apollo®

D O M A I N

Order No. 010005
Revision 00

Apollo Token Ring Media Access Control Layer and Physical Layer Protocols

Apollo Token Ring Media Access Control Layer and Physical Layer Protocols

Order No. 010005
Revision 00

Apollo Computer Inc.
330 Billerica Road
Chelmsford, MA 01824

Preface

Apollo Token Ring Media Access Control Layer and Physical Layer Protocols specifies the formats and protocols used by the Apollo token ring's Media Access Control (MAC) and physical layers. This manual also provides pointers to other Apollo–specific documents that describe methods for physically attaching workstations to our token ring network.

We've organized this manual as follows:

Chapter 1	Presents an overview of the Apollo token ring, using the network models of the Institute of Electrical and Electronics Engineers (IEEE) and the International Standards Organization (ISO) as reference points.
Chapter 2	Describes the formats and protocols within the media access control layer of the Apollo token ring.
Chapter 3	Describes the formats and protocols within the physical layer of the Apollo token ring.
Appendix A	Describes coaxial driver, receiver, and cable specifications for an Apollo token ring network.

Audience

Readers of this book should be technical professionals who are familiar with the vocabulary and basic concepts behind local area networks. Use this document and related Apollo manuals to gain an understanding of how communications take place over the Apollo token ring network at the media access control and physical layers. For definitions of these layers, see Section 1.2 in Chapter 1.

Related Manuals

For more information about physically attaching workstations to the the Apollo token ring, see *Domain Hardware Site Planning Specifications* (009859), *Installing Coaxial Cable and Accessories for a Domain Token Ring Network* (009860), and *Planning Domain Networks and Internets* (009916).

Problems, Questions, and Suggestions

We appreciate comments from the people who use our system. In order to make it easy for you to communicate with us, we provide the User Change Request (UCR) system for software-related comments, and the Reader's Response form for documentation comments. By using these formal channels you make it easy for us to respond to your comments.

You can get more information about how to submit a UCR by consulting the *Domain System Command Reference*. Refer to the CRUCR (CREATE_USER_CHANGE_REQUEST) Shell command description. You can view the same description on-line by typing:

```
$ HELP CRUCR <RETURN>
```

For your documentation comments, we've included a Reader's Response form at the back of each manual.

Contents

Illustrations

Tables

Chapter 1

Overview

1.1 Introduction

The Apollo token ring network is a 12–megabit–per–second Local Area Network (LAN) that links workstations or servers (nodes) together. Each node can gain access to the network by using a token (a special bit pattern that circulates around the ring, passing through each node). Only one token exists on the ring at any given time. Consequently, the network never experiences collisions (or retries due to collisions).

In the Apollo token ring network, data travels around the ring in *one* direction, and each node transmits bits of the serial data stream as it receives them. This process is called transceiving.

Apollo token ring controller hardware allows you to:

- Detect and isolate network failures
- Connect hundreds of nodes in a single network (without using external transceivers and repeaters)
- Connect/disconnect nodes instantly.

A well–known limitation of serially wired ring networks in general is that individual node failures can interrupt network operations. However, the unique features of the Apollo token ring enable you to pinpoint the failing node and instantly remove it from the ring, without compromising network integrity or performance.

Figure 1–1 illustrates the Apollo token ring.

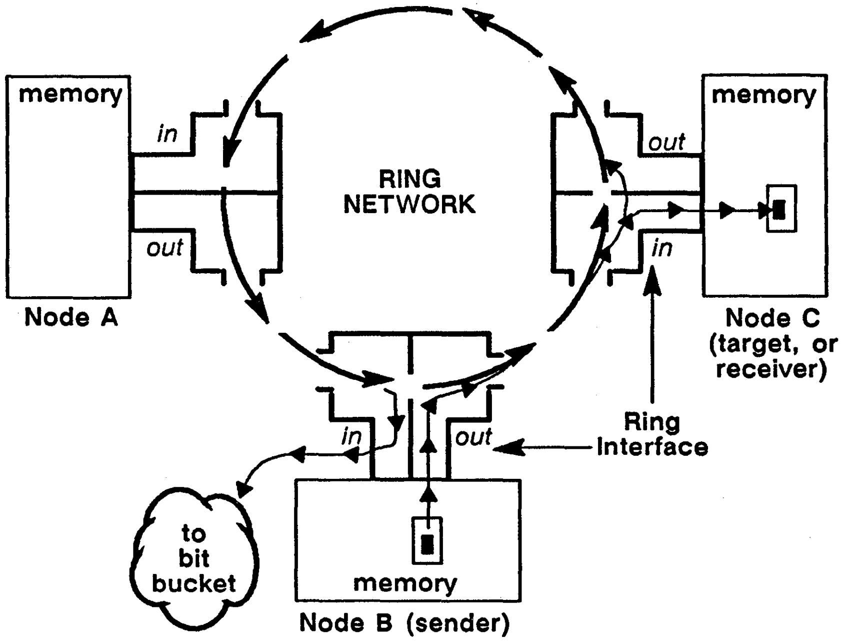


Figure 1-1. Apollo Token Ring

1.2 Protocol Layers

Protocols determine the methods by which orderly communications take place over the ring. Specifically, protocols at the physical layer ensure that serial bit data is accurately transmitted and received over a communications channel. Protocols at the media access control layer synchronize and error-check the transmitted data, and provide arbitration for access to the network.

Using the network models of the Institute of Electrical and Electronics Engineers (IEEE) and the International Standards Organization (ISO) as reference points, Table 1-1 shows the two Apollo-specific protocol "layers" (media access control and physical) described within this document.

