DAT MAC frame is returned indicating that there is not another station on the ring with the same address (A=0), an SMP PDU is enqueued for transmission awaiting the receipt of a usable token, timers TNT and TSM are reset, MA_STATUS is indicated to NMT, and transition is made to Standby state (State 4).

4.2.3.5 State 3 : TX CLAIM_TOKEN (Transmit Claim Token). In this state, claim token MAC frames are continuously transmitted. If the SUA value is unknown, a null (all zeros) address will be used as the SUA.

(31). If a Claim Token MAC frame is received in which the source address is greater than the station's address, or a beacon frame is received in which the source address does not equal the station's address, or a purge frame is received, timers TNT and TSM are reset, MA_STATUS is indicated to NMT, and transition is made to Standby state (State 4).

(32). However, if timer TNT expires, timer TSM is reset, and transition is made to Beaconing state (State 5).

(33). Or, if the station receives a FR_CL_TK with a source address equal to the station's address and an RUA equal to the SUA, the bid for active monitor has been won. The latency buffer shall be inserted in the ring, timer TNT reset, and transition made to ACTIVE MONITOR Purge state (State 2).

4.2.3.6 State 4 : STANDBY. In this state the monitor is in standby mode, monitoring the ring to ascertain that there is a properly operating active monitor on the ring. It does so by observing the tokens and AMP frames as they are repeated on the ring. If tokens and AMP frames are not periodically detected, the standby monitor will time-out and initiate claiming token.

(41). If timers TNT or TSM expire, timer TNT is reset and transition made to Claiming Token state (State 3).

(42A). If a beacon frame is received, timers TNT and TSM are reset and MA_STATUS is indicated to NMT without changing state.

(42B). If a claim token frame, a purge frame, or a token is received, timer TNT is reset without changing state.

(42C). If an FR_SMP whose A and C bits equal 0 is received, the SA of the SMP frame shall be stored as the SUA, and timer TQP shall be reset.

(42D). If an FR_AMP whose A and C bits equal 0 is received, the SA of the AMP frame shall be stored as the SUA, and timers TQP and TSM shall be reset.

(42E). If an FR_AMP whose A and C bits do not equal 0 is received, timer TSM shall be reset.

(42F). If timer TQP expires, an SMP PDU shall be enqueued for transmission.

4.2.3.7 State 5 : TX BCN (Transmit Beacon). This state is entered when a serious ring failure has occurred. MAC supervisory beacon frames will continue to be transmitted until beacon MAC frames are received at which time:

(51). If SA does not equal MA, timers TNT and TSM shall be reset, MA_STATUS is indicated to NMT, and transition made to Standby state (State 4).

(52). However, if SA does equal MA then transition shall be made to Claiming Token state (State 3) after resetting timer TNT and indicating MA_STATUS.

(53). If, while transmitting FR_BCN, timer TSM expires, MA_STATUS will be indicated to NMT of the event and timer TSM reset.

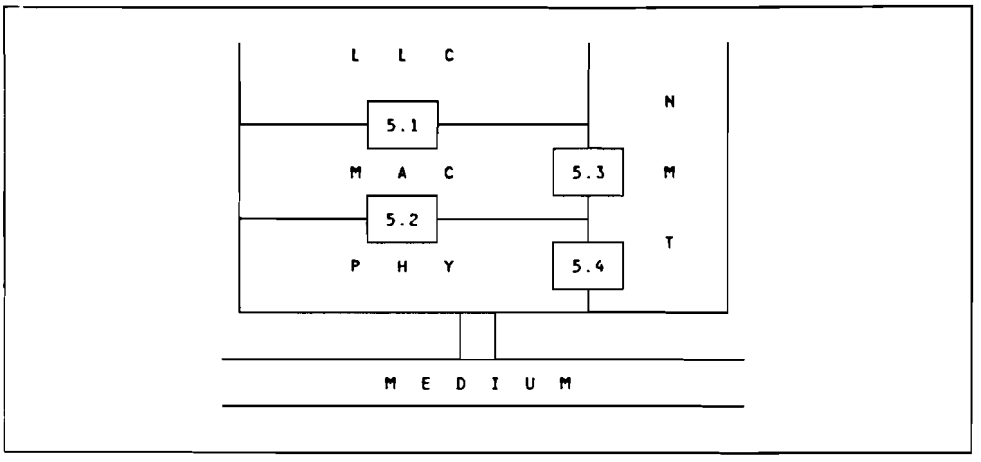
B.2 Standardized Service Specifications

5. Service Specifications

This section specifies the services provided:
(1) By the MAC sublayer to the Logical Link Control (LLC) sublayer
(2) By the PHY layer to the MAC sublayer
(3) By the MAC sublayer to NMT
(4) By the PHY layer to NMT

The services are described in an abstract way and do not imply any particular implementation or any exposed interface.

The diagram below serves as a guide to the subsections (5.1 through 5.4) that define the services provided.



5.1 MAC to LLC Service. This section specifies the services required of the MAC sublayer by the LLC to allow the local LLC sublayer entity to exchange LLC data units with peer LLC sublayer entities.

5.1.1 Interactions. The following primitives are defined for the LLC sublayer to request service from the MAC sublayer:
MA_DATA.request
MA_DATA.indication
MA_DATA.confirmation
All primitives described in this section are mandatory.

5.1.2 Detailed Service Specifications. All primitives are specified in an exemplary form only. Each service shall name the particular primitive and the required information that is passed between the LLC sublayer and MAC sublayer.

5.1.2.1 MA_DATA.request. This primitive defines the transfer of a MAC service data unit from a local LLC sublayer entity to a single-peer LLC entity, or multiple-peer LLC entities in the case of group addresses.

Semantics of the Service Primitive

```
MA_DATA.request (
                    frame_control,
                    destination_address,
                    m_sdu,
                    requested_service_class
                   )
```

The frame_control parameter specifies the value for the frame's FC octet. The destination_address parameter may specify either an individual or a group MAC entity address. It shall contain sufficient information to create the DA field that is appended to the frame by the local MAC sublayer entity as well as any lower-level address information. The m_sdu parameter specifies the MAC service data unit to be transmitted by the MAC sublayer entity. There is sufficient information associated with m_sdu for the MAC sublayer entity to determine the length of the data unit. The requested_service_class parameter specifies the priority (Pm) desired for the data unit transfer.

When Generated. This primitive shall be generated by the LLC sublayer entity whenever data must be transferred to a peer LLC entity or entities. This can be in response to a request from higher layers of protocol or from data generated internally to the LLC sublayer, such as required by LLC Type 2 service as defined by ANSI/IEEE Std 802.2-1985.

Effect of Receipt. The receipt of this primitive shall cause the MAC entity to append all MAC specific fields, including DA, SA, and any fields that are unique to the particular medium access method, and pass the properly formed frame to the lower layers of protocol for transfer to the peer MAC sublayer entity or entities.

Additional Comments. Requested_service_class is one of 8 levels.

5.1.2.2 MA_DATA.indication. This primitive defines the transfer of data from the MAC sublayer entity to the LLC sublayer entity or entities in the case of group addresses.

Semantics of the Service Primitive

```
MA_DATA.indication (
                      frame_control,
                      destination_address,
                      source_address,
                      m_sdu,
                      reception_status
                     )
```

The frame_control parameter is the FC octet received. The destination_address parameter may be either an individual or a group address as specified by the DA field of the incoming frame. The source_address parameter must be an individual address as specified by the SA field of the incoming frame. The m_sdu parameter shall specify the MAC service data unit as received by the local MAC entity. The reception_status parameter indicates the success or failure of the incoming frame. It consists of the following elements:

(1) frame_status: FR_GOOD, FR_WITH_ERROR. If an FR_WITH_ERROR is reported, the reason for the error shall also be reported. The reason shall be one of the following:

(a) invalid_FCS: calculated FCS does not match the received FCS

(b) code_violation: J or K symbol received between the SD and ED

(c) frame_truncated: the received frame, although free from errors, exceeded the internal buffer space

(d) short_frame: the received frame was shorter than the minimum

(2) E_value: zero, one, invalid

(3) A_&_C_value: zero_zero, one_zero, one_one, invalid

When Generated. The MA_DATA.indication primitive shall be generated by the MAC sublayer entity to the LLC sublayer entity or entities to indicate the arrival of an LLC frame at the local MAC sublayer entity. Such frames shall be reported only if they are validly formed and their destination address designates the local MAC entity, or the source address designates the local MAC entity if the station was so initialized (see 5.3.2.1).