Table 1-1. **Protocol Layers Described within this Document (Shown Highlighted)**

ISO/OSI MODEL	IEEE 802 MODEL	APOLLO TOKEN RING PROTOCOLS DESCRIBED HERE
Application Layer	Application Layer	Application Layer
Presentation Layer	Presentation Layer	Presentation Layer
Session Layer	Session Layer	Session Layer
Transport Layer	Transport Layer	Transport Layer
Network Layer	Network Layer	Network Layer
Data Link Layer	Logical Link Control (LLC)	Media Access Control (MAC)
	Media Access Control (MAC)	
Physical Layer	Physical Layer (PHY)	Physical Layer

NOTE: In this document, we discuss Apollo token ring protocols up to the ISO/OSI data-link level described in: *ISO 7498 Information Processing Systems – Open System Interconnection – Basic Reference Model.*

Chapter 2

Media Access Control (MAC) Layer

2.1 Introduction

The sequence for transmitting on the Apollo token ring occurs like this:

1. A node generates a *packet* and enables its transmitter. The packet consists of a data stream; it contains addressing and control information, and it may contain a message as well.
2. A bit pattern called the *free token* passes through each node on the ring. When the free token passes through a node that is waiting to transmit, that node modifies the free token into a *claimed token* in order to acquire the network.
3. Once a node has acquired the network, it breaks ring recirculation and begins to transmit its packet. Concurrently, it begins to discard received data (including its own packet, which will eventually come back around the ring). The process of discarding received data is called *stripping*.
4. The transmitting node *pads the bit serial stream with Zeros*, just after transmitting the newly modified claimed token. The Zeros provide the transmitting and receiving nodes time to perform setup tasks required for data transfer.
5. Here, the MAC protocols create an envelope, or *frame*, around the packet. A frame consists of a frame start sequence, the packet (itself broken into a packet header sequence and a packet data sequence), a frame check sequence, and an end–of–frame sequence.
6. The transmitting node sends out a new free token to follow the frame. This new free token may be modified (into a claimed token) by other nodes waiting to transmit.

7. The transmitting node continues to strip all data from the ring until it finishes receiving its own frame, or until a 10.9 msec (2^{14} byte) timeout occurs. This timeout prevents a node from stripping bits forever (for example, if its frame has gotten lost on the ring). The transmitting node pads the bit serial stream with Zeros again, until its own frame is completely stripped from the ring.

8. When a node stops stripping, recirculation resumes around the ring.

The following section provides a detailed description of the protocols at the Apollo token ring's MAC layer.

2.2 Media Access Control (MAC) Protocols

Bits travel in a serial stream around the ring, and hardware uses a Synchronous Data Link Control–(SDLC–) type of bit–stuffing algorithm to align them. Bit–stuffing yields two discrete bit patterns (sometimes called characters). These are

1. Normally expected bit patterns (these contain "normal" data, such as a message)

2. Out–of–band bit patterns (these contain control information, such as "the frame starts here").

Out–of–band characters generate byte boundaries within the bit stream. Such boundaries synchronize the way sender and receiver view the bits traveling between them. If the out–of–band characters are not present in the appropriate order and if they are not properly bit–aligned, protocol errors will occur.

At the MAC layer, hardware must "see" all the data — including the out–of–band characters — sequentially, and all data must be transmitted most–significant–bit first (in a byte) and most–significant–byte first (in a multi–byte field). Figure 2–1 shows how data must be formatted for transmission and reception according to MAC layer protocols.

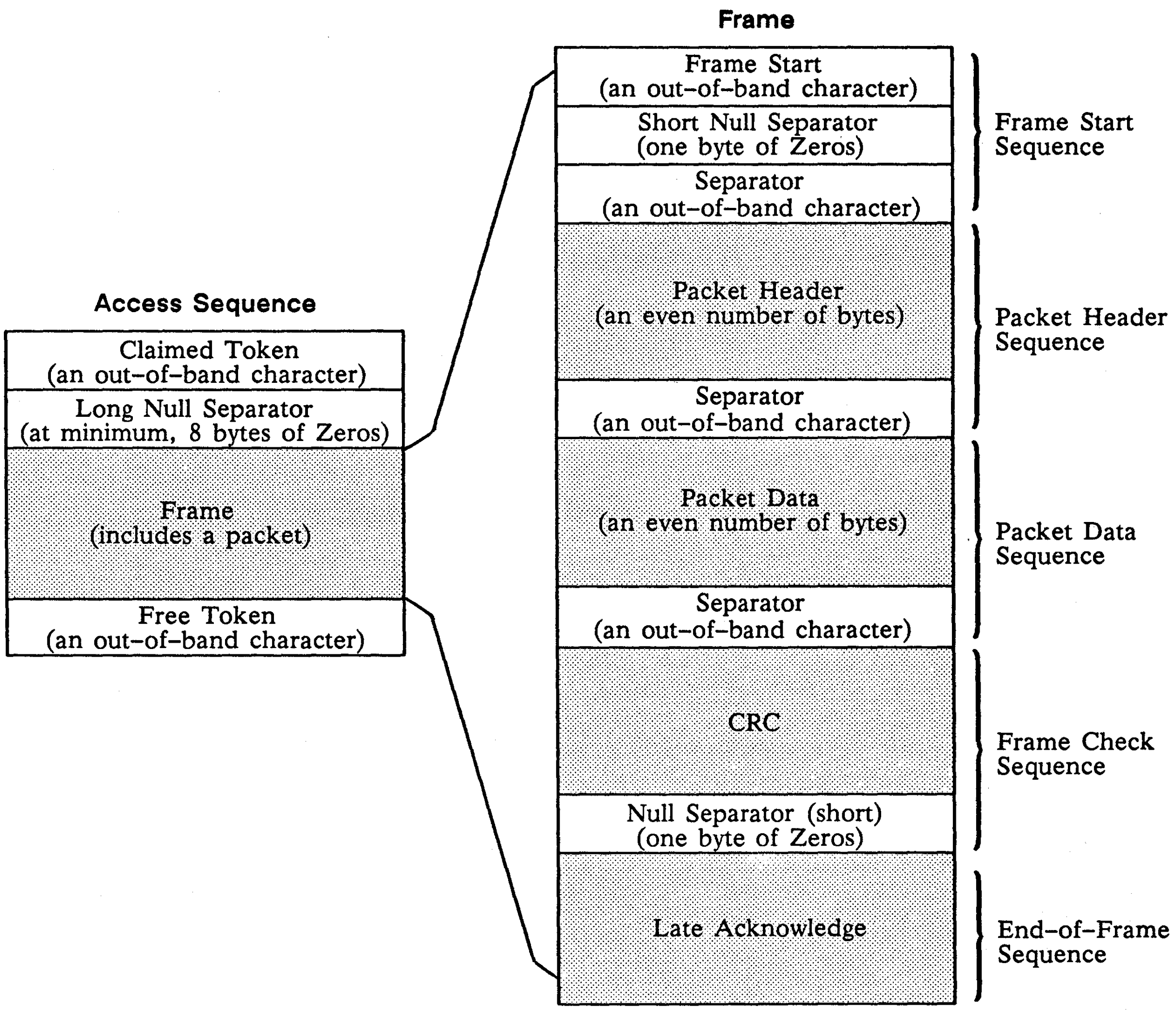

Frame
Frame Start (an out-of-band character)
Short Null Separator (one byte of Zeros)
Separator (an out-of-band character)
Packet Header (an even number of bytes)
Separator (an out-of-band character)
Packet Data (an even number of bytes)
Separator (an out-of-band character)
CRC
Null Separator (short) (one byte of Zeros)
Late Acknowledge
Frame Start Sequence
Packet Header Sequence
Packet Data Sequence
Frame Check Sequence
End-of-Frame Sequence
Access Sequence
Claimed Token (an out-of-band character)
Long Null Separator (at minimum, 8 bytes of Zeros)
Frame (includes a packet)
Free Token (an out-of-band character)

Figure 2-1. Data Formatted According to MAC Layer Protocols

2.2.1 Media Access Control (MAC) Data Formats

Normally, a transmitting node inserts a Zero (this is called bit-stuffing) into the serial bit stream after every five successive Ones. Likewise, a receiving node extracts each Zero that follows five successive Ones. This arrangement means that the receiver reconstructs the original pattern (i.e., before bit-stuffing) of the data.

The presence of the six successive Ones tells a receiver: "the bit-stuffing protocol has been intentionally violated by the transmitter." When the bit-stuffing protocol has been violated, a receiver knows that it is seeing an out-of-band bit pattern (i.e., that this pattern contains control information).

All out-of-band characters begin with a Zero; the leading Zero exists to break a potential string of Ones. To determine what the character means, the receiver then looks at the two least-significant bits within the character (i.e., at the two bits immediately following the six successive Ones). These two bits yield the definitions for specific out-of-band characters:

- Free Token (an out-of-band character used for gaining control of the ring)
- Claimed Token (an out-of-band character used for sending a frame)
- Frame Start Character (an out-of-band character that denotes the start of a frame)
- Separator Character (an out-of-band character that delimits the packet header and packet data fields within a frame).

MAC protocols dictate that out-of-band characters (and sometimes bytes of Zeros, called null separators) must appear in their correct respective positions and that they must contain the appropriate values, or protocol errors will occur. The following text describes each out-of-band character and the null separators.

2.2.1.1 Tokens: Free and Claimed

The *free token* (an out-of-band-character) circulates around the ring. When any node wants to transmit, it changes the free token into a *claimed token* (by changing the state of the character's last bit). The claimed token gives the node that modified it the right to transmit.

For each frame that a node wants to send, it must acquire a free token, and change it to a claimed token. If no token exists on the network (for example, if the ring has broken), any node that wants to transmit can generate a claimed token (after a specified timeout) in order to force transmission.

The bit-alignment of free and claimed tokens must be correct (they must be transmitted most-significant-bit first). Figure 2-2 shows the free and claimed token (out-of-band) characters, respectively.

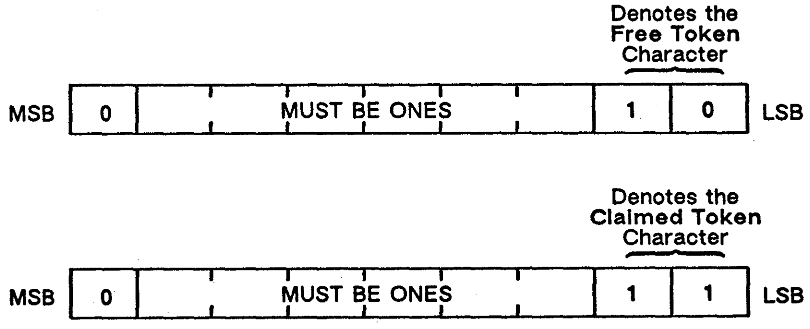

Denotes the Free Token Character
MSB
0
MUST BE ONES
1
0
LSB
Denotes the Claimed Token Character
MSB
0
MUST BE ONES
1
1
LSB

Figure 2-2. Free and Claimed Tokens

2.2.1.2 Frame Start and Separator Characters

The *frame start character* is transmitted first in the frame; the uniqueness of this out-of-band character ensures that it always signals the start of a frame. The frame start character sets up byte synchronization for the entire frame, and its bit alignment must be correct (it must be transmitted most-significant-bit first). Figure 2-3 shows the frame start character.