Effect of Receipt. The effect of receipt of this primitive by the LLC sublayer is dependent upon the validity and content of the frame.

Additional Comments. If the local MAC sublayer entity is designated by the destination_address parameter of an MA_DATA.request primitive, the indication primitive shall also be invoked by the MAC entity to the local LLC entity. This full duplex characteristic of the MAC sublayer may be due to unique function capabilities within the MAC sublayer or full duplex characteristics of the lower layers; for example, all frames transmitted to the broadcast address shall invoke MA_DATA.indication primitives at all stations in the network including the station that generated the request.

5.1.2.3 MA_DATA.confirmation. This primitive has local significance and shall provide an appropriate response to the LLC sublayer MA_DATA.request primitive signifying the success or failure of the request.

Semantics of the Service Primitive

```
MA_DATA.confirmation (
                        transmission_status,
                        provided_service_class
                      )
```

The transmission_status parameter shall be used to pass status information back to the local requesting LLC sublayer entity. It shall be used to indicate the success or failure of the previous associated MA_DATA.request. The provided_service_class parameter specifies the service class that was provided for the data unit transfer.

When Generated. This primitive shall be generated by the MAC entity in response to an MA_DATA.request primitive from the local LLC sublayer entity.

Effect of Receipt. The effect of receipt of this primitive by the LLC sublayer is unspecified.

Additional Comments. It is assumed that sufficient information is available to the LLC sublayer to associate the response with the appropriate request.

5.2 PHY to MAC Service. The services provided by the PHY layer allow the local MAC sublayer entity to exchange MAC data units with peer MAC sublayer entities.

NOTE. All PHY data units have the duration of one symbol period.

5.2.1 Interactions. The following primitives are defined for the MAC sublayer to request service from the PHY layer:

PH_DATA.request
PH_DATA.indication
PH_DATA.confirmation

All primitives described in this section are mandatory.

5.2.2 Detailed Service Specifications. All primitives are specified in an exemplary form only. Each service shall name the particular primitive and the required information that shall be passed between the MAC sublayer and PHY layer.

5.2.2.1 PH_DATA.request. This primitive defines the transfer of data from a local MAC sublayer entity to the station's PHY layer.

Semantics of the Service Primitive

```
PH_DATA.request (
                 symbol
                )
```

The symbol specified shall be one of the following:

0 = binary zero
1 = binary one
J = non-data-J
K = non-data-K

When Generated. The MAC sublayer shall send the PHY layer a PH_DATA.request every time the MAC sublayer has a symbol to output. Once the MAC sublayer has sent a PH_DATA.request to the PHY layer, it may not send another PH_DATA.request until it has received a PH_DATA.confirmation from the PHY layer.

Effect of Receipt. Upon receipt of this primitive, the PHY entity shall encode and transmit the symbol. When the PHY entity is ready to accept another PH_DATA.request, it shall return to the MAC sublayer a PH_DATA.confirmation.

Additional Comments. None.

5.2.2.2 PH_DATA.indication. This primitive defines the transfer of data from the PHY layer to the MAC sublayer entity.

Semantics of the Service Primitive

```
PH_DATA.indication (
                    symbol
                   )
```

The symbol specified shall be one of the following:

0 = binary zero
1 = binary one
J = non-data-J
K = non-data-K

When Generated. The PHY layer shall send the MAC sublayer a PH_DATA.indication every time the PHY layer decodes a symbol. This indication is sent once every symbol period.

Effect of Receipt. Upon receipt of this primitive the MAC sublayer accepts a symbol from the PHY layer.

Additional Comments. None.

5.2.2.3 PH_DATA.confirmation. This primitive has local significance and shall provide an appropriate response to the MAC sublayer PH_DATA.request primitive signifying the acceptance of a symbol specified by the PH_DATA.request and willingness to accept another symbol.

Semantics of the Service Primitive

PH_DATA.confirmation (
transmission_status
)

The transmission_status parameter shall be used to signify the transmission completion status.

When Generated. The PHY layer shall send the MAC sublayer PH_DATA.confirmation in response to every PH_DATA.request received by the PHY layer. The purpose of the PH_DATA.confirmation is to synchronize the MAC sublayer data output with the data rate of the PHY layer medium.

Effect of Receipt. The receipt of this primitive enables the MAC sublayer to send another PH_DATA.request to the PHY layer.

Additional Comments. The PHY layer provides a *synchronous* service, that is, upon completion of a PH_DATA.confirmation, it expects an immediate PH_DATA.request.

5.3 MAC to NMT Service. This section specifies the services provided at the boundary between the network management and the MAC sublayer. This interface is used by NMT to monitor and control the operations of the MAC sublayer.

5.3.1 Interactions. The following primitives are defined for the NMT to request service from the MAC sublayer:

MA_INITIALIZE_PROTOCOL.request
MA_INITIALIZE_PROTOCOL.confirmation
MA_CONTROL.request
MA_STATUS.indication
MA_NMT_DATA.request
MA_NMT_DATA.indication
MA_NMT_DATA.confirmation

All primitives described in this section are mandatory.

5.3.2 Detailed Service Specifications. All primitives are specified in exemplary form only. Each service shall name the particular primitive and the required information that will be passed between the MAC sublayer and NMT.

5.3.2.1 MA_INITIALIZE_PROTOCOL.request. This primitive has local significance and is used by NMT to reset the MAC sublayer and optionally to change operational parameters of the MAC sublayer.

Semantics of the Service Primitive

```
MA_INITIALIZE_PROTOCOL.request
                                (
                                 individual_MAC_address,
                                 group_MAC_addresses,
                                 all_stations_this_ring_address,
                                 THT_value,
                                 TRR_value,
                                 TVX_value,
                                 TNT_value,
                                 TQP_value,
                                 TSM_value,
                                 TAM_value,
                                 priority_of AMP_data_unit,
                                 indicate_for_frame_with_SA=MA,
                                 indicate_for_rcv_only_good_frames
                                )
```

(1) The individual_MAC_address is the octet string the MAC sublayer will use as its individual address.

(2) The group_MAC_addresses is the octet string the MAC sublayer will use as its group addresses.

(3) The all_stations_this_ring_address parameter is the octet string the MAC sublayer will use as the destination address in frames sent to all stations, this ring. This value will also be used to determine whether to copy a frame sent by another station with a destination address of all stations, this ring. The default value is all ones.

(4) The THT_value is the value the MAC sublayer will use for its Timer, Holding Token (THT).

(5) The TRR_value is the value the MAC sublayer will use for the time-out value of its Timer, Return to Repeat (TRR).

(6) The TVX_value is the value the MAC sublayer will use for the time-out value of its Timer, Valid Transmission (TVX).

(7) The TNT_value is the value the MAC sublayer will use for the time-out value of its Timer, No Token (TNT).

(8) The TQP_value is the value the MAC sublayer will use for the time-out value of its Timer, Queue PDU (TQP).

(9) The TSM_value is the value the MAC sublayer will use for the time-out value of its Timer, Standby Monitor (TSM).

(10) The TAM_value is the value the MAC sublayer will use for the time-out value of its Timer, Active Monitor (TAM).

(11) The priority_of_AMP_data_unit parameter is the value the MAC sublayer will use for the requested_service_class when sending the AMP data unit (see 3.3.3).

(12) The indicate_for_frame_with_SA=MA parameter is the value the MAC sublayer will use to initialize the station to generate MA_DATA.indication and MA_NMT_DATA.indication primitives for frames that the station itself transmitted (that is, SA=MA).

(13) The indicate_for_rcv_only_good_frames parameter is the value the MAC sublayer will use to decide whether to generate MA_DATA.indication and MA_NMT_DATA.indication primitives only on frames that are good (see 4.2.1) or alternatively on all frames that are validly formed. In both cases data is terminated when a bit synchronization error is recognized.