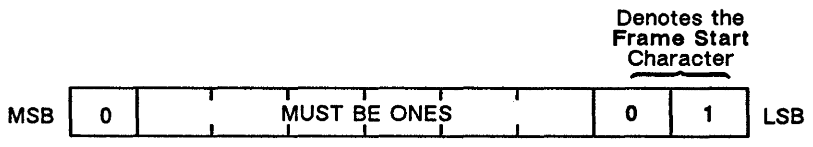


Figure 2-3. Frame Start (Out-of-Band) Character

Three *separator characters* delimit variable-length (packet header and packet data) fields within the frame. Separator characters must occur on even byte boundaries, and their bit alignment must be correct (they must be transmitted most-significant-bit first). Figure 2-4 shows the separator character.

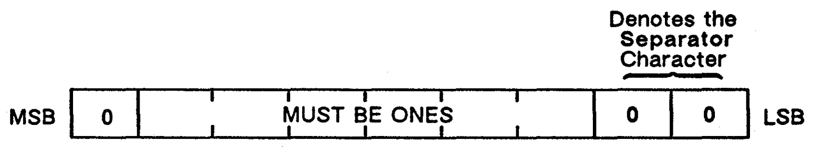


Figure 2-4. Separator (Out-of-Band) Character

2.2.1.3 Null Separators

The *long null separator* transmitted by the sender before the packet consists of a minimum of 8 bytes of Zeros. It gives the transmitter and receiver time to complete those "overhead" tasks that precede a data transfer. For example, the transmitting node needs time to acquire the memory bus so that it can get the message from memory. The receiving node needs time to finish writing a message into memory that it may just have received.

The *short null separator* (a byte of Zeros) occurs within the frame start sequence to help the hardware understand that this is really the start of a frame (and that the frame start character isn't a corrupted token, for example). A short null separator must be present within the frame check sequence, as well.

2.2.2 Media Access Control (MAC) Frames

Refer again to Figure 2–1; it shows a frame and the five sequences into which frames are broken at the MAC layer:

1. Frame Start Sequence
2. Packet Header Sequence
3. Packet Data Sequence
4. Frame Check Sequence
5. End–of–Frame Sequence.

MAC protocols dictate that each sequence in a frame must appear in the appropriate order, and that the data within each sequence must be formatted correctly, or protocol errors will occur. Subsections 2.2.2.1 through 2.2.2.5 describe each of the five frame sequences (and their contents) in detail.

2.2.2.1 Frame Start Sequence

The frame start sequence contains the frame start (out–of–band) character, a null separator, and a separator character. For more information about the frame start character, separator character, and the null separator, refer to subsection 2.2.1.

2.2.2.2 Packet Header Sequence

The packet header sequence contains a packet header and a separator character. The packet header is described here. For more information about the separator character, refer to subsection 2.2.1.

Although a packet header can vary in size from 12 to 1024 bytes, it must always consist of an even number of bytes. The Apollo token ring controller will always transmit the first 12 bytes of a packet header. Since the hardware guarantees that the first 12 bytes of a packet header will be transmitted (even in a broken network), our operating system uses bits within this first 12 bytes of the packet header to force transmission of a "network broken" message to the rest of the ring. This diagnostic technique is known as "beaconing." For more information about beaconing, refer to the "Type Field" later in this subsection.

The controller may abort transmitting the packet header field on any even byte after the 12th byte, if it encounters an error, or if it determines (by checking the packet header's early acknowledge field) that the packet is not being copied by another node. Whenever a transmission is aborted, the transmitter sets the error bit in the late acknowledge field. Figure 2–5 shows the contents of the packet header. The text that follows Figure 2–5 describes each field within the packet header.

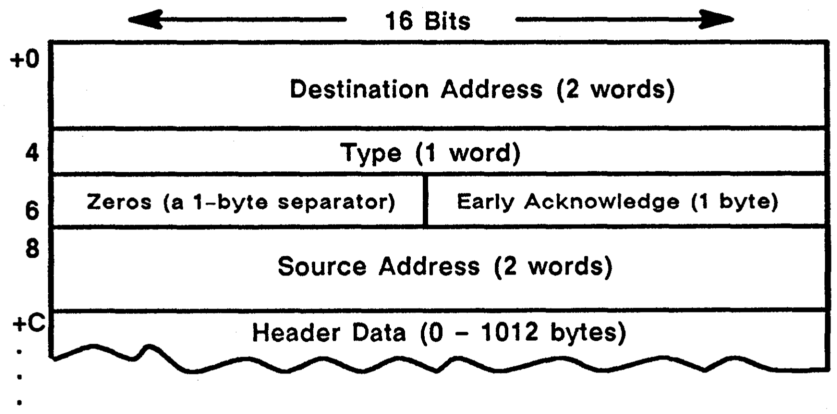

16 Bits
+0
Destination Address (2 words)
4
Type (1 word)
6
Zeros (a 1-byte separator)
Early Acknowledge (1 byte)
8
Source Address (2 words)
+C
Header Data (0 - 1012 bytes)

Figure 2–5. Packet Header (within Packet Header Sequence)

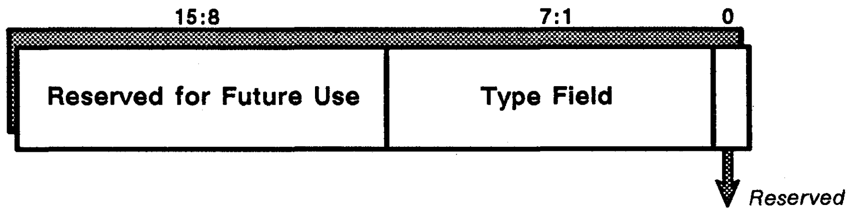
Destination Address Field

The hardware compares the contents of the 32-bit destination address field with the node address of the target. Briefly, a node receives a message if the destination address field matches its node address, or if the broadcast bit in the type field (described next) is set in the packet header. Apollo token ring protocols do not support "grouped" addresses.

Type Field

The type field determines whether a node receives or ignores a message; our operating system modifies the type field to control the flow of information to each receiver. Such software control enables nodes to selectively discard or ignore messages at the hardware level, effectively enhancing ring performance.

When the broadcast bit is set in the type field, the bits in the destination address field are free for beaconing. (Beaconing occurs when our operating system, using information at the physical layer, determines that the cable upstream from a node may be broken.) Figure 2-6 shows and describes the type field.



Broadcast <7>
This bit indicates that a packet is intended for broadcast (to all receivers). If it is set, receivers ignore the destination address field in the packet header.

Currently, our operating system uses the remaining bits in the Type Field like this:

Hardware Diagnostics <6>
This bit is used for diagnostics only.

Thank You <5>
This bit is set to indicate that the packet is a reply (i.e., a response from another node).

Please <4>
This bit is set to indicate that the packet is a request (i.e., a packet's originator is asking for service from the target node).

Paging <3>
This bit is set on page-outs and protocol changes.

User <2>
This bit indicates that the packet is being used for interprocess communications (rather than for the operating system).

Software Diagnostics <1>
This bit identifies packets used for network monitoring and maintenance.

Figure 2-6. Type Field

Early Acknowledge Field

A node's ring transmitter inserts an early acknowledge field; another node's receiver modifies it. The CRC doesn't have to be recalculated each time a receiver modifies the early acknowledge field because ring hardware treats this field as a string of Zeros in its CRC calculation. Figure 2–7 shows and describes the early acknowledge field.

7	6:5	4	3	2	1	0
Must Be Zero	Reserved for Future Use	Must Be Zero	Intend-to-Copy	Reserved for Future Use	Parity	Must Be Zero

Intend-to-Copy <3>
An addressed receiver sets this bit.

Parity <1>
This bit is used for odd parity. When it is set, an odd number of Ones appears in the frame's early acknowledge field.

Figure 2–7. Early Acknowledge Field

Source Address Field

The 32-bit source address field contains the node address of the node that originated the frame.

Header Data

The header data field can vary in size from 0 to 1012 bytes, but it must always consist of an even number of bytes. Software determines the contents of this field. It can include routing and other types of information.

2.2.2.3 Packet Data Sequence

The packet data sequence contains packet data and a separator character. The packet data are described here. For more information about the separator character, refer to subsection 2.2.1.

The packet data field can vary in size from 0 to 4096 bytes, but it must always consist of an even number of bytes. Typically, packet data consists of 1024 bytes. The controller may abort transmitting the packet data field after any even byte, if it encounters an error, or if it determines (by checking the early acknowledge field) that its packet is not being copied by another node. Whenever a packet is aborted, the transmitter sets the error bit in the late acknowledge field. For more information about the late acknowledge field, refer to subsection 2.2.2.5.

2.2.2.4 Frame Check Sequence

The frame check sequence contains a cyclic redundancy field and a null separator. The cyclic redundancy field is described here. For more information about the null separator, refer to subsection 2.2.1.

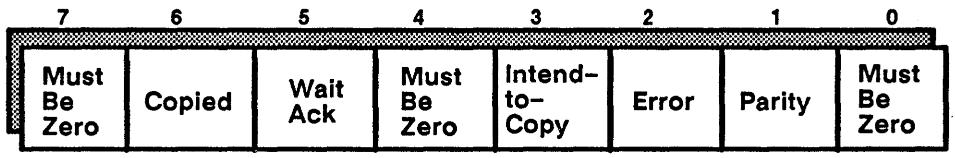
The 32-bit cyclic redundancy check is initialized to zero and based on the following generator polynomial:

$$g(X) = (X^{21} + 1)(X^{11} + X^2 + 1).$$

The sender transmits the CRC most-significant bit first. The receiver calculates the CRC to include the packet header and data sequences, and the separators (treating the early acknowledge field as "Zeros," and ignoring those bits added during the bit-stuffing process).

2.2.2.5 End-of-Frame Sequence

The end-of-frame sequence contains the late acknowledge field. This field tells the sending node whether or not the frame has been received and the packet has been copied. The transmitter inserts a late acknowledge field; the receiver modifies it. Figure 2-8 shows and describes the late acknowledge field.



Copied <6>
A receiver that has successfully copied the packet (without errors) sets this bit.

Wait Ack <5>
An addressed receiver that wasn't enabled to copy the packet sets this bit.