NOTE. All parameters of this primitive are optional. If a parameter is omitted, the MAC sublayer will use the most recently provided value for this parameter or if no value has been previously provided, the default value for the parameter will be used. The default value for the individual_MAC_address parameter is not defined here.

When Generated. This primitive shall be generated by NMT whenever NMT requires the MAC sublayer to reset and reconfigure.

Effect on Receipt. Receipt of this primitive shall cause the MAC sublayer to reset its protocol and establish the values of its addresses, timers, and other initialization parameters. Upon completion of this primitive, the MAC sublayer shall generate a MA_INITIALIZE_PROTOCOL.confirmation.

Additional Comments. The timer values specified by NMT to the MAC sublayer by this primitive, may effect the maximum length frame that LLC may request the MAC sublayer to transmit. It is the responsibility of NMT to inform the appropriate higher layers responsible for segmenting or blocking messages of the MAC sublayer maximum frame size.

5.3.2.2 MA_INITIALIZE_PROTOCOL.confirmation. This primitive is used by the MAC sublayer to inform NMT that the MA_INITIALIZE_PROTOCOL.request primitive is complete.

Semantics of the Service Primitive

```
MA_INITIALIZE_PROTOCOL.confirmation (
                                      status
                                    )
```

The status parameter indicates the success or failure of the MA_INITIALIZE_PROTOCOL.request.

When Generated. This primitive shall be generated by MAC upon completion of a MA_INITIALIZE_PROTOCOL.request.

Effect on Receipt. Unspecified.

Additional Comments. None.

5.3.2.3 MA_CONTROL.request. This primitive has local significance and is used by NMT to control the operation of the MAC sublayer.

Semantics of the Service Primitive

```
MA_CONTROL.request (
                        control_action
                    )
```

The control_action parameter shall be one of the following:
MASTER RESET (see 4.2.3)
INSERT (see 4.2.3)

When Generated. This primitive shall be generated by NMT whenever NMT requires the MAC sublayer to take specific actions.

Effect on Receipt. Receipt of this primitive shall cause the MAC sublayer to take the action specified by the control_action parameter.

Additional Comments. None

5.3.2.4 MA_STATUS.indication. This primitive is used by the MAC sublayer to inform NMT of errors and significant status changes. The specific errors and status changes reported are defined in the following section.

Semantics of the Service Primitive

```
MA_STATUS.indication (
                        status_report
                    )
```

The status_report parameter shall be one of the following:

FRAME_CONDITION. See Receive Actions reference R-A.
TX_CLAIM_TOKEN_STATE. See Standby Monitor FSM transitions 11, 41, 52.
TX_BEACON_STATE. See Standby Monitor FSM transitions 53.
RECEIVE_FRAME_BEACON. See Standby Monitor FSM transition 42A.
ENTER_ACTIVE_STATE. See Active Monitor FSM transition 11.
ENTER_STANDBY_STATE. See Active Monitor FSM transitions 04, 22, and Standby Monitor FSM transitions 22,31,51.
DUPLICATE_ADD_DETECTED. See Standby Monitor FSM transition 21 and Receive Actions reference R-G.

When Generated. This primitive shall be generated by the MAC sublayer by the operation of the Operational, Standby Monitor, or Active Monitor FSMs.

Effect on Receipt. Unspecified.

Additional Comments. None

5.3.2.5 MA_NMT_DATA.request. This primitive defines the transfer of data from a local NMT entity to the local MAC entity.

Semantics of the Service Primitive

```
MA_NMT_DATA.request (
                     frame_control,
                     destination_address,
                     m_sdu,
                     requested_service_class
                    )
```

The frame_control parameter specifies the value for the frame's FC octet. The destination_address parameter may specify either an individual or a group MAC entity address. It shall contain sufficient information to create the DA field that is appended to the frame by the local MAC sublayer entity as well as any lower level address information. The m_sdu parameter specifies the MAC service data unit to be transmitted by the MAC sublayer entity. There is sufficient information associated with m_sdu for the MAC sublayer entity to determine the length of the data unit. The requested_service_class parameter specifies the priority (Pm) desired for the data unit transfer.

When Generated. This primitive shall be generated by the NMT entity whenever data must be transferred to one or more peer NMT entities.

Effect of Receipt. The receipt of this primitive shall cause the MAC entity to append all MAC specific fields, including DA, SA, and any fields that are unique to the particular medium access method, and pass the properly formed frame to the lower layers of protocol for transfer to the peer NMT entity or entities.

Additional Comments. Requested_service_class is one of 8 levels.

5.3.2.6 MA_NMT_DATA.indication. This primitive defines the transfer of data from the MAC sublayer entity to the NMT entity.

Semantics of the Service Primitive

```
MA_NMT_DATA.indication (
                        frame_control,
                        destination_address,
                        source_address,
                        m_sdu,
                        reception_status
                       )
```

The frame_control parameter is the FC octet received. The destination_address parameter may be either an individual or a group address as specified by the DA field of the incoming frame. The source_address parameter must be an individual address as specified by the SA field of the incoming frame. The m_sdu parameter shall specify the MAC service data unit as received by the local MAC entity. The reception_status parameter indicates the success or failure of the incoming frame. It consists of the following elements:

(1) frame_status: FR_GOOD, FR_WITH_ERROR.
If an FR_WITH_ERROR is reported, the reason for the error shall also be reported. The reason shall be one of the following:
(a) invalid_FCS: calculated FCS does not match the received FCS
(b) code_violation: J or K symbol received between the SD and ED
(c) frame_truncated: the received frame, although free from errors, exceeded the internal buffer space
(d) short_frame: the received frame was shorter than the minimum
(2) E_value: zero, one, invalid
(3) A_&_C_value: zero_zero, one_zero, one_one, invalid

When Generated. The MA_NMT_DATA.indication primitive shall be generated by the MAC sublayer entity to the NMT entity or entities to indicate the arrival of a MAC frame at the local MAC sublayer entity. Such frames shall be reported only if they are validly formed and their destination address designates the local MAC entity, or the source address designates the local MAC entity if the station was so initialized (see 5.3.2.1).

Effect of Receipt. The effect of receipt of this primitive by NMT is dependent upon the validity and content of the frame.

Additional Comments. If the local MAC sublayer entity is designated by the destination_address parameter of a MA_NMT_DATA.request primitive, the indication primitive shall also be invoked by the MAC entity to the local NMT entity. This full duplex characteristic of the MAC sublayer may be due to unique function capabilities within the MAC sublayer or full duplex characteristics of the lower layers (for example, frames transmitted to the broadcast address shall invoke MA_NMT_DATA.indication primitives at all stations in the network including the station that generated the request).

5.3.2.7 MA_NMT_DATA.confirmation. This primitive has local significance and shall provide an appropriate response to the NMT's MA_NMT_DATA.request primitive signifying the success or failure of the request.

Semantics of the Service Primitive

```
MA_NMT_DATA.confirmation (
                          transmission_status,
                          provided_service_class
                         )
```

The transmission_status parameter shall be used to pass status information back to the local requesting NMT entity. It shall be used to indicate the success or failure of the previous associated MA_DATA.request. The provided_service_class parameter specifies the service class that was provided for the data unit transfer.

When Generated. This primitive shall be generated by MAC in response to an MA_NMT_DATA.request from the local NMT entity.

Effect of Receipt. The effect of receipt of this primitive by the NMT is unspecified.

Additional Comments. It is assumed that sufficient information is available to the NMT entity to associate the response with the appropriate request.

5.4 PHY to NMT Service. The services provided by the PHY layer to NMT allow the local NMT to control the operation of the PHY layer.

5.4.1 Interactions. The following primitives are defined for the NMT to request services from the PHY layer

PH_CONTROL.request
PH_STATUS.indication

All primitives described in this section are mandatory.

5.4.2 Detailed Service Specifications. This primitive is specified in exemplary form only. The service shall name the primitive and specify the information that will be passed between PHY and NMT.

5.4.2.1 PH_CONTROL.request. This primitive shall be generated by NMT to request the PHY layer to insert or remove itself to/from the ring.