Intend-to-Copy <3>
An addressed receiver that is set up to copy the packet (and whose type field matches) sets this bit.

Error <2>
A station that observes an error in the packet going by sets this bit. Alternatively, a transmitter that aborts the send sets this bit.

Parity <1>
This bit is used for odd parity. When it is set, an odd number of Ones appears in the frame's late acknowledge field.

Figure 2-8. Late Acknowledge Field

Chapter 3

Physical Layer

3.1 Introduction

Data travels over the Apollo token ring at 12 Mb per second. Typically, we use 75–ohm, CATV–type coaxial cable to carry the serial bit stream from node to node, but users can choose alternate media types (fiber–optic cable, for example).

The Apollo token ring is "continuously synchronous." In a continuously synchronous network, all of the nodes participate in maintaining clock and bit synchronization with one another; each node is responsible for making minor frequency adjustments and for regulating its own rate of data transmission (no "master" node exists).

Figure 3–1 shows a functional block diagram of the Apollo token ring controller, representing the major components visible to protocols at the physical layer. The sections that follow Figure 3–1 provide a detailed description of Apollo token ring protocols at the physical layer.

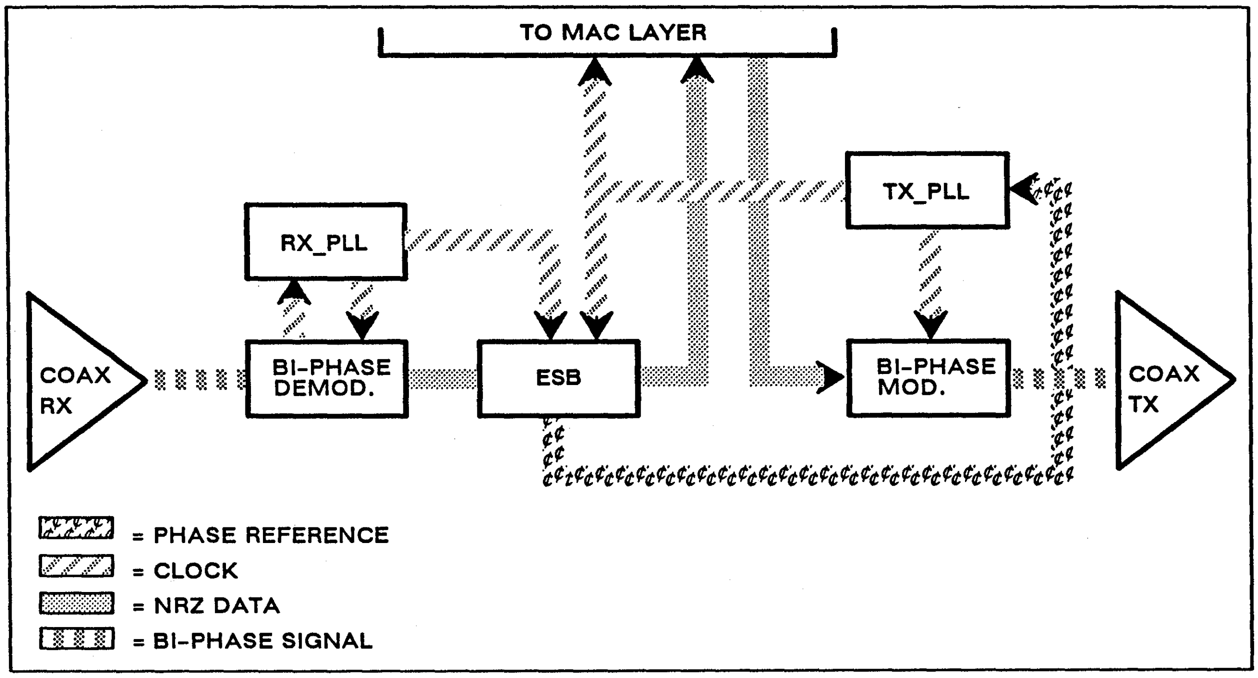


Figure 3–1. Apollo Token Ring Controller Functional Block Diagram

3.2 Data Stream

Nodes in the Apollo token ring transceive. This means that they receive data in, and then immediately send it out (alternatively, they strip it from the ring). For more information about transceiving, refer to Section 1.1. For more information about stripping, refer to Section 2.1.

When transmitting, each node uses the 24-MHz clock generated by its transmit phase-lock loop to encode an NRZ (non-return-to-zero) signal into a bi-phase waveform containing clock and data. When receiving, each node uses the 24-MHz clock generated by its receive phase-lock loop to servo around — and track the transitions contained within — the bi-phase waveform. This enables each node to recover the bi-phase signal and then to accurately decode clock and data information from within it.

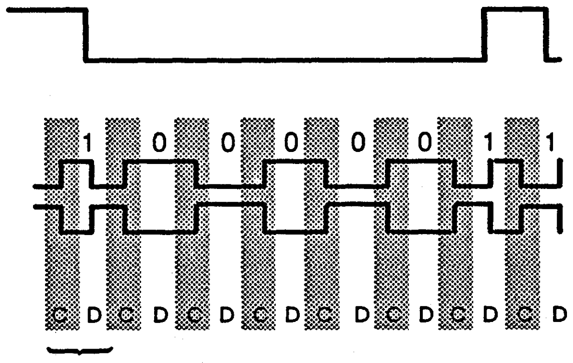
In the time it takes to transmit one bit (this is a bit cell, or 83.33 nsec), two windows exist: the *clock window* and the *data window*. In each clock window, a transition (i.e., a clock signal) must always be present or a bi-phase error will occur and the corresponding data will be interpreted as having a bit value of Zero. In each data window, transitions (or the lack of them) signal bit values. A transition within the data window indicates a bit value of One; no transition within the data window signals a bit value of Zero.