Semantics of the Service Primitives

```
PH_CONTROL.request (
                    control_action
                   )
```

The control_action parameter shall be one of the following:

INSERT: signal insertion into ring
REMOVE: signal removal from ring

When Generated. This primitive shall be generated by NMT when NMT requires insertion or removal of the station from the ring.

Effect Upon Receipt. The PHY layer shall take appropriate action to cause insertion or removal from the ring. See 7.4 for specific actions for shielded twisted pair medium.

Additional Comments. None.

5.4.2.2 PH_STATUS.indication. This primitive is used by the PHY layer to inform NMT of errors and significant status changes. The specific errors and status changes reported are defined in the following section.

Semantics of the Service Primitives

```
PH_STATUS.indication (
                        status_report
                      )
```

The status_report parameter shall be one of the following:

BURST_CORRECTION_START. The PHY layer has begun generating 0 or 1 symbols and passing them to the MAC sublayer (on the PH_DATA.indication) to correct detected silence on the medium.
BURST_CORRECTION_END. The PHY layer has stopped generating symbols; transitions have again been detected on the medium.
LATENCY_BUFFER_OVERFLOW. The PHY layer has attempted to expand the latency buffer beyond 30 bits.
LATENCY_BUFFER_UNDERFLOW. The PHY layer has attempted to contract the latency buffer beyond 24 bits.

When Generated. This primitive shall be generated by the PHY layer by its operation, as defined in 6.1 and 6.5.

Effect Upon Receipt. Unspecified.

Additional Comments. None.

B.3 Standardized Physical Layer Specification

6. Physical Layer

The following sections define physical (PHY) layer specifications. These include data symbol encoding and decoding, symbol timing, and reliability.

Throughout this section the word *repeater* is used to mean the repeater part of a station or a separate unit.

6.1 Symbol Encoding. The PHY layer encodes and transmits the four symbols presented to it at its MAC interface by the MAC sublayer.

The symbols exchanged between the MAC and PHY layers are shown below. (Specific implementations are not constrained in the method of making this information available.)

0 = binary zero
1 = binary one
J = non-data-J
K = non-data-K

As shown in Fig 6-1, the symbols are transmitted to the medium in the form of differential Manchester-type coding which is characterized by the transmission of two line signal elements per symbol.

In the case of the two data symbols, binary one and binary zero, a signal element of one polarity is transmitted for one half the duration of the symbol to be transmitted, followed by the contiguous transmission of a signal element of the opposite polarity for the remainder of the symbol duration. This provides two distinct advantages:

(1) The resulting signal has no dc component and can readily be inductively or capacitively coupled

(2) The forced *mid-bit* transition conveys inherent timing information on the channel

In the case of differential Manchester coding, the sequence of line signal element polarities is completely dependent on the polarity of the trailing signal element of the previously transmitted data or non-data symbol (bit). If the symbol to be transmitted is a binary zero, the polarity of the leading signal element of the sequence is opposite to that of the trailing element of the previous symbol and, consequently, a transition occurs at the bit (symbol) boundary as well as mid-bit. If the symbol to be transmitted is a binary one, the algorithm is reversed and the polarity of the leading signal element is the same as that of the trailing signal element of the previous bit. Here there is no transition at the bit (symbol) boundary.

The non-data symbols, J and K, depart from the above rule in that a signal element of the same polarity is transmitted for both signal elements of the symbol and there is therefore no mid-bit transition. A J symbol has the same polarity as the preceding symbol whereas a K symbol has the opposite polarity to the preceding symbol. The transmission of only one non-data symbol introduces a dc component on the ring. To avoid an accumulating dc component,

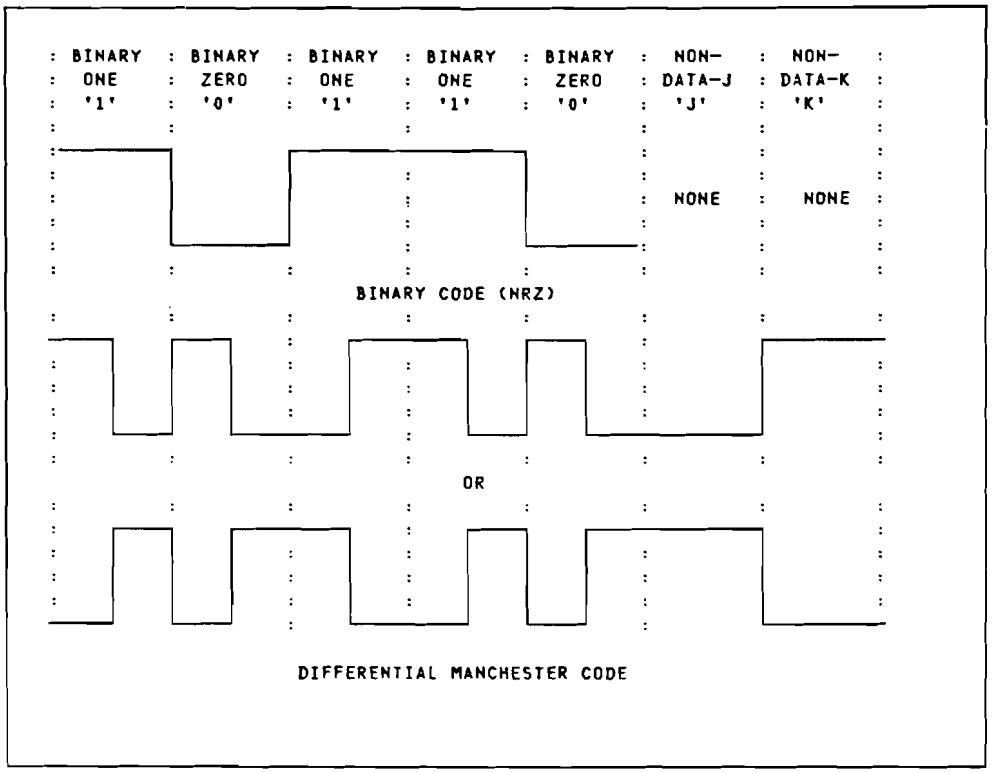


Fig 6-1
Example of Symbol Encoding

non-data symbols are normally transmitted as a pair of J and K symbols. (By its nature a K symbol is opposite to the polarity of the preceding symbol.)

6.2 Symbol Decoding. Received symbols shall be decoded using an algorithm that is the inverse of the one described for symbol encoding, and the decoded symbols shall be presented at the MAC interface.

If the PHY layer receives more than four signal elements of the same polarity in succession, it shall introduce a change of polarity (that is, a transition) at the end of the fourth signal element in the received bit stream and continue to introduce a transition each signal element time until a transition is received from the ring. The resulting bit stream is then decoded and the symbols presented to the MAC interface.

In a similar manner, during periods of loss of clock synchronization or underrun/overrun of the latency buffer, the PHY layer shall generate a transition each signal element time, decode the new bit stream, and present the resulting symbols to the MAC interface.

6.3 Data Signalling Rates. The data signalling rates shall be 1 or 4 Mbit/s with a tolerance of ±0.01%.

6.4 Symbol Timing. The PHY layer shall recover the symbol timing information inherent in the transitions between levels of the received signal. It shall minimize the phase jitter in this recovered timing signal to provide suitable timing at the data signalling rate for internal use and for the transmission of symbols on the ring. The rate at which symbols are transmitted is adjusted continuously in order to remain in phase with the receive signal.

In normal operation there is one station on the ring that is the active monitor. All other stations on the ring are frequency and phase locked to this station. They extract timing from the received data by means of a phase locked loop. The phase locked loop design shall be based on the following criteria:

(1) It shall limit the dynamic alignment jitter at any station in the ring to a 3 sigma value of 10°.

(2) Whenever a station is inserted into the ring or loses phase lock with the upstream station, it shall, upon receipt of a signal which is within specification from the upstream station (re)acquire phase lock within 1.5 ms.

(3) It shall accommodate at least a combined total of 250 stations and repeaters on the ring.

(4) It shall operate with a receive signal as specified in Section 7.

(5) It shall operate with a jitter power spectral density of $2.5 \cdot 10^{-25}$ s^2/Hz, which may have been added by the medium interface cable and medium to the output of the upstream station.

NOTE: Items 1, 2, and 3 above require the design of the phase lock loop to meet the simultaneous requirements of large loop bandwidth to meet the 1.5 ms clock acquisition, and high damping to meet the 250 station capability. The loop transfer function must be designed to have a gain overshoot that is less than 0.2 dB above 0.0 dB.

6.5 Latency Buffer. The latency buffer is provided by the active monitor. It serves two distinct functions.

Assured Minimum Latency. In order for the token to continuously circulate around the ring when all stations are in repeat mode, the ring must have a latency (that is, time, expressed in number of bits transmitted, for a signal element to proceed around the entire ring) of at least the number of bits in the token sequence, that is, 24. Since the latency of the ring varies from one system to another and no a priori knowledge is available, a delay of at least 24 bits shall be provided by the active monitor.

Phase Jitter Compensation. The source timing or master oscillator of the ring shall be supplied by the active monitor station. All other stations in the ring track the frequency and phase of the incoming signal they receive. Although the mean data signalling rate around the ring is controlled by the active monitor station, segments of the ring can, instantaneously, operate at speeds slightly higher or lower than the frequency of the master oscillator. The cumulative effect of these variations in speed are sufficient to cause effec-

tive variations of up to ± 3 bits in the latency of a ring that has been configured with a maximum number of stations (that is, 250).

However, unless the latency of the ring remains constant, bits will be either dropped (not retransmitted) as the latency of the ring decreases or added as the latency increases. In order to maintain a constant ring latency, an elastic buffer with a length of 6 bits (12 signal elements) is added to the fixed 24-bit buffer. The resulting 30-bit buffer is initialized to 27 bits. If the received signal at the active monitor station is slightly faster than the master oscillator, the buffer will expand, as required, to 28, 29, or 30 bits to avoid dropping bits. If the received signal is slow, the buffer will contract to 26, 25, or 24 bits to avoid adding bits to the repeated bit stream.

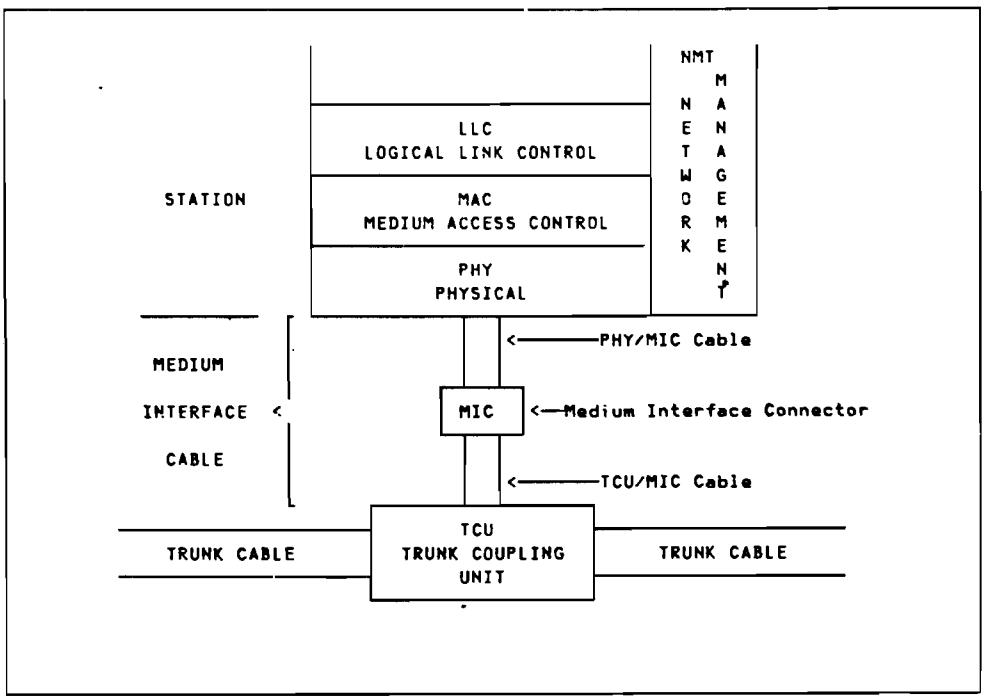
7. Station Attachment Specifications— Shielded Twisted Pair

7.1 Scope. This section specifies the functional, electrical, and mechanical characteristics of balanced, baseband, shielded twisted pair attachment to the trunk cable of a token ring.

7.2 Overview. The function of the trunk cable medium is to transport data signals between successive stations of a baseband ring local area network. This communications medium consists of a set of TCU's interconnected sequentially by trunk cable links. Each TCU is connected to a TCU/MIC cable to which a station may be connected. The relationship between these embodiments and the LAN model are shown in Fig 7-1.

Repeaters may be used, where required, to extend the length of a trunk link beyond limits imposed by normal signal degradation due to link impairments. These repeaters serve to restore the amplitude, shape, and timing of signals passing through them. The repeater's regenerative functions have the same characteristics as a repeating station on the ring and must be included in the count of the number of stations supported by the ring.

Fig 7-1
Partitioning of the Physical Layer and Medium



The medium interface cable (MIC) shown in Fig 7-1 may be as shown or may include multiple sections of cable joined by connectors identical to the MIC. By definition, the MIC is the connector at which all transmitted and received signal specifications shall be met. It may be attached to the station directly or on a *pig tail*.

7.3 Coupling of the Station to the Ring. The connection of the station to the trunk cable medium shall be via a shielded cable containing two balanced, 150 ± 15 Ω twisted pairs. The station transmitter shall deliver the specified signal at the MIC, and the station receiver shall have sufficient sensitivity and distortion margin to operate properly with the appearance of the specified signal levels and distortion at this interface point. The shield of the cables shall be connected to the shield terminal of the MIC.

An exemplary implementation of the connection, in bypass mode, of the station to the ring is shown in Fig 7-2.

7.4 Ring Access Control. Station insertion into the ring is controlled by the station. The mechanism for effecting the insertion or bypass of the station resides in the TCU. The station exercises control of the mechanism via the media interface cable using a phantom circuit technique. The phantom circuit impresses a dc voltage on the MIC. This dc voltage is transparent to the passage of station-transmitted symbols, hence the name *phantom*. The voltage impressed is used within the TCU to effect the transfer of a switching action to cause the serial insertion of the station in the ring. Cessation of the phantom drive causes a switching action which will bypass the station and cause the station to be put in a looped (*wrapped*) state. This loop may be used by the station for off-line self-testing functions.

The phantom drive circuit is designed such that the station may detect open-wire and certain short-circuit faults in either the receive pair or transmit pair of signal wires. This is done by detecting dc current imbalance in two separate phantom circuits. In order to do this the transformers (or their equivalent) in the TCU and the station must provide two coils which are dc isolated but ac signal coupled to each other. Circuits attached between the transmit pair and the receive pair of conductors shall be designed such that a line-to-line dc current balance is maintained within each pair.

7.4.1 Current and Voltage Limits. The point of measurement of the voltage and current limits is at the MIC.

Insertion shall be effected with a voltage of 4.1 to 7.0 V on MIC pin B and O with return on pin G and R, respectively, within the current range of 0.65 to 2.0 mA.

Bypass shall be effected when a voltage of less than 1 V is present on MIC pins B and O with respect to pins R and G.

A load with a dc resistance within 5% of the insertion/bypass mechanism resistance shall be presented by the TCU on pins G and O.