Figure 3-2 shows an NRZ data stream before and after bi-phase demodulation takes place.

An NRZ data stream. It is encoded with clock and data information during bi-phase modulation.

Once encoded, the bi-phase stream can look like either of these waveforms.

The clock signal — generated by the receive phase-lock loop during bi-phase demodulation — appears here in gray.



The bracket shows a bit cell (83.33 nsec). Each bit cell contains two windows: a clock window ("C") and a data window ("D").

Figure 3-2. NRZ Data Stream Before and After Bi-Phase Demodulation

3.3 Network Stability

In order for a ring to be stable (i.e., to maintain clock and bit synchronization), the total delay around the network must be exactly an integral — rather than a fractional — number of bit-times.

Under normal circumstances, delay always exists in the ring. Such delay can be caused by static elements (for example, cable plant and connected nodes) and by dynamic conditions (for example, electrical distortion).

Each node in the Apollo token ring uses its receive and transmit phase-lock loops and an elastic-store buffer to ensure that the total amount of delay within the ring will be an integral number of bit-times. For more information about phase-lock loops and the elastic-store buffer, refer to subsections 3.3.1 and 3.3.2.

3.3.1 Phase-Lock Loops

Phase-lock loops are the means by which the network is kept continuously synchronous. Each node in the ring contains one receive and one transmit phase-lock loop. Since receive phase-lock loops can track over a greater range of frequencies and adjust faster than transmit phase-lock loops, a node's receive phase-lock loop will always be synchronized to the preceding node's transmit phase-lock loop.

In steady state, a node's receive phase-lock loop locks to the frequency and phase of the incoming (received) bi-phase data. A node's transmit phase-lock loop tracks the frequency of the receive phase-lock loop, but the relationship between a node's receive and transmit phase-lock loops may include a phase offset, and always includes some phase jitter.

Phase offset is proportional to the deviation of the received frequency from the transmit phase-lock loop's nominal center frequency (i.e., 24 MHz). Phase offset is at its minimum value (0.5 bit-times or less) at 24 MHz -3 kHz. It increases linearly to its maximum value (1.5 bit-times or more) at 24 MHz +3 kHz.

In the Apollo token ring, the phase offset between the transmit and receive phase-lock loops allows the network to achieve stability (i.e., an integral number of bit-times) without changes in frequency greater than ±3 kHz from nominal center. Because the transmit phase-lock loop tracks the receive phase-lock loop with a damped response, most phase jitter never propagates to the next node. Because the transmit phase-lock loop's phase gain is always less than one, stability around the network is ensured.

In steady state, all nodes in the Apollo token ring settle at the same frequency (i.e., within ±3 kHz of 24 MHz, the nominal center frequency). Each node uses a circular buffer, or FIFO (the elastic-store buffer, described next) to insert some fractional bit delay into the serial bit stream, thus keeping ring length an integer number of bits.

3.3.2 Elastic-Store Buffer

The elastic-store buffer introduces a variable delay into the NRZ serial data stream. Essentially, the elastic-store buffer holds the phase offset between each node's transmit and receive phase-lock loops.

Nominally, when a node's transmit and receive phase-lock loops are in phase, the elastic-store buffer introduces a 1-bit delay. When the phase offset between a node's transmit and receive phase-lock loops exceeds the range of the elastic-store buffer (0.5 bits ≤ ESB delay ≤1.5 bits) — i.e., when frequency excursions greater than ±3 kHz occur — elastic-store buffer errors occur.

Specifically, an elastic-store buffer underflow will occur if a node's transmit and receive phase-lock loops are out-of-phase by 0.5 bit-times or less. An overflow will occur if a node's transmit and receive phase-lock loops are out-of-phase by 1.5 bit-times or more. Under these circumstances, the network is forced to re-initialize at a new operating frequency (24 MHz).

3.4 Signal Characteristics

The coaxial driver's output signal is band–limited with a cutoff frequency of 18 MHz. Transmitted signal power is 18 dBm into 75 ohms (typically, 2.5 V peak–to–peak). The coaxial receiver has a –20 dBm minimum sensitivity, based upon this transmitted signal over a maximum cable length of 1 km between nodes.

For additional information about signal characteristics and coaxial cable for the Apollo token ring, see Appendix A.

3.5 Passive Network Bypass

When powered off or under command of the controller, relays connect a node's input coaxial cable to its output coaxial cable. At the same time, these relays connect the node's transmit output to its receive input. This allows the network to bypass a disconnected node, and enables the node to perform loopback self–tests. The passive network bypass connection provides an insertion loss of $\leq$ 1 dB in the signal band.

Appendix A

Coaxial Driver and Receiver

This appendix describes the coaxial line driver and receiver. It also contains brief specifications for Apollo token ring coaxial cable, and provides pointers so that you can easily find additional information about it.

A.1 Coaxial Driver

The coaxial driver pre-equalizer is a fourth-order Bessel low-pass filter with a cutoff frequency of approximately 18 MHz. The transmitted output power is approximately +18 dBm into 75 ohms. Figure A-1 shows a typical voltage waveform.