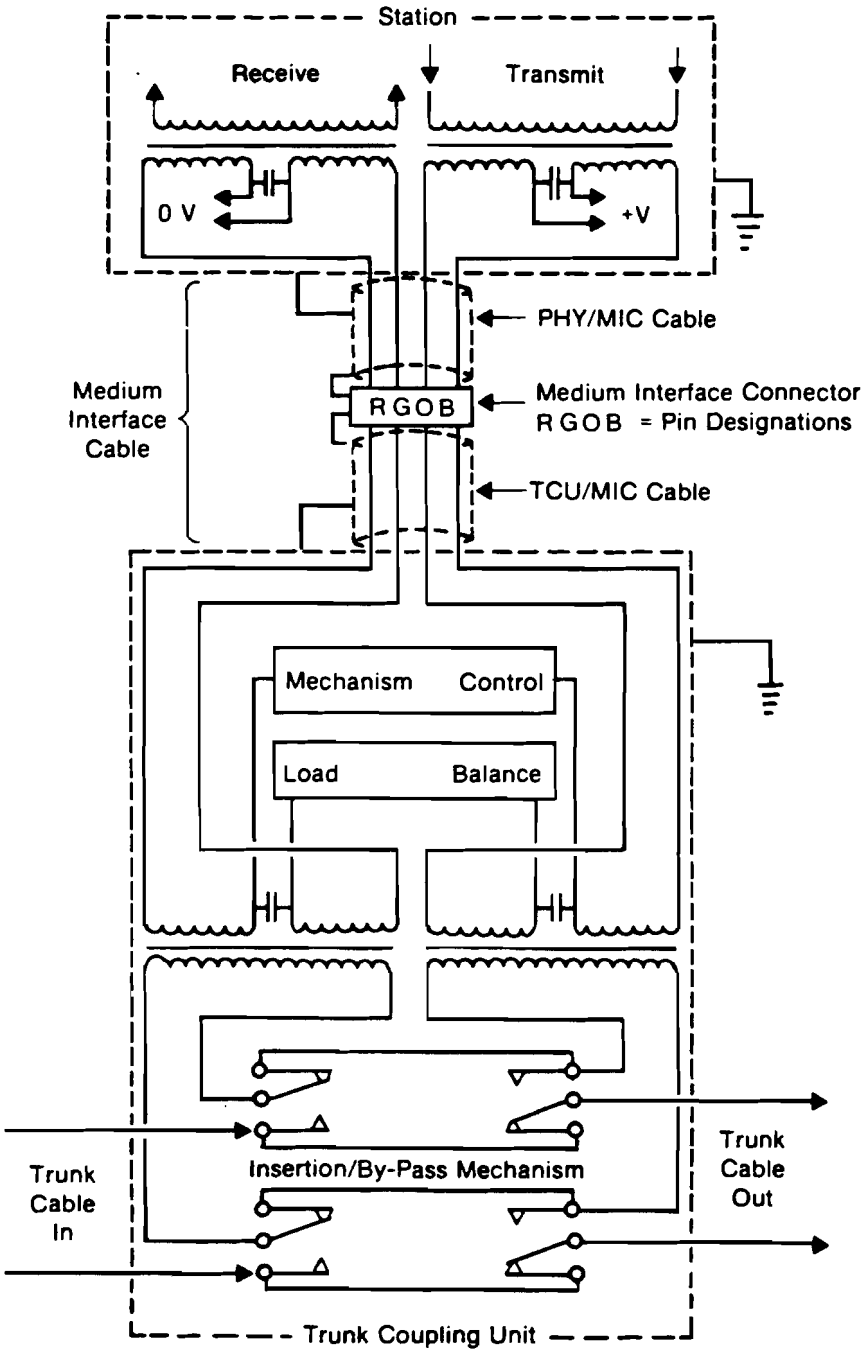


Fig 7-2
Example of Station Connection to the Medium

The operating voltage supplied by the station on MIC pins B and O shall be within 1% of each other over the operating current range of 0.65 to 2.0 mA.

The MIC, as described later, will automatically short circuit pin R to pin O and pin G to B when it is withdrawn. Therefore, the station shall provide means to assure the short circuit current will not exceed 20 mA.

7.4.2 Insertion/Bypass Transfer Timing. The insertion/bypass mechanism shall break the existing circuit before establishing the new circuit. The maximum time that the ring trunk circuit is open shall not exceed 5 ms.

7.5 Signal Characteristics

7.5.1 Transmitted Signals

Data Signalling Rates. The data signalling rates are 1 or 4 Mbit/s. The permitted tolerance for each signalling rate is ± 0.01%.

Signal Jitter. Maximum cumulative deviation of a transmitted signal element transition from the ideal transition (that is, timing distortion and *jitter*) measured at the MIC shall have a 3 sigma value of 10°.

Signal Level. The magnitude of the transmitted signal, measured at the MIC, with a 150 Ω resistive termination, shall be 3.0 to 4.5 V, peak to peak. The amplitude of the positive and the negative transmitted levels shall be balanced within 5%.

Rise/Fall Times. During transitions of the transmitted signals between alternating binary states, the differential voltage measured across a 150 ± 15 Ω test load at the MIC shall be such that the voltage changes between the 10% and 90% points of the output signal within a time interval shall be no greater than 25 ns for a 4 Mbit/s data rate (100 ns for a 1 Mbit/s data rate). In addition, the harmonic content of the transmitted signal generated by a pattern of all 0's or all 1's shall meet the following requirement:

(1) 2nd and 3rd harmonics: each at least 10 dB below fundamental
(2) 4th and 5th harmonics: each at least 15 dB below fundamental
(3) 6th and 7th harmonics: each at least 20 dB below fundamental
(4) all higher harmonics: each at least 25 dB below fundamental

7.5.2 Received Signals. The transmission medium may distort the transmitted signal. The distortion is bounded by the distortion produced by the cable which has a square root of the frequency attenuation characteristic.

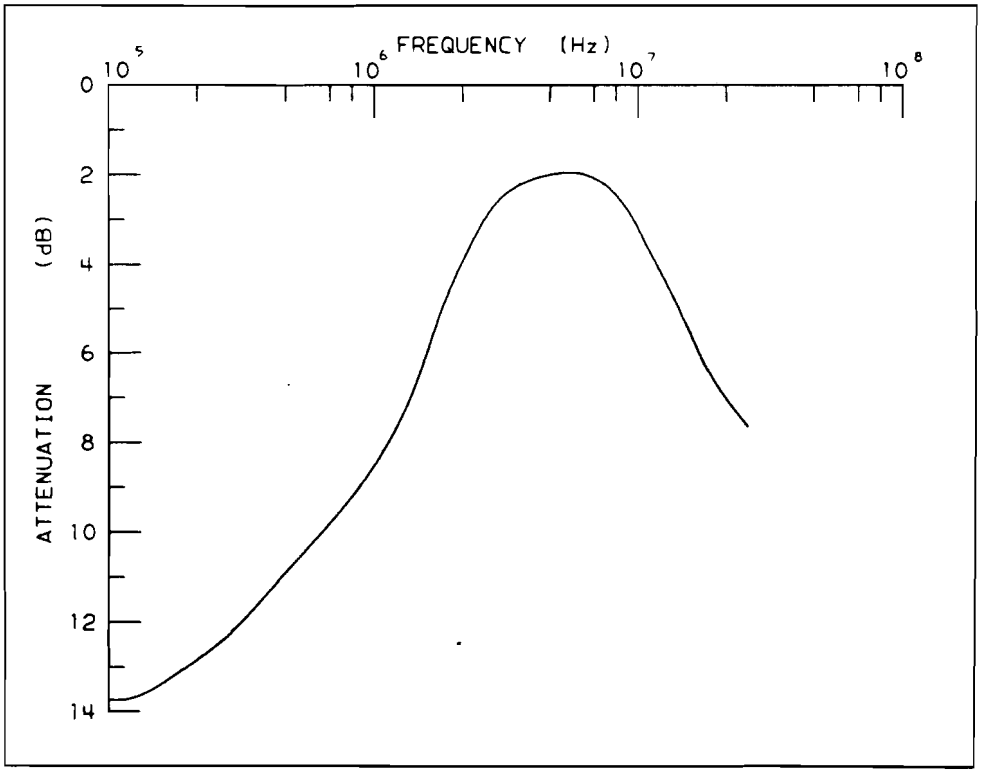
(NOTE: This characteristic is well known and can be found in many reference text books. Specifically, the form of the characteristic is in *Reference Data for Radio Engineers* [4].[2]

[2]The numbers in brackets correspond to those in the references at the end of this section.

In addition, flat (non-distorting) attenuation may be caused by the medium, especially TCU's and connectors. The total attenuation may vary from 0 to 29 dB at 4 MHz (at 1 MHz for 1 Mbit/s data rate) including flat attenuation not exceeding 15 dB, and cable attenuation not exceeding 26 dB at 4 MHz (at 1 MHz for 1 Mbit/s data rate). The total allowable attenuation may be less than 29 dB based on the actual noise level at the MIC and the required error rate of the LAN. The error rate required of a LAN shall be established by mutual agreement among the users of the LAN but in no case shall it be less than 10^{-8}.

In order to specify meaningful measurements at the MIC, a measurement is outlined that, while not part of the specification, allows confirmation of system level conformance. All received signals and noise will be specified at the output of an equalizing filter. The filter is a 2-pole, 1-zero device. For 4 Mbit ring operation the filter shall have poles at 2.7 MHz and 16 MHz, and zero at 540 kHz, each with a tolerance of ±5%. (For 1 Mbit operation, the frequency points are all divided by 4.) A plot of the characteristics of the filter are shown in Fig 7-3.

Fig 7-3
Receive Filter Characteristics for 4 Mbit/s Operational
150 Ω Impedance

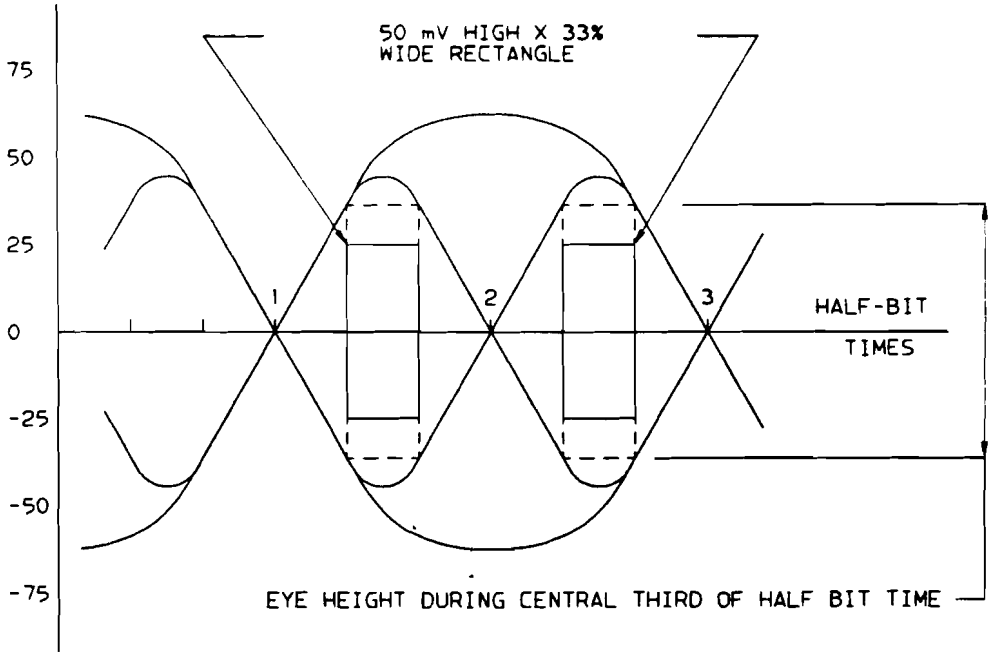


Signal Level. The receive signal at the output of the terminated filter shall have at least a magnitude of 25 mV during the central third of the half-bit time. Fig 7-4 is the characteristic *eye* pattern of the received signal when viewed on an oscilloscope triggered by a non-critical phase lock loop with a band width equal to or less than 0.01 times the data rate. A compliant signal shall have an opening such that a rectangular area of 50 mV high (2 × 25 mV) and a width of 33% of the half-bit time will fit, symmetrically, within the eye as shown in Fig 7-4.

Error Rate. The station shall provide an output with an error rate of $\leq 10^{-9}$ when the S/N (signal-to-noise ratio) at the output of the specified filter is ≥ 22 dB. S/N, measured in dB, is defined as 20 log ($\frac{1}{2}$ minimum eye height during the central third of the half-bit time divided by rms noise).

7.6 Reliability. The MAC, PHY layers, and connecting cable up to and including the MIC of each station shall be designed to minimize the probability of causing communication failure among other stations attached to the local network. The mean time to the occurrence of such a failure shall be at least one million hours of operation without requiring manual intervention to restore the network to operational status.

Fig 7-4
Receive Signal Eye Pattern



7.7 Safety and Grounding Requirements. All stations meeting this Standard shall conform to either IEC Standard 380 [2] or IEC Standard 435 [3].

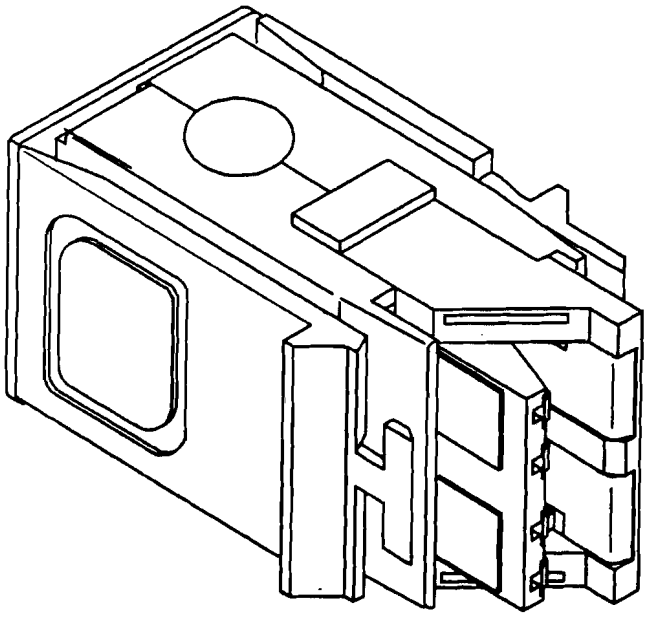
All exposed materials shall meet appropriate flammability requirements. Low smoke and fume materials shall be used as mandated by local requirements.

7.8 Electromagnetic Susceptibility. Sources of interference from the environment include but are not limited to electromagnetic fields, electrostatic discharge, and transient voltages between earth connections.

The station hardware shall meet its specifications when operating in an ambient plane wave field of 2 V/m from 10 kHz to 30 MHz and 5 V/m from 30 MHz to 1 GHz.

7.9 Medium Interface Connector (MIC). Figure 7-5 shows an isometric view of the medium interface connector as it would be oriented when it is wall-mounted. It has four signal contacts with a ground contact and is hermaphroditic in design so that two identical units will mate when oriented 180° with respect to each other.

Fig 7-5
Medium Interface Connector—Isometric View



Electrical Characteristics

crosstalk rejection	> 62 dB @ 100 kHz to 4 MHz
connector insertion loss in a 150 Ω impedance line	< 0.1 dB @ 100 kHz to 4 MHz
dc contact resistance (connection according to IEC 130-14 [1])	
pins	20 mΩ average, 100 mΩ maximum
shield	25 mΩ average, 100 mΩ maximum
self-shorting path	40 mΩ average, 100 mΩ maximum
carry current	≥ 0.1 A
voltage proof contact-contact	≥ 750 V dc

Mechanical Characteristics

contact force	0.5 - 1.0 N
insertions	> 1000
life span	> 15 years
surface treatment (compatible with the following):	
point-of-pin contact—plating with 3 μm of hard gold	
point-of-shield contact—plating with 5 μm of tin	

7.9.1 Medium Interface Connector — Contactor Detail. Figure 7-6 shows the details of the signal and ground contractors. When the connector is disconnected, pin R shall be shorted to pin O and pin G shorted to pin B for automatic looping capability. Only those dimensions that are essential to mating are shown.

7.9.2 Medium Interface Connector — Locking Mechanism Detail. Figure 7-7 shows the locking mechanism of the connector. Only those dimensions that are essential to mating are shown.

7.10 References

When the following standards referred to in this standard are superseded by an approved revision, the latest revision shall apply.

[1] IEC Publication 130-14 (1975), Part 14, Multi-Row Board Mounted Printed Board Connectors Having Contact and Termination Spacing on a 2.54 mm (0.1 in) Square Grid.[3]

[2] IEC Publication 380 (1977) (Second Edition), Safety of Electrically Energized Office Machines.