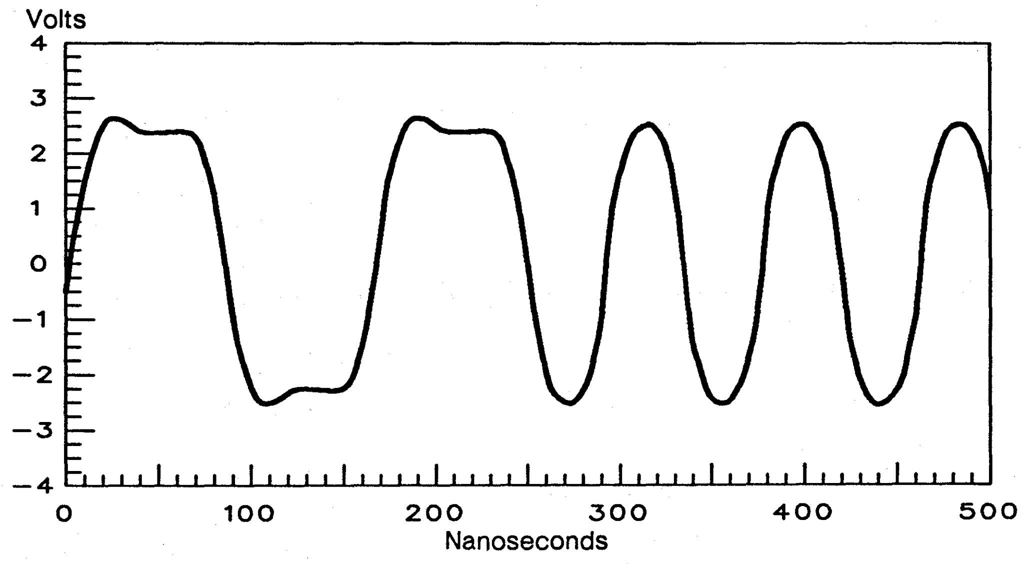


Figure A-1. Typical Voltage Waveform

A.2 Coaxial Receiver

The coaxial receiver provides for line isolation, equalization, amplification, and logic-level conversion of the incoming network signal. The nominal input impedance of this receiver is 75 ohms.

An RF transformer is used for line isolation. The line equalizer/amplifier compensates for intersymbol interference and provides approximately 30 dB of gain in the signal band. The received signal is converted to logic levels suitable for use in the bi-phase demodulator and remaining physical layer circuitry.

A.3 Coaxial Cable

Although it is beyond the scope of this document to provide complete cable specifications, we provide an abbreviated set of them in Table A-1. For additional information about using coaxial cable in the Apollo token ring, see *Installing Coaxial Cable and Accessories for a Domain Token Ring Network* (009860) and *Planning Domain Networks and Internets* (009916).

Table A-1. Apollo Token Ring Coaxial Cable Specifications

Characteristic	**Specification**	
Operating Voltage	30 volts rms minimum	
Delay	1.3 nsec/ft maximum	
Conductor	18 AWG solid copper	
Conductor Resistance	7.5 ohms/304.8 m (1000 ft) maximum	
Shield Resistance	5.2 ohms/304.8 m (1000 ft) maximum	
Impedance	75 ohms	
Capacitance	17.3 pF/ft	
Velocity of Propagation	78%	
Attenuation	MHz / dB Loss/30.5 m (100 ft) 10 / 0.70 (+/-0.10) 50 / 1.40 (+/-0.10) 100 / 2.10 (+/-0.10)	
Jacket Material	**PVC**	**TEFLON-FEP**
Outside Diameter (Jacket)	6.86 (+/-0.25) mm 0.27 (+/-0.01) in.	6.38 (+/-0.25) mm 0.25 (+/-0.01) in.
Temperature Range	10° to 60° C (14° to 140° F)	60° to 200° C (-76° to 394° F)
Conductor Diameter	0.00091 to 0.01016 mm (0.036 to 0.040 in. — 18 AWG)	0.01016 mm (0.014 in. — 18 AWG)
Dielectric	Cellular Foam Polyethylene	Foam TEFLON
Outside Diameter (Dielectric)	45.7 mm (0.180 in.)	45.7 mm (0.180 in.)
Shield Tape	Aluminum/Mylar 100% coverage	Aluminum/Mylar 100% coverage
Shield Braid	Tinned Copper 60% coverage minimum	Tinned Copper 60% coverage minimum

Reader's Response

Please take a few minutes to send us the information we need to revise and improve our manuals from your point of view.

Document Title: *Apollo Token Ring Media Access Control Layer and Physical Layer Protocols*
Order No.: 010005 Revision: 00 Date of Publication: October, 1987

What type of user are you?

_____ System programmer; language ________________________
_____ Applications programmer; language ____________________
_____ System maintenance person _____ Manager/Professional
_____ System Administrator _____ Technical Professional
_____ Student Programmer _____ Novice
_____ Other

How often do you use the Domain system?____

What parts of the manual are especially useful for the job you are doing?________________________
____
____
____

What additional information would you like the manual to include?____________________________
____
____
____
____

Please list any errors, omissions, or problem areas in the manual. (Identify errors by page, section, figure, or table number wherever possible. Specify additional index entries.)_________________________
____
____
____
____
____
____
____

____
Your Name Date

____
Organization

____
Street Address

____
City State Zip

No postage necessary if mailed in the U.S.

cut or fold along dotted line

FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 78 CHELMSFORD, MA 01824

POSTAGE WILL BE PAID BY ADDRESSEE

APOLLO COMPUTER INC.
Technical Publications
P.O. Box 451
Chelmsford, MA 01824

FOLD