[3] IEC Publication 435 (1983) (Second Edition), Safety of Data Processing Equipment.

[4] Reference Data for Radio Engineers, 4th ed, ITT, p 574, 1956.

[3]IEC Standards are available in the US from American National Standards Institute, 1430 Broadway, New York, NY 10018.

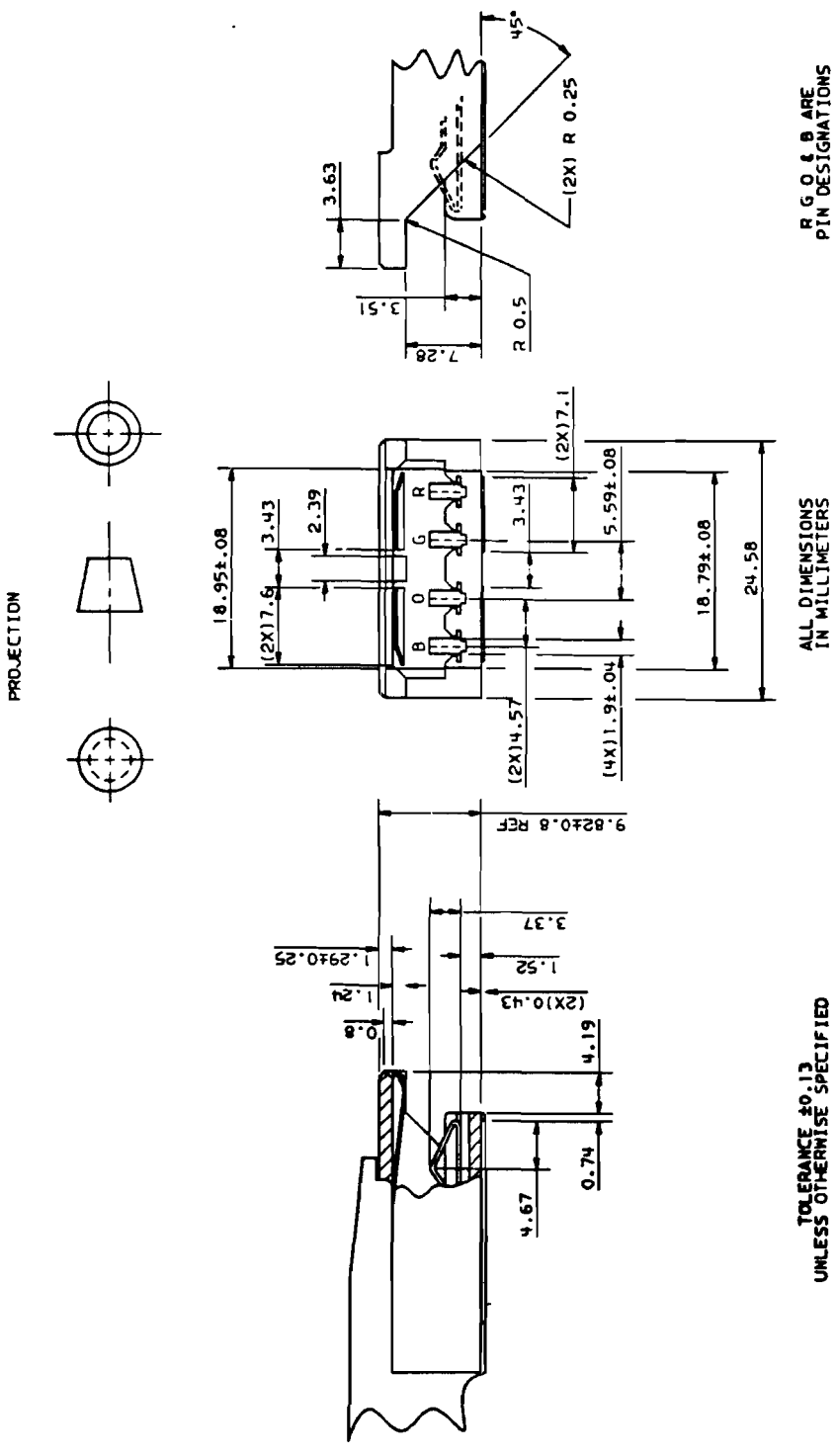


Fig 7-6
Medium Interface Connector — Contactor Detail

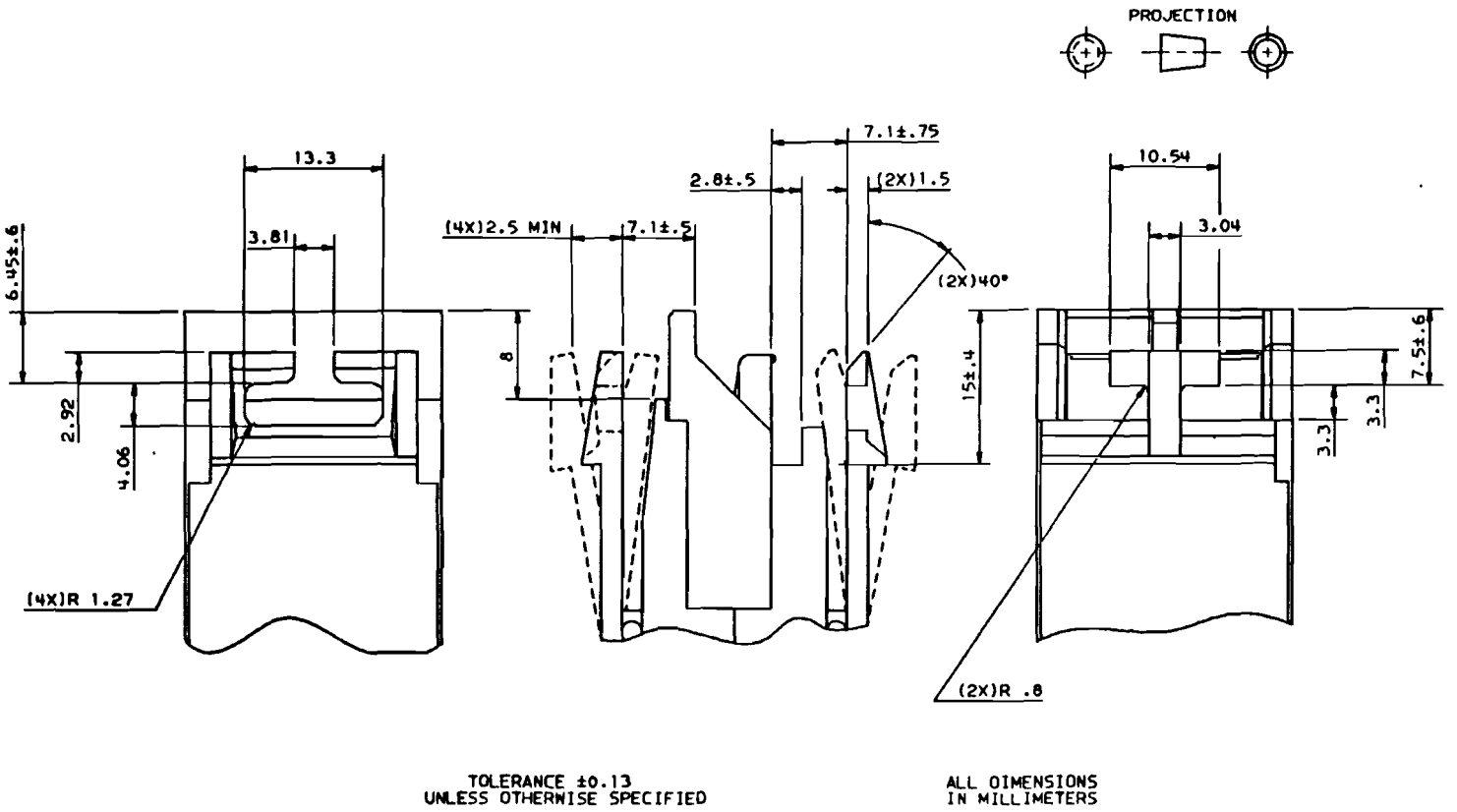


Fig 7-7
Medium Interface Connector — Locking Mechanism Detail